Beyond Recommender Systems: Helping People Help Each Other

Loren Terveen and Will Hill

AT&T Labs - Research

Abstract

The Internet and World Wide Web have brought us into a world of endless possibilities: interactive Web sites to experience, music to listen to, conversations to participate in, and every conceivable consumer item to order. But this world also is one of endless choice: how can we select from a huge universe of items of widely varying quality? Computational *recommender systems* have emerged to address this issue. They enable people to share their opinions and benefit from each other's experience. We present a framework for understanding recommender systems and survey a number of distinct approaches in terms of this framework. We also suggest two main research challenges: (1) helping people form communities of interest while respecting personal privacy, and (2) developing algorithms that combine multiple types of information to compute recommendations.

Introduction

The new millennium is an age of information abundance. The 1990s have seen an explosion of information and entertainment technologies, and thus of choices a person faces. People may choose from dozens to hundreds of television channels, thousands of videos, millions of books, CDs, and multimedia, interactive documents on the World Wide Web, and seemingly countless other consumer items presented in catalogs or advertisements in one medium or another. The web in particular offers myriad possibilities – in addition to interactive documents, there are conversations to join and items to purchase. Not only is there a vast number of possibilities, but they vary widely in quality. Evaluating all these alternatives, however, still takes about the same time and effort it always has. Our attention remains as it was – the information explosion has not been accompanied by a leap in human evolution. Therefore, individuals cannot hope to evaluate all available choices by themselves unless the topic of interest is severely constrained.

So what can we do? When people have to make a choice without any personal knowledge of the alternatives, a natural course of action is to rely on the experience and opinions of others. We seek *recommendations* from people who are familiar with the choices we face, who have been helpful in the past, whose perspectives we value, or who are recognized experts. We might turn to friends or colleagues, the owner of a neighborhood bookstore, movie reviews in a newspaper or magazine, or Consumers Union product ratings. And we may find the social process of meeting and conversing with people who share our interests as important as the recommendations we receive.

Today increasing numbers of people are turning to computational *recommender systems*. Emerging in response to the technological possibilities and human needs created by the World Wide Web, these systems aim to mediate, support, or automate the everyday process of sharing recommendations.

We explore the field of recommender systems in the remainder of the paper. The main goal is to identify challenges and suggest new opportunities. We begin by developing a conceptual framework for thinking about recommender systems that builds on everyday examples of recommendation and identifies basic concepts and design issues. The bulk of this chapter is devoted to surveying several distinct approaches to recommender systems and analyzing them in terms of the design issues they address and how they do so. Finally, we suggest a number of challenges and opportunities for new research and applications. Two main challenges are: (1) assisting people in forming communities of interest while respecting privacy concerns, and (2) developing recommendation algorithms that combine multiple types of information.

Recommendation: Examples and Concepts

Everyone can bring to mind examples of recommendation. You might think of reading movie reviews in a magazine to decide which movie to see. Or you might recall visits to your local bookstore, where you've talked to the owner about your interests and current mood, and she then recommended a few books you'd probably like. Finally, you might

remember walking through your neighborhood and noticing that a particular sidewalk cafe always is crowded. You think that its popularity must be a good sign, so you decide to give it a try.

Reflecting on these examples helps to clarify the concept of recommendation. A person is faced with a *decision*, which for our purposes is a choice among a universe of alternatives. The universe typically is quite large, and the person probably doesn't even know what all the alternatives are, let alone how to choose among them[1]. If the person doesn't have sufficient personal knowledge to make the choice, he or she may seek recommendations from others. Recommendation, therefore, is a communicative act.

Recommendation is based on the *preferences* of the recommender (and perhaps of the seeker and other individuals). For our purposes, a preference is an individual mental state concerning a subset of items from the universe of alternatives. Individuals form preferences based on their experience with the relevant items, such as listening to music, watching movies, tasting food, etc. For example, I might prefer vanilla or strawberry ice cream (among ice cream flavors), Bach, Mozart, and Haydn (among classical composers), and Joel Coen and Kevin Smith (among contemporary film directors). Of course, preferences can be more complicated: I might prefer one item over another (The Simpsons over The X-Files) or even think in terms of some scoring system ("on a scale of 1 to 10, Bob Dylan's Highway 61 Revisited is a 10").

A recommendation may be directed to specific individuals or "broadcast" to anyone who's interested. For the person who receives it, a recommendation is a resource that helps in making a choice from the universe of alternatives. The recommendation serves as a view or filter onto the whole, often inaccessible, universe. A recommendation may be based not just on the recommender's preferences, but also on those of the recommendation seeker. For example, in recommending books to you, I might find out which genres you like (e.g., Science Fiction) and even which books you've really enjoyed (e.g., Robinson's *Mars* trilogy). I then can recommend books that are both good (in my opinion) and will meet your preferences. I even can recommend books based on the preferences of others: maybe I'm not a science fiction fan, but I have friends that are, so I can make recommendations based on what they like. Further, I may put you in touch with people who share your interests: maybe there's a Science Fiction reading group you might like to join. Finally, a recommendation may include explanatory material that helps the recommendation seeker evaluate it (why you'd like the Mars trilogy, what's good about The Simpsons, and why it's better than The X-Files, etc.).

A Model of the Recommendation Process

Figure 1 summarizes these concepts and situates them in a general model of recommendation. A recommendation seeker may ask for a recommendation, or a recommender may produce recommendations with no prompting. Seekers may volunteer their own preferences, or recommenders may ask about them. Based on a set of known

[1] Some researchers have characterized a user's path through a space of alternatives as a process of *navigation*. The book edited by Munro, Höök, and Benyon [32] collects a set of papers written from this perspective; the contribution of Dourish [14] discusses the navigation metaphor clearly.

preferences – his/her own, the seeker's, and those of other people, often people who received recommendations in the past – the recommender recommends items the seeker probably will like. In addition, the recommender may identify people with similar interests. The seeker may use the recommendation to select items from the universe or to communicate with like-minded others.

