University of Crete
School of Sciences and Engineering
Computer Science Department

CRAWLING THE ALGORITHMIC FOUNDATIONS OF RECOMMENDATION TECHNOLOGIES

by

MANOS PAPAGELIS

Master's Thesis

Heraklion, March 2005

University of Crete
School of Sciences and Engineering
Computer Science Department

Crawling the Algorithmic Foundations of Recommendation Technologies

by

Manos Papagelis

A thesis submitted in partial fulfillment of the requirements for the degree of

Master of Science

Author:

____
Manos Papagelis, Computer Science Department

Supervisory Committee:

____
Dimitris Plexousakis, Associate Professor, Supervisor

____
Vassilis Christophides, Associate Professor, Member

____
Grigoris Antoniou, Professor, Member

Approved by:

____
Dimitris Plexousakis, Associate Professor
Chairman of the Graduate Studies Committee

Heraklion, March 2005

Abstract

The World-Wide-Web has emerged during the last decade as one of the most prominent research fields. However, its size, heterogeneity and complexity to a large extent overcome our ability to efficiently manipulate data using traditional techniques. In order to cope with these characteristics several Web applications require intelligent tools that may help to extract the proper information relevant to the user's requests.

In this thesis we report on the algorithmic aspects of recommendation technologies, which refer to algorithms and systems that have been developed to help users find items that may be of their interest from a variety of available items. Collaborative Filtering (CF), the prevalent method for providing recommendations, has been successfully adopted by research and industrial applications. However, its applicability is limited due to the *sparsity* and the *scalability* problems. Sparsity refers to a situation that transactional data are lacking or are insufficient, while scalability refers to the expensive computations required by CF.

For addressing the scalability problem we propose a method of Incremental CF (ICF) that is based on incremental updates of user-to-user similarities. Our ICF algorithm (i) is not based on any approximation method, thus it gives the potential for high-quality recommendations formulation, and (ii) provides recommendations orders of magnitude faster than classic CF and thus, is suitable for online application.

To provide high-quality recommendations even when data are sparse, we propose a method for alleviating sparsity using trust inferences. Trust inferences are transitive associations between users in the context of an underlying social network and are valuable sources of additional information that help dealing with the sparsity and the cold-start problems. Our experimental evaluation indicates that our method of trust inferences significantly improves the quality performance of the classic CF method.

Finally, we provide a roadmap for future research directions that extend recommendation technologies to more complex types of applications and identify various research opportunities for developing them.

Supervisor: Dimitris Plexousakis

Associate Professor

Περίληψη

Ο Παγκόσμιος Ιστός στη διάρκεια της τελευταίας δεκαετίας έχει αναδειχθεί σε ένα από τα σημαντικότερα πεδία έρευνας. Εντούτοις, το μέγεθος, η ετερογένεια και η πολυπλοκότητά του υπερισχύουν σε μεγάλο βαθμό της δυνατότητά μας να χειριστούμε αποτελεσματικά τα δεδομένα χρησιμοποιώντας παραδοσιακές τεχνικές. Προκειμένου να αντιμετωπιστούν αυτά τα χαρακτηριστικά διάφορες εφαρμογές Ιστού απαιτούν την ανάπτυξη και υιοθέτηση ευφυών εργαλείων για την επιλογή κατάλληλων πληροφοριών σχετικών με τα αιτήματα του χρήστη.