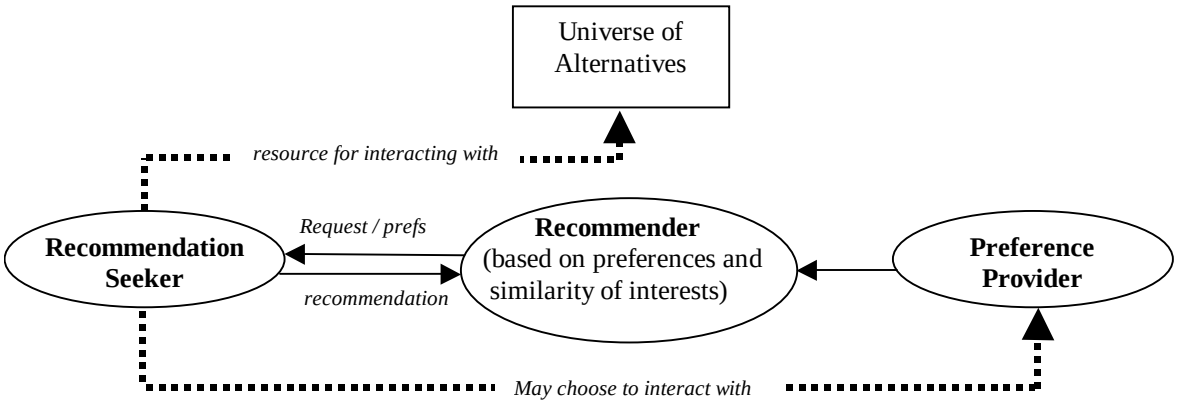

Figure 1: Model of the Recommendation Process

This model is intended to be general enough to cover a broad range of recommendation activities. Real activities may vary significantly; in particular, they may not instantiate some aspects of the model. For example, movie reviewers publish their reviews based on their own preferences, without any specific knowledge of reader preferences or explicit requests. In a case like the "crowds at the sidewalk café" example, the recommendation activity itself may seem to disappear. The preferences of a group of people (the diners) are directly visible to all who pass by, and thus can be used to select restaurants to visit. (As we shall see, in computational analogues, the recommender can't quite disappear. Computation plays a vital, though perhaps hidden role in making preferences visible.) Sometimes seekers aren't interested in communication with others – all they want is a good book to read – while in other cases, that's the whole point. Finally, the structure and content of recommendations vary from quite complex – e.g., movie reviews in *Entertainment Weekly* consist of a few hundred words of text, a letter grade, and sometimes ratings on specific features such as "language", "violence", and "nudity" – to quite simple – e.g., a list of recommended movies.

Issues for Computational Recommender Systems

A computational recommender system automates or supports part of the recommendation process. An automated recommender system assumes the recommender role: it offers recommendations to users based on their preferences (and perhaps also based on the preferences of other people as well). A recommendation support system makes it easier for people to create and share recommendations.

We can identify four main issues to characterize the design space for recommender systems. The issues concern preferences, roles & communication, algorithms, and human-computer interaction. We introduce each briefly at this point.

Preferences

Recommendation is based on preferences. Thus, an automated recommender system must obtain preferences from people concerning the relevant domain. This raises a number of questions, including:

- Whose preferences are used? Those of the person seeking the recommendation, those of previous users of the system? Or perhaps preferences expressed by people in an altogether different context, such as a public forum (e.g., a chat room, bulletin board, or newsgroup)?
- How are preferences obtained? For example, do recommendation users have to express their own preferences as part of the process of seeking a recommendation? Are preferences expressed explicitly or implicitly (as with the "popular restaurant" example given above)?
- What incentives are there for people to offer preferences?
- What is the form of a preference? How are preferences represented?

Roles & Communication

- Is the recommender role filled by a computational system or a person? If the latter, what role does computation play in supporting the recommendation process?
- Do people play distinct roles, or do all users of a system play the same role? Are roles fixed, or do they evolve?
- How is the interaction between the recommendation user and the recommender initiated? Who initiates it? Is the recommendation directed to a specific person or is it broadcast to anyone who's interested? Is there opportunity for recommendation users to give feedback to the recommender?
- What information about the people whose preferences are used in computing a recommendation is revealed to the recommendation user? Is there an opportunity for communities of like-minded people to form? If information about preference providers is revealed, are any measures taken to safeguard privacy?

Algorithms for Computing Recommendations

- How does an automated recommender system determine whose preferences to use in computing a recommendation? If we think of all the people who have expressed their preferences for a given domain as being placed in a large, multi-dimensional space, this is the problem of finding *neighbors* in that space for the person seeking a recommendation.
- How are recommendations computed? For example, given that a set of neighbors for the recommendation seeker has been determined, how are the preferences of these neighbors weighted and combined?

Human-Computer Interaction

- How are recommendations presented to the person who sought them? The most simple and common example is an ordered list. More complicated examples include

2D and 3D visualizations, as well as visual annotations of existing information spaces.

Major Types of Recommender Systems

Many different recommender systems were developed during the 1990s. Terminology proliferated, too, with labels such as "collaborative filtering", "social filtering", and "social navigation" used to describe various bodies of work. We attempt to make sense of the field by characterizing different approaches in terms of the four issues introduced above. The approaches[2] can be distinguished by which of the four main issues they focus on, and how they address the issues.

- *Content-based* systems use only the preferences of the seeker; they attempt to recommend items that are similar to items the user liked in the past. Their focus is on algorithms for learning user preferences and filtering a stream of new items for those that most closely match user preferences.
- *Recommendation support* systems do not automate the recommendation process; thus, they do not have to represent preferences or compute recommendations. Instead, they serve as tools to support people in sharing recommendations, helping both those who produce recommendations and those who look for recommendation.
- *Social data mining* systems mine implicit preferences from computational records of social activity, such as Usenet messages, system usage history, citations or hyperlinks. These systems also have focused on the HCI issues involved in visualizing the results. These visualizations often have been presented to aid the navigation of information spaces like the World Wide Web; this helped motivate the term *social navigation*.
- *Collaborative filtering* systems require recommendation seekers to express preferences by rating a dozen or two items, thus merging the roles of recommendation seeker and preference provider. These systems focus on algorithms for matching people based on their preferences and weighting the interests of people with similar taste to produce a recommendation for the information seeker.

[2] The proliferation of approaches has meant that there is no accepted clustering of approaches, nor accepted names for the approaches (*content-based* systems are the exception to this rule). With *recommendation support* and *social data mining*, we chose descriptive terms that accurately characterized the approach. With *collaborative filtering*, we chose a term that originally was used more generally, to refer to *all* (social) recommender systems. However, gradually the term *recommender system* has become preferred, perhaps at the urging of Resnick and Varian [37]. The term collaborative filtering still is used, typically in a narrower sense; it is this narrower sense that we use.

Issues	*Approaches*			
	Content-based	**Rec. Support**	**Social Data Mining**	**Collaborative Filtering**
Preferences	Seeker's preferences only		Mines preferences; seeker's preferences typically not used	Seekers must state preferences
Roles & Communication	System automates Role asymmetry	System supports human recommenders and seekers	System automates High potential for community; raises significant privacy concerns Role asymmetry vs. role uniformity	
Algorithms	Machine learning, information retrieval		Data mining	Preference matching and weighting
HCI			Visualization; visual annotation	

Table 1: Recommender Systems Issues and Approaches

Table 1 summarizes the four main approaches and the issues they focus on. We consider the four main approaches in more detail in the remainder of the paper. We characterize the different ways they support the recommendation process and identify challenges and opportunities for future work.

Content-based Recommenders

Content-based recommenders [26, 27] build on the intuition "find me things like I have liked in the past". They *learn* preferences through user feedback. The feedback may be *explicit* – for example, users may rate items as "good" or "bad". Or the feedback may be *implicit* – for example, based on whether users choose to read a recommended document and how much time they spend reading the document. Preferences are represented as a *profile* of user interests in particular types of content, often expressed as a set of weighted keywords. Techniques from machine learning and information retrieval are applied to learn and represent user preferences.

Content-based recommenders are not the primary concern of this chapter. However, they serve as a point of contrast that helps clarify the type of system we are interested in, *social* recommender systems. Social recommender systems create a mediated (perhaps indirect) interaction between a person seeking a recommendation and a set of people who previously have expressed relevant preferences.

Content-based and social recommenders have complementary strengths. For example, if, in the past, you have liked books about the exploration of Mars, you're likely to be interested in a new book about Mars, independent of a recommendation from anyone else. (In other words, this book can be recommended based on its *content*.) On the other hand, a friend may recommend a book on a completely new subject, say, on the role of disease in deciding the outcome of battles throughout history. If you take the recommendation and like the book, you may find yourself developing a completely new

interest. This potential for serendipity is very important, since it may help people break out of a rut and broaden their horizons.

There also is a crucial difference from a systems point of view; content-based recommender systems must be able to represent and manipulate the content of items. This is technically challenging even in the most well understood case, i.e., for text, and currently is virtually impossible for non-text items, such as music or images. However, social approaches have no such problem, since they don't (have to) process content at all; instead, they work with user preferences, opinions, and behaviors. Because of the complementary aspects of content-based and social recommenders, an attractive research tactic is to create hybrid systems. We discuss this type of work later.

Recommendation Support Systems

Recommendation support systems are computational tools to support people in the natural activity of sharing recommendations, including both producing and finding them.

Researchers at Xerox PARC developed Tapestry, the first recommendation support system[3] [15]. Tapestry was an electronic messaging system that allowed users to either rate messages ("good" or "bad") or associate free text annotations with messages. Messages were stored in a database, and could be retrieved based not only on their content, but also on the opinions of others. For example, one could retrieve documents rated highly by a particular person or persons, or could retrieve documents whose annotations contained particular keywords.

Maltz and Ehrlich [30] further developed this approach. They observed existing practice within organizations and noticed that a few individuals always played a very active role in making recommendations. They built a system designed expressly to support the two distinct roles of recommendation producer and user (or seeker). Their system enabled people to create recommendations consisting of pointers to documents, which could be organized into "digests". The recommendations then could be directed to specified colleagues. The system also supported recommendation users in reading these digests.

Similar ideas have been popular on the World Wide Web since its origin – the early incarnation of the personal home page, with its "Cool Links", was the prime example. More recently, however, this activity has matured and evolved. As it has done so, it gained a new name – *weblogs* – and much attention [7, 22]. More and more people are creating annotated logs of links. They select links and write annotations to reflect their unique interests and perspectives. Many weblogs are updated daily, so content is naturally ordered chronologically, but some also offer topical categorizations of links. Some weblogs are done entirely by a single individual, some are a group effort, and some fall somewhere in the middle. Some have thousands of readers, while others have only a few. Some are strongly focused on a single topic (e.g., new media), but most tend to be fairly eclectic (not surprisingly, since they tend to encompass whatever their editors find interesting, and most people have more than one interest).

[3] Tapestry usually is considered the first recommender system of any sort.

About.com commercializes a related notion, that of a human topic guide. The About.com site hosts hundreds of topic-specific areas, each maintained by a human topic expert. The content in each topic area includes organized and annotated collections of web links, news, polls, FAQs, and other "community" features, as well as commercial features.

Recommendation support systems follow existing practice quite closely. They don't posit new roles or new activities. Rather, they build on a well attested, naturally occurring division of labor: a few people are highly motivated to produce recommendations, while most people, most of the time, prefer to use them.

Recommendation support systems are effective when there are enough people who are willing to put in the effort of finding and recommending information. People almost never are paid to do this; usually, it's a labor of love. Additionally, the needs of both recommenders and users must be met. For recommenders, primary needs are recognition and feedback; eliciting grateful, encouraging, or provocative responses often is all (and just what) recommenders want. For users, the recommendations they receive must be relevant and interesting. To have your mailbox flooded with messages that a friend finds interesting but you don't is just a personalized version of spam.