Σε αυτήν την εργασία εξετάζουμε τις αλγοριθμικές πτυχές των τεχνολογιών σύστασης, οι όποιες αναφέρονται στους αλγορίθμους και τα συστήματα που έχουν αναπτυχθεί για να βοηθήσουν τους χρήστες να βρουν αντικείμενα που πιθανόν θα τους φανούν ενδιαφέροντα. Η «Συνεργατική Διήθηση» (ΣΔ), η επικρατούσα μέθοδος για τη δημιουργία συστάσεων, έχει υιοθετηθεί επιτυχώς από ερευνητικές και εμπορικές εφαρμογές. Εντούτοις, η δυνατότητα εφαρμογής της περιορίζεται λόγω των προβλημάτων «σποραδικότητας» και «κλιμακοσημότητας». Η σποραδικότητα αναφέρεται σε μια κατάσταση που τα δεδομένα συναλλαγών μεταξύ του χρήστη και του συστήματος στερούνται ή είναι ανεπαρκή, ενώ η κλιμακοσημότητα αναφέρεται στους ακριβούς υπολογισμούς που απαιτούνται από τη ΣΔ.

Για την αντιμετώπιση του προβλήματος κλιμακοσημότητας προτείνουμε μια μέθοδο Αυξητικής Συνεργατικής Διήθησης (ΑΣΔ) που βασίζεται σε αυξητικές αναπροσαρμογές των ομοιοτήτων μεταξύ χρηστών. Ο ΑΣΔ αλγόριθμός μας (α) δεν είναι βασισμένος σε κάποια μέθοδο προσέγγισης, κατά συνέπεια δίνει τη δυνατότητα για υψηλής ποιότητας συστάσεις, και (β) παρέχει συστάσεις γρηγορότερα από τη μέθοδο κλασικής ΣΔ και είναι κατάλληλος για την ηλεκτρονικές εφαρμογές.

Για την αντιμετώπιση του προβλήματος σποραδικότητας προτείνουμε μία μέθοδο βασισμένη σε χρήση λογικών συμπερασμάτων εμπιστοσύνης. Τα λογικά συμπεράσματα εμπιστοσύνης είναι μεταβατικές ενώσεις μεταξύ των χρηστών στα πλαίσια ενός υποκείμενου κοινωνικού δικτύου και λειτουργούν ως πολύτιμες πηγές πρόσθετων πληροφοριών που βοηθούν στην ελάφρυνση του προβλήματος της

σποραδικότητας. Η πειραματική αξιολόγηση που ακολουθούμε αποδεικνύει ότι η μέθοδός μας βελτιώνει σημαντικά την ποιοτική απόδοση της κλασικής μεθόδου ΣΔ.

Τέλος, παρέχουμε έναν οδικό χάρτη για μελλοντικές ερευνητικές κατευθύνσεις που επεκτείνουν τις τεχνολογίες σύστασης σε πιο σύνθετους τύπους εφαρμογών και προσδιορίζουν διάφορες ερευνητικές ευκαιρίες.

Επόπτης: Δημήτρης Πλεξουσάκης
Αναπληρωτής Καθηγητής

To my parents Eleni and Ioannis, and my brother Sakis

for their carry and support in all aspects of my life

Acknowledgements

I am foremost grateful to my supervisor, Professor Dimitris Plexousakis, for always being available, for his mentoring style that leaves space for conducting individual research, for his invaluable advice and guidance throughout this work, for being a motivator of my next steps in academia.

I am also very thankful to the rest members of the supervisory committee, Professor Grigoris Antoniou for his creative comments and helpful questions, and Professor Vassilis Christophides for encouraging the conducting of high quality research, mainly through his own energy, acting and performance that serve as an exemplar, and certainly for being an inspired instructor and a great adviser.

Special thanks are given to Dr. Martin Doerr, Principal Researcher of ICS-FORTH for being an outstanding mentor and educator since my initial research steps and for all the attention-grabbing discussions we had that usually extended through the narrow frameworks of our discipline.

Part of this work has been carried out by Ioannis Rousidis, Themistoklis Kutsuras and Elias Theoharopoulos; therefore I would like to express my gratitude for their extensive work and supporting ideas throughout this research.

I would also like to thank the members of the Information Systems Laboratory of ICS-FORTH for their creative comments, their collaborative spirit and the harmonic and enjoyable coexistence they provided. It was truly beneficial to share my workspace with them.