People such as weblog editors who go into the "recommendation business" have several needs. First, keeping their weblogs useful over periods of time confronts them with a significant information management task. They need to check for stale links. They also need to provide a non-chronological organization of content, e.g., developing content categories into which links can be placed, or indexing the content and providing a search engine. Second, they may need recommender systems to suggest new and interesting items that fit their theme. Finally, techniques to help recommenders find the right audience are crucial. The proliferation of viewpoints, as represented in the growing number of weblogs, almost seems to guarantee that every recommender can find an audience, and every information seeker can find a like-minded guide. However, the familiar specter of information overload soon appears: with more and more choices, how do people find each other?

Social Data Mining

The motivation for this approach goes back at least to Vannevar Bush's *As We May Think* essay [9]. Bush envisioned scholars blazing trails through electronic repositories of information and realized that these trails subsequently could be followed by others. Everyone could walk in the footsteps of the masters. In our work, we have formulated a similar intuition using the metaphor of a path through the woods. However, this metaphor highlights the role of collective effort, rather than the individual. A path results from the decisions of many individuals, united only by where they choose to walk, yet still reflects a rough notion of what the walkers find to be a good path. The path both reflects history of use and serves as a resource for future users.

Social data mining approaches seek analogous situations in the computational world. Researchers look for situations where groups of people are producing computational records (such as documents, Usenet messages, or web sites and links) as part of their normal activity. Potentially useful information implicit in these records is identified, computational techniques to harvest and aggregate the information are invented, and

visualization techniques to present the results are designed. Thus, computation discovers and makes explicit the "paths through the woods" created by particular user communities.

The "history-enriched digital objects" line of work [18, 19] was a seminal effort in this approach. It began from the observation that objects in the real world accumulate *wear* over the history of their use, and that this wear — such as the path through the woods or the dog-eared pages in a paperback book or the smudges on certain recipes in a cookbook — informs future usage. *Edit Wear* and *Read Wear* were terms used to describe computational analogues of these phenomena. Statistics such as time spent reading various parts of a document, counts of spreadsheet cell recalculations, and menu selections were captured. These statistics were then used to modify the appearance of documents and other interface objects in accordance with prior use. For example, scrollbars were annotated with horizontal lines of differing length and color to represent amount of editing (or reading) by various users.

The World Wide Web, with its rich content, link structure, and usage logs, has been a major domain for social data mining research. A basic intuition is that a link from one web site to another often indicates both similarity of content between the sites and an endorsement of the linked-to site. Various clustering and rating algorithms have been designed to formalize this intuition. Kleinberg's algorithm [24] is a well-known example. In the commercial world, the Google search engine (www.google.com) uses a similar link analysis algorithm to group and order URLs. Other work has focused on extracting information from web usage logs. Footprints [42] records user browsing history, analyzes it to find commonly traversed links between web pages, and constructs several different visualizations of this data to aid user navigation through a web site. Chalmers and colleagues [11] tale the activity *path* – e.g., a sequence of URLs visited during a browsing session – as the basic unit. They have developed techniques to compute similarities between paths and to make recommendations on this basis – for example, to recommend pages to you that others browsed in close proximity to pages you browsed. Other techniques extract information from multiple sources. For example, Pirolli, Pitkow, and Rao [33, 34] combined web links with web usage data and text similarity to categorize and cluster web pages.

Other work has focused on extracting information from online conversations, such as Usenet. PHOAKS [18] mines messages in Usenet newsgroups looking for mentions of web pages. It categorizes and aggregates mentions to create lists of popular web pages for each group. Donath and colleagues [41] have harvested information from Usenet newsgroups and chats and have used them to create visualizations of the conversation. These visualizations can be used to find conversations with desirable properties, such as equality of participation or many regular participants.

Discussion

Social data mining systems do not require users to engage in any new activity; rather, they seek to exploit user preference information implicit in records of existing activity. Like recommendation support systems, they work in situations where people naturally take on different roles, i.e., a few produce and share opinions and preferences, while most people are content to use this information when they have a need for it.

Systems can preserve and transmit information about the activity context from which preferences were extracted. This can lead to more informative and potentially more useful recommendations. It also creates opportunities for community building – people can be put in touch with others who share their preferences. However, unlike recommendation support systems, which assist people who *intend* to share recommendations, social data mining systems extract preference data from contexts where the providers may have had no such intention. Thus the opportunities for community building must be balanced against a consideration of the privacy of the people who produced the preferences in the first place. We discuss this challenge later.

Most social data mining systems create "broadcast" recommendations; that is, the recommendations are made available (perhaps as visualizations and navigation aids) to anyone who uses the system. However, nothing about the approach forces this: if preferences are extracted and associated with the people who produced them, an algorithm can match users based on their preferences and thus compute personalized recommendations. The system of Chalmers et al [11] is one example of a system that does this.

Issues

The first set of issues concern the data that is mined and the mining algorithms. We refer to our experience with the PHOAKS and TopicShop systems to illustrate the issues.

- *Is there useful data (i.e., preferences) hidden in the activity records*? Experiments we ran as part of the PHOAKS project in 1996 showed that about a quarter of all Usenet messages contained mentions of URLs, and about 20% of the time people mentioned a URL, they were expressing a preference for it. This means that there are many thousands of URL recommendations in Usenet every day. Therefore, the next challenge is:

- *Can the data be extracted, accurately and efficiently*? In the PHOAKS experiments, we showed that our rules for classifying mentions of URLs as recommendations were nearly 90% accurate (in both *precision* and *recall*[4]).

- *Is the extracted data of high quality*? We wanted to know whether the URLs PHOAKS recommended for a given topic actually were *good* recommendations. Specifically, we asked whether our ordering metric – which assigned one vote to an URL for each distinct person who recommended it – accorded with human judgements of quality. We showed a positive correlation between this metric and the probability that a given URL was included in a FAQ (Frequently Asked Question list) for the relevant newsgroup. In other words, the more people who recommended a URL, the more likely an individual topic expert was to have included it in a list of relevant resources.

[4] Precision and recall are well-known information retrieval metrics. Precision is the proportion of items that a system classifies as being in a given category that actually do belong to that category (according to prior human judgement). Recall is the proportion of items known to belong to a given category that the system classifies as being in that category.