I am also grateful to the Computer Science Department of University of Crete and the Institute of Computer Science, Foundation of Research and Technology (ICS-FORTH) for their support in terms of a graduate fellowship and above all for providing an academic and research environment that welcomes creativity, fosters the interchange of ideas and academic conversations, and encourages collaboration.

Above all, I would like to thank my friends for sharing their comments and my worries throughout this research. I wish for all of them the best in their life.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

"We haven't the money, so we've got to think"

-Ernest Rutherford

Contents

1.1 Motivation

The World-Wide-Web (Web) has emerged during the last decade as one of the most prominent research fields. The Web has many topics in common with the traditional research areas, while at the same time it differs in several aspects and opens up new challenges. As an example, one could argue that the Web is the rough equivalent of a huge distributed database. However, it differs substantially from traditional databases regarding the way it stores information, its size, or the number of concurrent users.

The Web's size, heterogeneity and complexity to a large extent overcome our ability to efficiently manipulate data using traditional techniques. In order to cope with those characteristics several Web applications require intelligent tools that may help to extract, from this excess of data, the proper information relevant to the user's requests. The development of intelligent tools over the Web requires the innovative use of Artificial Intelligence and advanced Information Technology techniques.

In this work we report on the algorithmic aspects of recommendation technologies. Recommendation technologies refer to *algorithms* and *systems* that have been developed to help users find items that may be of their interest from a pool of numerous available items that exist on the Web or are defined in a specific application area. Internet users often have diverse, conflicting needs. Differences in personal preferences, social and educational backgrounds, and private or professional interests are pervasive. As a result, it seems desirable to have personalized intelligent systems that process, filter, and display available information in a manner that suits each individual using them. The understanding and development of recommendation technologies have been a popular topic of research ever since the ubiquity of the web made it clear that people of hugely varying backgrounds would be able to access and query the same underlying data. The initial human computer interaction challenge has been made even more challenging by the observation that customized services require sophisticated data structures and well thought-out architectures to be able to scale up to thousands of users and beyond.

1.2 Contributions

The primary contributions of this thesis are:

- The formulation of the recommendation problem and the description of the current state-of-the-art technologies that have emerged
- A qualitative analysis of several prediction algorithms that serves as an initial survey on recommendation technologies
- The development and assessment of a methodology for addressing the scalability limitation of collaborative filtering based recommendation algorithms
- The development and assessment of a methodology for addressing the sparsity and cold-start limitations of collaborative filtering based recommendation algorithms
- A roadmap for future research on recommendation technologies

1.3 Published Work

Study of recommendation algorithms has been a long-term agenda item that has already led to some publications and to the development of a recommendation system, which serves both as a research platform and as a free service. Our former engagement with recommendation algorithms research was when we first studied the way in which these algorithms can be employed in order to discover dynamic, virtual, online communities. Part of this work has been published in terms of a short paper in the International Conference on Advanced Information Systems Engineering [Papagelis and Plexousakis, 2003]. Since then, initial achievements and ideas were extended and led to some other interesting work.

We spent the next period developing and evaluating the quality of collaborative filtering recommendation algorithms that are based on item similarities, instead of user similarities. The results of this work have been published in the International Workshop on Cooperative Information Agents [Papagelis and Plexousakis, 2004]. This contribution has also been invited for publication in a Special Issue of the International Journal on Engineering Applications of Artificial Intelligence [Papagelis and Plexousakis, 2005].

Next, we focused attention to the scalability problem of recommendation systems. We argued for a methodology of incremental computation of user similarities that could improve the performance of recommendation algorithms without reducing their quality. The methodology led to a promising algorithm named Incremental Collaborative Filtering. Preliminary results of this work have been published in the Hellenic Data Management Symposium [Papagelis et al., 2004], while most recent results have been accepted for publication in the proceedings of the International Symposium on Methodologies for Intelligent Systems [Papagelis, Rousidis, Plexousakis, and Theoharopoulos, 2005].