We also investigated this question for the TopicShop system [4, 40]. TopicShop mines information from web pages and links, and its interface provides users easy access to this interface. Experiments showed that TopicShop subjects were able to select about 80% more high-quality sites, while taking less time and considering fewer sites than users who did not have access to this data.

Other issues arise when recommendation seekers are interested as much or more in finding a person as in finding information. ReferralWeb [23] analyzes web documents, processes co-occurrence of names within documents to create a social network, and associates people with their expertise. It can answer queries like "Find documents on collaborative filtering written by people who are socially close to Loren Terveen." McDonald and Ackerman's system [31] analyzes software artifacts and documents to associate individuals with specific code modules. Help desk personnel can then be directed to people who are likely to have expertise about specific aspects of the code. Other systems like PHOAKS, the Designer Assistant [39], and Answer Garden [1, 2] present information first, but then allow users to get in touch with the people responsible for the information.

This issue relates to the large body of work on *awareness* in collaborative systems [13]. Recommender systems can address the issue of *who should be made aware of whom*, i.e., how to form communities. Maglio et al [29] identified a number of different techniques for defining communities or "places" on the web. Communities can be formed from people from the same organization, users who are browsing the same or closely related pages, or users with similar browsing histories [11].

Collaborative Filtering

You would expect to get the best recommendation from someone with similar taste. The problem, though, is how to find such a person. You may have to engage in many interactions with lots of different people, through which you slowly learn about each others' preferences, before you start to receive recommendations you are confident in.

Collaborative filtering explores technique for matching people with similar interests and then making recommendations on this basis. Three pillars of this approach are (1) many people must participate (making it likely that any given person will find others with similar preferences), (2) there must be an easy way for people to represent their interests to the system, and (3) algorithms must be able to match people with similar interests.

Collaborative filtering has made the user task quite simple: you express your preferences by rating items (like books or movies or CDs) that the system presents to you. These ratings then serve as an approximate representation of your taste in this domain. The system then matches these ratings against ratings submitted by all other users of the system. The result is the set of your "nearest neighbors"; this formalizes the concept of people with similar taste. Finally, the system recommends items that your nearest neighbors rated highly that you have not rated (and presumably are thus not familiar with); a key issue is how to combine and weight the preferences of your neighbors. You can immediately rate the recommended items if they do not interest you; therefore, over time, the system acquires an increasingly accurate representation of your preferences.

Seminal collaborative filtering systems included GroupLens [35], the Bellcore Video Recommender [20], and Firefly [38]. The systems varied in how they weighted the ratings of different users (i.e., determined who your neighbors were and how close they were) and how they combined the ratings.

Collaborative filtering has found many applications on the web. Electronic commerce sites such as Amazon.com and CDNow feature recommendation centers, where, in addition to expert reviews, users can rate items and then receive personalized recommendations computed by a collaborative filtering engine. User preference also is inferred from site usage: for example, purchasing a book may be taken as evidence of interest not just in that book, but also in the book's author.

Discussion

The primary strength of collaborative filtering is that recommendations are *personalized*. To the extent that your nearest neighbors really have similar taste, you can find out about items you wouldn't have thought of on your own that you are quite likely to find interesting. Second, you don't have to go looking for a recommendation or recommender – you simply state your preferences and receive recommendations. Finally, from a computational view, the data representation is simple and uniform – a user-item matrix whose cells represent ratings – and thus is amenable to many different computational manipulations.

Collaborative filtering does not simply support an existing activity. Instead, it requires users to engage in a somewhat novel computationally mediated activity. This activity has a single combined role, the recommendation seeker / preference provider. We describe this as *role uniformity*. Everyone does the same work (rates items) and receives the same benefits (gets rated items as recommendations). We might describe rating items as an "ante" – to get recommendations, you have to give them. This leads naturally to growth in the system's knowledge (and thus to better recommendations), since using the database leads to the database being updated [20].

Role uniformity has both good and bad aspects. On the one hand, observed practice suggests that most people don't want to offer recommendations; instead, they just want to make use of them. On the other hand, rating items is not particularly onerous work, and you do this work precisely when you want a recommendation.

Finally, collaborative filtering separates out personal contact from the recommendation process [20] – there need be no contact between recommendation producer and receiver. Of course, if the system designers wish, the results of the matching algorithm can be used to introduce people to their nearest neighbors. Indeed, this is an excellent technique for community formation, since people can be linked automatically with others who share their interests.

Issues

There are several technical challenges for collaborative filtering algorithms, including the "first rater" and "sparsity" problems [6, 16]. No recommendation for an item can be offered until someone has rated it. Further, if the number of people who have rated items is relatively small compared to the number of items in the database, it is likely that there won't be significant similarity between users. This in turn means that nearest neighbors

really won't be all that near, thus recommendations won't be all that good. These problems become more urgent as the number of items increases.

One major tactic for addressing these problems is to combine collaborative filtering with content-based recommenders. A simple example can illustrate the benefits of such hybrid systems. For example, suppose one user has rated the NBA page from ESPN.com favorably, while another has rated the NBA page from CNNSI.com favorably. Pure collaborative filtering would find no match between the two users. However, content analysis can show that the two items are in fact quite similar, thus indicating a match between the users. The Fab [6] system builds on this intuition. It analyzes the content of items that users rate favorably to build content-based profiles of user interest. It then applies collaborative filtering techniques to identify other users with similar interests. In another effort, the GroupLens research group is experimenting with using collaborative filtering as a technique to combine the opinions of other users and personal information filtering agents [16].

Other researchers have analyzed the problem of incentives (a generalization of the "first rater" problem) theoretically. Again, the issue is why I should rate first and get no benefit, when I can wait for others to rate so I do benefit. Avery et al [5] carried out a game theoretic analysis of incentive systems to encourage optimal quantity and order of ratings.

Billsus and Pazzani [8] took another approach to addressing problems with collaborative filtering. They observed that the task of predicting items a user would like based on other user's ratings for these items can be conceptualized as *classification*, a well-investigated task within the machine learning community. They take the singular value decomposition of the initial ratings matrix to extract features, then apply a learning algorithm such as a neural network. By exploiting "latent structure" in the user ratings (as Latent Semantic Analysis [12] exploits latent structure in text), the system greatly reduces the need for users to rate common items before one user can serve as a predictor for another. Experiments showed that this approach significantly outperformed previous collaborative filtering algorithms.

Recently Aggarwal et al [3] invented a new graph-theoretic approach to collaborative filtering that appears to avoid some of the limitations of previous algorithms. In particular, it can compute more accurate recommendations given sparse data.

Like any system that offers results to people on the basis of significant computational processing, a collaborative filtering system faces the issue of *explanation* – why does the system think I should like this item? Herlocker [17] proposes techniques for explaining recommendations computed by a collaborative filtering system and experiments for evaluating the efficacy of the explanations.

A final important issue concerns the notion of *serendipity*. Stated informally, I want a recommender system to "tell me something I don't already know." Many current systems fail this test. For example, one of the authors of this paper (Terveen) uses Amazon.com's recommendation center. After Terveen rated a number of items, the system recommended Shakespeare's *MacBeth*, which he rated positively (by indicating "I own it"). At this point, the system began to recommend more Shakespeare plays, such as *King Lear*, *Hamlet*, and *Twelfth Night*. It seems unlikely that someone who is familiar

with *any* of Shakespeare's work will be unaware of the rest of his plays. Thus, these recommendations carried no new information.

Such situations are common. To generalize the Shakespeare example, it may seldom be useful to recommend books by an author to someone who already has rated books by that author highly. An analogous argument can be made for CDs and artists. In fact, the argument can even be strengthened. If someone rates CDs by Nirvana highly, that person is highly likely to already have an opinion about CDs by Hole and Foo Fighters (because of overlap and/or relationships between members of these groups).

Thus, a system can be improved through knowledge of correlations in user opinions about items, i.e., if a user has an opinion about item X, it is quite likely that he or she already has an opinion about item Y. One approach to this problem is to build in knowledge about the items, essentially creating a hybrid content-based and collaborative system. Aggarwal et al's algorithm incorporates a hierarchical classification structure which can be used to make so-called "creative" recommendations that span categories. Perhaps this scheme also could serve as the basis for making serendipitous recommendations, too. A more challenging (but ultimately more scalable) approach is to invent algorithms that can determine these sorts of correlations automatically. Experimenting with the technique of Billsus and Pazzani may be a promising place to start.

Current Challenges and New Opportunities

We close by considering several current challenges for recommender systems. The first set of challenges concerns issues of bringing people together into communities of interest. A major concern here is respecting people's privacy. The second challenge is to create recommendation algorithms that combine multiple types of information, probably acquired from different sources at different times.

Forming and Supporting Communities of Interest

Naturally Occurring Communities as Laboratories

A first challenge is to study, learn from, and better support naturally occurring communities. For example, where we have spoken of only a few roles – recommendation seeker, recommender, and preference provider – and have treated them as distinct, neither of these assumptions may hold in real communities. Observations of weblogs illustrate this.

Slashdot.com is a well-known weblog / web community whose slogan is "News for Nerds". Topics such as Linux, Java, and open source software are core interests. Slashdot was started by a few people as a place to collect and discuss information they found interesting. It grew rapidly, and soon fell prey to the very problems of information overload a weblog tries to avoid. Dozens of stories and thousands of comments are posted each day – too much for anyone to read.

Slashdot's editor has developed an interesting moderation mechanism (see http://slashdot.com/moderation.shtml) to cope with the problem. "Good" Slashdot participants are given limited moderation powers. For a limited amount of time, they can

rate a few comments as good or bad, thus incrementing or decrementing a score. Readers can set filters to see only content with a certain score.

Slashdot is a community with multiple, transient, and shifting roles. Rather than a set of recommendations produced by a few people and consumed by others, it serves as a community notebook, a medium in which all participants can propose and comment on ideas. Communities like Slashdot should serve as laboratories for researchers to study and conduct experiments.

Human participants in such communities may need recommender systems. If you want to maintain a FAQ or if you contribute to a weblog, it does not matter how motivated you are – it is impossible for you to read and evaluate all the information on the web for any reasonably broad topic. You – a human recommender – need a recommender system.

Different types of recommender systems could be used to suggest content. A content-based recommender could observe the documents an editor considers for inclusion, note which ones he or she selects and rejects, and gradually evolve a filter to capture these preferences. A social data mining systems can mine relevant newsgroups or continuously crawl and analyze relevant web localities for new or popular items. With a social recommender systems, the editor gets access to the opinions of many different individuals; this is a sort of "œommunity pulse". Thus, he or she might come across new ideas and information. With the content-based recommender, on the other hand, the editor will get suggestions for items that are like items he or she has selected in the past. This may lead to a more coherent, but narrow and static offering of information.

Forming Communities Automatically – While Respecting Privacy

Recommender systems can link people based on shared interests. Systems that mine preferences from activity records can choose to convey the identity of people who produced preferences. Collaborative filtering systems may communicate the set of "neighbors" that were used to compu te a recommendation. In either case, users of the system have the opportunity to contact and form a community with others who share their interests. However, this opportunity raises significant privacy concerns.

These concerns are more acute for social data mining systems, since they extract information from its original context. Consider PHOAKS as an example: it extracts preferences concerning web pages from newsgroup messages, and aggregates these preferences into recommendations that are made available on a web site. Presumably, most of the people who view the web site were not participants in the original newsgroup. Neither the way in which the information was processed nor the new audience which can access the results was intended or foreseen by the original producers of the information.

A system designer has various choices for balancing individual and group privacy against opportunities for expanded contacts between people. First, one can "play it safe" – that is, present only information that has been aggregated and decontextualized. For PHOAKS, this could mean presenting only ordered lists of recommended URLs, with no information about the recommending messages or the persons who posted the recommenders. However, this both results in less rich and informative recommendations and gives up on the opportunity for people to make new contacts with others who share their interests. Second, one can make it all explicit. For example, we could have

designed PHOAKS to make the identify of recommenders prominent and make it technically easy to contact a recommender (by including mailto: links). We chose not to go this far. We included the email addresses of recommenders, but not at "top-level". If PHOAKS users want to find this information, they have to dig around a bit, and, if they want to email recommenders, they must explicitly cut and paste email addresses.