In a slightly different context, we were initially investigating trust implications in web based social networks. Trust management has been an emerging discipline of research. We were particularly interested in developing a computational model for trust and investigating the way in which trust fits architectonically into large scale information discovery systems that function within highly-distributed environments. However, in order for the proposed computational method to acquire both a scientific ground and practical credibility, it would need to be applied on specific application area. Thereafter, we argued that trust propagation techniques can be efficiently employed so as to alleviate the sparsity problem of recommendation algorithms. Results of this work have been accepted for publication in the proceedings of the International Conference on Trust Management [Papagelis, Plexousakis, and Kutsuras, 2005].

1.4 Organization

In this thesis, we study the algorithmic foundations of recommendation technologies and provide methods to address specific shortcomings that harmfully affect their further adoption into commerce and research applications. The remainder of the thesis is organized as follows:

Chapter 2 introduces recommendation algorithms, formulates the recommendation problem and provides a classification of existing approaches to develop them. Then, it presents specific shortcomings that need to be addressed like sparsity and scalability and describes existing methodologies to cope with them.

Chapter 3 presents a survey on recommendation algorithms. Several prediction algorithms are described and evaluated, some of which are novel in that they combine user-based and item-based similarity measures derived from either explicit or implicit ratings. Both statistical and decision-support accuracy metrics of the algorithms are compared against different levels of data sparsity and different operational thresholds.

Chapter 4 proposes a novel method for addressing the scalability problem based on incremental updates of user-to-user similarities. This method, named Incremental Collaborative Filtering algorithm is not based on any approximation method and gives the potential for high-quality recommendation formulation. Furthermore, it provides recommendations orders of magnitude faster than classic Collaborative Filtering and thus, is suitable for online applications.

Chapter 5 proposes a method for alleviating the sparsity problem of collaborative filtering based on trust inferences. Trust inferences are transitive associations between users in the context of an underlying social network and are valuable sources of additional information that help dealing with the sparsity and the cold-start problems.

Chapter 6 concludes the research contributions of the thesis, discusses ways to extend the capabilities of recommendation algorithms and draws directions for further research work.

Finally, there is a chapter committed to relative bibliography that is referred throughout this thesis.

Chapter 2

Recommendation Algorithms

"It is through science that we prove, but through intuition that we discover"

-Jules Henri Poincare

Contents

2.1 Research Topic Placement

Our research topic concerns the better understanding of recommendation algorithms and the development of possible methodologies to address specific shortcomings that arise and may negatively affect their wider application. In our attempt to formulate these methodologies we had to familiarize ourselves with current advancements and considerations of several interesting research areas such as *information retrieval*, *personalization*, *social networks* and *trust management*. Figure 2.1, places our research topic with respect to these areas and next paragraphs provide a short description for each one of these to facilitate following reading.

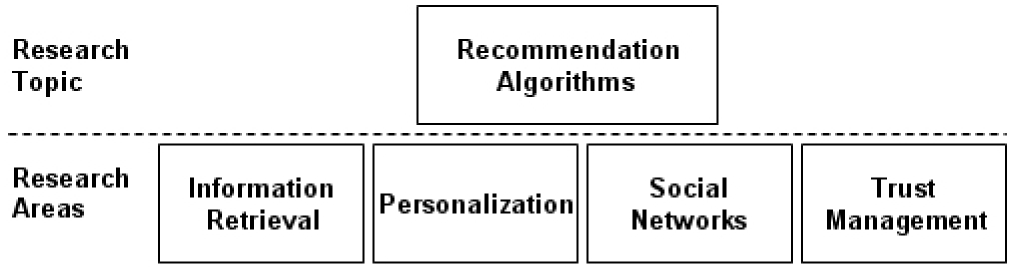

Figure 2-1. Research Topic Placement

2.1.1 Information Retrieval

Information retrieval, the study of information systems for indexing, searching, and recalling data, particularly text or other complex forms, like digital images and video, typically recovers a set of documents or files that match a user's queries. A user often then interacts by moderating and refining a list depending on the information intended by the initial query. Internet search engines and searchable bibliographic databases are recent results of information retrieval research. However, due to the distributed and decentralized nature of the Web a critical challenge that arises is the automatic gathering, filtering and verification of the online information. Intelligent agents may soon collect the desired information automatically, while other relative projects like the Open Directory Project and the Semantic Web that have recently emerged may provide the groundwork for better information retrieval techniques..

2.1.2 Personalization

Modern Internet users are progressively exposed to a wider range of experiences. Therefore, they become ever more proficient in their use of the web and they may well become more demanding in their search for quality services. Personalization is a growing feature of on-line services that is manifested in different ways and contexts, harnessing a series of developing technologies.

Personalization involves a process of gathering user-information during interaction with the user, which is then used to deliver appropriate content and services, tailor-made to the user's needs. The aim is to improve the user's experience of a service. User satisfaction is the most important intention of personalization. It is motivated by the recognition that a user has needs, and meeting them successfully is likely to lead to a satisfying relationship and re-use of the services offered. Beyond the common goal, however, there is great diversity in how personalization can be achieved. Information about the user can be obtained from a history of previous sessions, or through interaction in real time. "Needs" may be those stated by the user as well as those perceived by the underlying system. Once the user's needs are established, rules and techniques, such as *collaborative filtering*, are used to decide what content might be appropriate. This approach would give quality content without explicitly building the one-to-one relationship that requires gathering knowledge on individuals.

2.1.3 Social Networks

A social network is a map of the relationships between individuals, indicating the ways in which they are connected through various social familiarities ranging from casual acquaintance to close familial bonds. The analysis of social networks has emerged as a key technique in modern sociology, anthropology, and organizational studies, as well as a popular topic of speculation and study. Research in a number of academic fields have demonstrated that social networks operate on many levels, from families up to the level of nations, and play a critical role in determining the way problems are solved, organizations are run, and the degree to which individuals succeed in achieving their goals.

Social network theory views social relationships in terms of nodes and ties. Nodes are the individual actors within the networks, and ties are the relationships between the actors. There can be many kinds of ties between the nodes, depending on the relationships being studied. In its most simple form, then, a social network is a map of all of the relevant ties between the nodes being studied. These concepts are often displayed in a social network diagram, such as the one in Figure 2.2, where nodes are the points and ties are the lines.

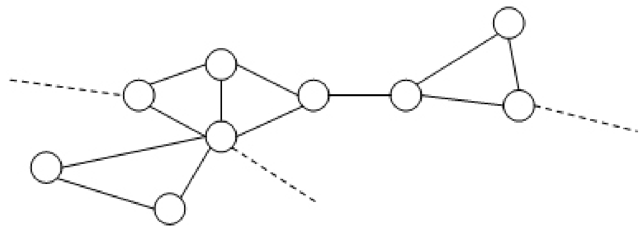

Figure 2-2. Social Networks

The shape of the social network has been found to be a key factor in a network's usefulness to the individuals it includes. Smaller tighter networks, for example, can actually be less useful to their members than networks with lots of loose connections (weak ties) to other individuals outside the main network. More "open" networks, with many weak ties and social connections, are more likely to introduce new ideas and opportunities to their members than closed networks with many redundant ties. In other words, a group of friends who only do things with each other already share the same knowledge and opportunities. A group of individuals where each has connections to other social worlds is likely to have access to a wider range of information. It is better for individual success to have connections to a variety of networks rather than many connections within a single network. Similarly, individuals can exercise influence or act as brokers within their social networks by bridging between two networks that are not directly linked (called filling social holes).