More generally, we are interested in middle ground solutions that lie between the extremes of complete disclosure and complete anonymity. A good place to start is with techniques used in places such as online "personals" or dating services. In these cases, the system is a trusted intermediary, mediating interaction between people. Participants can progressively reveal more about themselves, perhaps beginning only with their system login, then their personal email address, then other information as they become comfortable.

Combining multiple types of information to compute recommendations

Authority/expertise, not just similarity of taste

The basis of collaborative filtering algorithms is matching people based on similar interests. While getting recommendations from somebody with similar tastes is a good start, you might also want something else: that the person making the recommendation is an *expert* on these topics. You might prefer getting a recommendation based on the opinions of one expert, rather than 10 other people, even if the 10 others actually have interests somewhat closer to yours.

This raises multiple challenges, including obtaining expertise information, qualifying the information, and combining it with information about similarity of preferences to compute a recommendation. Various techniques for getting information about expertise may be explored. For example, in an online conversation, metrics such as the amount of messages a person contributes and the number of responses these messages receive could be used to assess a participant's expertise. In academic contexts, citation indexing methods could be used. A further consideration is that expertise is topic specific – for example, in the music domain, one person might be an expert on baroque music, another on big band swing, and a third on punk rock. Several techniques may help categorize a person's expertise. If a system has categorical information about items (e.g., their genre), then if a person rates items from one category more often (and more highly) than other genres, this may indicate expertise in this category. And in a conversational application, the messages an individual produces may be analyzed to extract topics that he or she discusses frequently. (We realize that discussing a topic a lot doesn't necessarily make one an expert, but this is a good place to begin experimenting.) The final issue is how to combine expertise and taste information. Existing collaborative filtering algorithms could simply be tweaked with another term that weights neighbors' influence on a recommendation by their expertise. However, how much weight to assign is not clear a priori; experiments are necessary, and perhaps different combinations will be appropriate in different circumstances.

Combining multiple sources of preferences

Thinking about combining expertise and preference information leads to another realization – preferences can be obtained from different sources, and algorithms should

be able to combine different types of preferences appropriately. For example, the preferences in PHOAKS were obtained by mining and aggregating opinions from Usenet messages. However, PHOAKS users were able to rate the recommended URLs, and the system captured usage history (i.e., how often each URL was browsed). Thus, in the end, we had three sources of preferences about URLs: mentions in newsgroup messages, usage history, and explicit ratings. How to combine the three types of preferences is a challenge: as above, it is not clear how much weight to assign to a given type. The analysis of usage history, in particular, requires some thought. For example, simply counting the number of times users clicked on each URL as a preference measure is an obvious strategy. However, one should expect users to click more often on URLs that were higher in the display list. Building on this intuition, deviation from the expected pattern – that is, URLs that were clicked on significantly more or less frequently than expected – probably needs to considered.

This example also illustrates an interesting general point: *using* a recommendation to make a decision also may yield additional preference data that can be used to *evolve* the recommendation. And a single user may play multiple roles – recommendation seeker and preference producer – simultaneously.

Conclusion

Recommender systems have developed in response to a manifest need: helping people deal with the world of information abundance and overload. Further, it has become clear that they can link people with other people who share their interests, not just with relevant information. We identified a set of four major issues for recommender systems: (1) how preference data is obtained and used, (2) the roles played by people and by computation, and the types of communication involved, (3) algorithms for linking people and computing recommendations, and (4) presentation of recommendations to users. We then identified four major approaches to recommender systems, which can be distinguished in large part by which of the issues they address, and how they address them. Finally, we closed by suggesting several challenges that raise important opportunities for new research and application.

Acknowledgements

We thank Brian Amento for his participation in the PHOAKS and TopicShop projects. We thank Paul Resnick, Bonnie Nardi, Steve Whittaker, and anonymous reviewers for valuable comments on earlier drafts of this chapter.

References

1. Ackerman, M.S. Augmenting the Organizational Memory: A Field Study of Answer Garden, in *Proceedings of CSCW'94* (Chapel Hill NC, October 1994), ACM Press, 243-252.
2. Ackerman, M.S. and McDonald, D.W. Answer Garden 2: Merging Organizational Memory with Collaborative Help, in *Proceedings of CSCW'96* (Boston MA, November 1996), ACM Press, 97-105.

3. Aggarwal, C.A., Wolf, J.L., Wu, K-L., and Yu, P.S. Horting Hatches an Egg: A New Graph-Theoretic Approach to Collaborative Filtering, in *Proceedings of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1999.

4. Amento, B., Hill, W., Terveen, L., Hix, D., and Ju, P. An Empirical Evaluation of User Interfaces for Topic Management of Web Sites, in *Proceedings of CHI'99* (Pittsburgh, PA, May 1990), ACM Press, 552-559.

5. Avery, C., Resnick, P., and Zeckhauser, R. The Market for Evaluations. *American Economic Review* 89(3), (1999), 564-584.

6. Balabanovic, M. and Shoham, Y. Fab: Content-Based, Collaborative Recommendation, in Resnick and Varian (eds,), 66-72

7. Barrett, C. Anatomy of a Weblog, Camworld, January 26, 1999. http://www.camworld.com/journal/rants/99/01/26.html.

8. Billsus, D. & Pazzani, M. Learning Collaborative Information Filters, in *Proceedings of the International Conference on Machine Learning* (Madison WI, July 1998), Morgan Kaufmann Publishers.

9. Bush, V. As We May Think. *The Atlantic Monthly*, July 1945.

10. Card, S.K., Robertson, G.C., and York, W. The WebBook and the Web Forager: An Information Workspace for the World-Wide Web, in *Proceedings of CHI'96* (Vancouver BC, April 1996), ACM Press, 111-117.

11. Chalmers, M., Rodden, K., and Brodbeck, D. The Order of Things: Activity-Centred Information Access, in Proceedings of 7th International Conference on the World Wide Web, 1998. (Brisbane Australia, April 1998), 359-367.