The power of social network theory stems from its difference from traditional sociological studies, which assume that it is the attributes of individuals that matter. Social network theory produces an alternate view, where the attributes of individuals are less important than their relationships with other individuals within the network. This approach has turned out to be useful for explaining many real-world phenomena, but leaves less room for individual activity; the ability for individuals to influence their success, since much of it rests within the structure of their network.

Online social networks have recently been defined to describe the means of networking in virtual communities. They became popular with the advent of websites to examine how people interact with each other, characterizing the many informal connections that link individuals together.

2.1.4 Trust Management

In an environment of information excess and universal connectivity provided through the Web and other sorts of communication, social trust [McKnight and Chervany, 1996] between individuals becomes an interesting and invaluable aspect. Trust is regarded as a concept that provides evidence on whether to believe or disbelieve information asserted by other peers. Therefore, belief should only be assigned to statements from people we consider trustworthy.

However, when supposing huge networks such as the Semantic Web, trust statements based on personal experience become unfeasible. In general, trust is defined in [Mui et al., 2002] as the "subjective expectation an agent has about another's future behavior based on the history of their encounters". Trust in network environments composed of trust statements between known individuals, constitutes the basis for defining trust between individuals that are unknown to each other. Trust plays an important role to network theory, especially to decentralized infrastructures, such as the Semantic Web and serves as the core concept in defining the "Web of Trust" [Golbeck et al., 2003].

The meaning of trust has been made obvious through empirical evidence from social psychology and sociology, indicating that transitivity is an important characteristic of social networks [Holland and Leinhardt, 1972; Rapoport, 1963]. The drive towards transitivity can also be explained in terms of Heider's famous "balance theory" [Heider, 1958], i.e., individuals are more prone to interact with friends of friends than unknown peers. Adopting the simplest policy of trust propagation, if an individual *A* trusts an individual *B* and *B* trusts an individual *C*, then it can likewise be considered that *A* trusts *C*. Trust would thus propagate through the network and be effective whenever two individuals are connected via at least one trust path. However, owing to certain implications and application area, further properties need to be identified to define the trust concept. Social and psychological aspects must be taken into account and specific criteria of computability and scalability need to be satisfied.

2.2 Introduction to Recommendation Systems

Although the roots of recommendation systems can be traced back to the extensive work in approximation [Powell, 1981], in information retrieval [Salton, 1989], and also to the consumer choice modelling in marketing [Lilien et al., 1992], they emerged as an independent research area in the mid-1990's when researchers started focusing on recommendation problems that explicitly rely on the ratings structure, and especially since the appearance of the first papers on collaborative filtering [Resnick et al., 1994; Hill et al., 1995; Shardanand and Maes, 1995]. Since then, there has been much research work conducted both in the industry and academia on developing new approaches to recommendation systems. The interest remains high because it constitutes a problem-rich research area and because of the widely accepted practical applications that have been developed to help users deal with information overload.

2.2.1 Overview of Recommendation Systems

Recommendation systems, originally referred to as collaborative filtering systems, were developed to address two challenges that could not be addressed by existing keyword-based information filtering systems. First, they addressed the problem of overwhelming numbers of *on-topic* documents; ones which would be selected by a keyword filter and would be ranked according to human judgement about their quality. Second, they addressed the problem of filtering non-text documents mainly based on rating structure. For example, the Ringo system [Shardanand and Maes, 1995] applied collaborative filtering to recommend music to individuals.

In recent years, such systems are extensively adopted by both research and e-commerce applications in order to provide an intelligent mechanism to filter out the excess of information available and to provide customers with the prospect to effortlessly find out items that they will probably like according to their logged history of prior transactions. Examples of "e-markets" that take advantage of such systems are Amazon.com, CDNOW, Drugstore.com, eBay.com, Alexa, Myfreddy and Reel.com. There also exist non-profitable examples of recommendation systems such as GroupLens.

2.2.2 Formulation of the Recommendation Problem

In its most common formulation, the recommendation problem is reduced to the problem of estimating ratings for the items that have not been seen by a user. This estimation is based on prior transactions of the user with the system, usually in terms of submitted ratings to items or other profile information. Once we can estimate ratings for the yet unrated items, we can recommend to the user the item(s) with the highest estimated rating(s).

More formally, the recommendation problem can be formulated as follows. Let U be the set of all users and let I be the set of all possible items that can be recommended, such as books, movies, or restaurants. The space I of possible items can be very large, ranging in hundreds of thousands or even millions of items in some applications, such as recommending books or CDs. Similarly, the user space can also be very large – millions in some cases. Let f^+ be a *utility function* that measures usefulness of item i to user u, i.e., f^+: $U \times I \rightarrow R$, where R is a totally ordered set. Then for each user $u \in U$, we want to choose such item $i' \in I$ that maximizes the user's utility. More formally:

$$\forall u \in U \quad and \quad i \in I, \; i'_u = \max(f^+(u,i)) \tag{2.1}$$

In recommendation systems the utility of an item is usually represented by a rating, which indicates how a particular user liked a particular item, e.g., David gave the movie "Gladiator" the rating of 7 (out of 10). However, as indicated earlier, in general utility can be an arbitrary function, including a profit function. Each element of the user space U can be defined with a profile that includes various user characteristics, such as age, gender, income, marital status, etc. In the simplest case, the profile can contain only a single identifier element, such as User ID. Similarly, each element of the item space I is defined with a set of characteristics. For example, in a movie recommendation application, where I is a collection of movies, each movie can be represented not only by its ID, but also by its title, genre, director, year of release, leading actors, etc.

The central problem of recommendation systems lies in that utility function f^+ is usually not defined on the whole $U \times I$ space, but only on a subset of it. In order to define the utility f^+ on the whole space $U \times I$ predictions for the non-existent user-item combinations are required. In recommendation systems, utility is represented by ratings and is initially defined only on the items previously rated by the users. For example, in a movie recommendation application, such as the one at MovieLens.org, users initially rate some subset of movies that they have already seen. An example of a user-item rating matrix for a movie recommendation application is presented in

Table 2-1, where ratings are specified on the scale of 1 to 10. The “-” symbol for some of the ratings in the table means that the users have not rated these movies. Therefore, a recommendation engine needs to be able to predict the ratings of the non-rated movie-user combinations and formulate recommendations based on these predictions.

Predictions of unknown ratings are usually done by specifying *heuristics* that define the utility function and empirically validating its performance. Once the unknown ratings are predicted, actual recommendations of an item to a user are made by selecting the highest rating among all the estimated ratings for that user, according to equation (2.1). Alternatively, we can recommend the top-*N* items.

Table 2-1. Example of a rating matrix for a movie recommendation system

	Movie 1	Movie 2	Movie 3	Movie 4
Bill	8	6	3	6
Sakis	-	5	4	8
Elsa	8	7	2	-
Dim	3	-	9	5

2.3 Classification of the Recommendation Approaches

There are generally two methods to formulate recommendations, depending on the way that user models are constructed [Allen, 1990], on the prediction methods employed, as well as on the type of items to be recommended. The two different approaches are *content-based* [Balabanovic and Sholam, 1997; Kalles et al., 2003] and *collaborative filtering* [Herlocker et al., 2000; Hofmann, 2003], while additional *hybrid* methods have been proposed as well that combine the two most important methods [Balabanovic and Sholam, 1997].

2.3.1 Content Based Recommendation Algorithms

The content-based approach to recommendation has its roots in information retrieval [Salton, 1989; Baeza-Yates and Ribeiro-Neto, 1999] and information filtering [Belkin and Croft, 1992] research, and employs many techniques that have been extensively studied in the past. The improvement over the traditional information retrieval approaches comes from the use of *user profiles* that contain information about users' tastes, preferences, and needs. The profiling information can be elicited from users *explicitly*, e.g., through questionnaires, or *implicitly* – learned from their transactional behavior over time.