12. Deerwester, D., Dumais, S.T., Landauer, T.K., Furnas, G.W., and Harshman, R.A. Indexing by Latent Semantic Analysis. *Journal of the Society for Information Science, 41, 6* (1990), 391-407.

13. Dourish, P. and Bly, S. Portholes: Supporting Awareness in a Distributed Work Group, in *Proceedings of CHI'92* (Monterey CA, May 1992), ACM Press, 541-547.

14. Dourish, P. Where the Footprints Lead: Tracking Down Other Roles for Social Navigation, in Munro, Höök, and Benyon (Eds.), 15-34.

15. Goldberg, D., Nichols, D., Oki, B.M. and Terry, D. Using Collaborative Filtering to Weave an Information Tapestry. *Communications of the ACM*, 35, 12 (December 1992), 51-60.

16. Good, N., Schafer, J.B., Konstan, J., Borchers, A., Sarwar, B., Herlocker, J., and Riedl, J., Combining Collaborative Filtering with Personal Agents for Better Recommendations, in *Proceedings of AAAI'99* (July 1999).

17. Herlocker, J. Explanations in Recommender Systems, position paper in *CHI'99 Workshop Interacting with Recommender Systems*, available from http://www.darmstadt.gmd.de/rec99/schedule.html.

18. Hill, W.C., Hollan, J.D., Wroblewski, D., and McCandless, T., Edit Wear and Read Wear, in *Proceedings of CHI' 92*(Monterey CA, May 1992), ACM Press, 3-9.

19. Hill, W.C., Hollan, J.D. History-Enriched Digital Objects: Prototypes and Policy Issues. *The Information Society, 10, 2* (1994), 139-145.

20. Hill, W.C., Stead, L., Rosenstein, M. and Furnas, G. Recommending and Evaluating Choices in a Virtual Community of Use, in *Proceedings of CHI'95* (Denver CO, May 1995), ACM Press, 194-201.

21. Hill, W. C. and Terveen, L. G. Using Frequency-of-Mention in Public Conversations for Social Filtering. in *Proceedings of CSCW'96* (Boston MA, November 1996), ACM Press, 106-112.

22. Katz, J. Here Come The Weblogs, Slashdot May 24, 1999. http://slashdot.org/features/99/05/13/1832251.shtml.

23. Kautz, H., Selman, B., and Shah, M. The Hidden Web, *AI Magazine*, 18, 2 (Summer 1997), 27 - 36.

24. Kleinberg, J.M. Authoritative Sources in a Hyperlinked Environment, in *Proceedings of 1998 ACM-SIAM Symposium on Discrete Algorithms* (San Francisco CA, January 1998), ACM Press.

25. Konstan, J.A., Miller, B.N., Maltz, D., Herlocker, J.L., Gordon, L.R., and Riedl, J. GroupLens: Applying Collaborative Filtering to Usenet News, in Resnick and Varian (Eds,), 77-87.

26. Lieberman, H. Autonomous Interface Agents, in *Proceedings of CHI'97* (Atlanta GA, March 1997), ACM Press, 67-74.

27. Maes, P. Agents That Reduce Work and Information Overload. *Communications of the ACM 37,7, 31-40*, July 1994.

28. Mackinlay, J.D., Rao, R., and Card, S.K. An Organic User Interface for Searching Citation Links, in *Proceedings of CHI'95* (Denver CO, May 1995), ACM Press, 67-73.

29. Maglio, P.P., Farrell, S., and Barrett, R. How to Define "Place" on the Web, in CHI 2000 Workshop *Social Navigation: A Design Approach*?, edited by Höök, K., Munro, A., and Wexelblat, A.

30. Maltz, D. and Ehrlich, K. Pointing the Way: Active Collaborative Filtering, in *Proceedings of CHI'95* (Denver CO, May 1995), ACM Press, 202-209.

31. McDonald, D. and Ackerman, M. Just Talk to Me: A Field Study of Expertise Location, in *Proceedings of CSCW'98* (Seattle WA, November 1998), ACM Press, 315-324.

32. Munro, A.J, Höök, K., and Benyon, D (Eds.) *Social Navigation of Information Space*. Springer, 1999.

33. Pirolli, P., Pitkow, J., and Rao, R. Silk from a Sow's Ear: Extracting Usable Structures from the Web, in *Proceedings of CHI'96* (Vancouver BC, April 1996), ACM Press, 118-125.

34. Pitkow, J., and Pirolli, P. Life, Death, and Lawfulness on the Electronic Frontier, in *Proceedings of CHI'97* (Atlanta GA, March 1997), ACM Press, 383-390.

35. Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., Riedl, J. GroupLens: An Open Architecture for Collaborative Filtering of Netnews. in Proceedings of CSCW'94 (Chapel Hill NC, October 1994), ACM Press, 175-186.

36. Resnick, P., and Varian, H.R., guest editors, *Communications of the ACM*, Special issue on Recommender Systems, 40, 3 (March 1997).

37. Resnick, P. and Varian, H.R., Recommender Systems, in Resnick and Varian (eds.), 56-58.

38. Shardanand, U., and Maes, P. Social Information Filtering: Algorithms for Automating "Word of Mouth". in Proceedings of CHI'95 (Denver CO, May 1995), ACM Press, 210-217.

39. Terveen, L.G., Selfridge, P.G., and Long, M.D. Living Design Memory: Framework, Implementation, Lessons Learned. Human-Computer Interaction, 10, 1 (1995), 1-38.

40. Terveen, L.G., Hill, W.C., and Amento, B. Constructing, Organizing, and Visualizing Collections of Topically Related Web Resources. *ACM Transactions on Computer-Human Interaction 6,1, 67-94*, March 1999.

41. Viegas, F.B. and Donath, J.S. Chat Circles, in *Proceedings of CHI'99* (Pittsburgh, PA, May 1990), ACM Press, 9-16.

42. Wexelblat, A. and Maes, P. Footprints: History-Rich Tools for Information Foraging, in *Proceedings of CHI'99* (Pittsburgh PA, May 1990), ACM Press, 270-277.