Content-based algorithms are principally used when text documents are to be recommended, such as web pages (URLs), publications, jokes or news. The system maintains information about user preferences either by initial input about user's interests during the registration process or by rating documents. Recommendations are then formed by taking into account the content of documents and by filtering in the ones that better match the user's preferences and logged profile. For example, in a movie recommendation application, in order to recommend movies to a user, the content-based recommendation system tries to understand the commonalities among the movies that user has rated highly in the past (specific actors, directors, genres, subject matter, etc.). Then, only the movies that have a high degree of similarity to whatever user's preferences are would be recommended.

Besides the traditional heuristics that are based mostly on information retrieval methods, other techniques for content-based recommendation have also been used, such as Bayesian classifiers [Pazzani and Billsus, 1997; Mooney et al., 1998] and various machine learning techniques, including clustering, decision trees, and artificial neural networks [Pazzani and Billsus, 1997]. These techniques differ from information retrieval-based approaches in that they calculate utility predictions based not on a heuristic formula, such as a cosine similarity measure, but rather are based on a model learned from the underlying data using statistical learning and machine learning techniques. For example, based on a set of Web pages that were rated as "relevant" or "irrelevant" by the user, [Pazzani and Billsus, 1997] use the naïve Bayesian classifier [Duda et al., 2001] to classify unrated Web pages.

2.3.2 Collaborative Filtering Based Recommendation Algorithms

Collaborative filtering is the method of making automatic predictions (filtering) about the interests of a user by collecting taste information from many users (collaborating). For example, a collaborative filtering or recommendation system for music tastes could make predictions about which music a user should like given a partial list of that user's tastes (likes or dislikes). Note that these predictions are specific to the user, but use information gleaned from many users. This differs from the more simple approach of giving an average (non-specific) score for each item of interest, for example based on its number of votes.

In the age of information explosion such techniques can prove very useful as the number of items in only one category (such as music, movies, books, news, web pages) have become so large that a single person cannot possibly view them all in order to select relevant ones. Relying on a scoring or rating system which is averaged across all users ignores specific demands of a user, and is particularly poor in tasks where there is large variation in interest, for example in the recommendation of music. Obviously, other methods to combat information explosion exist such as web search, clustering, and more.

Therefore, collaborative filtering algorithms aim to identify users that have relevant interests and preferences by calculating similarities and dissimilarities between user profiles [Herlocker et al., 2004]. The idea behind this method is that, it may be of benefit to one's search for information to consult the behavior of other users who share the same or relevant interests and whose opinion can be trusted.

There have been many collaborative systems developed in the academia and the industry. The Tapestry system relied on each user to identify like-minded users manually [Goldberg et al., 1992]. GroupLens [Resnick et al., 1994; Konstan et al., 1997], Video Recommender [Hill et al., 1995], and Ringo [Shardanand and Maes, 1995] were the first systems to use collaborative filtering algorithms to automate prediction. Other examples of collaborative recommendation systems include the book recommendation system from Amazon.com, MovieCritic that recommends movies on the Web, the PHOAKS system that helps people find relevant information on the WWW [Terveen et al., 1997], and the Jester system that recommends jokes [Goldberg et al., 2001].

According to [Breese et al., 1998], algorithms for collaborative recommendations can be grouped into two general classes: *memory-based* and *model-based*.

2.3.2.1 Memory-based Collaborative Filtering Algorithms

Memory-based algorithms [Resnick et al., 1994; Shardanand and Maes, 1995; Breese et al., 1998; Nakamura and Abe, 1998; Delgado and Ishii, 1999] essentially are heuristics that make rating predictions based on the entire collection of previously rated items by the users. That is, the value of the unknown rating $r_{u,i}$ for user u and item i is usually computed as an aggregate of the ratings of some other (usually the N most similar) users for the same item i:

$$r_{u,i} = \underset{u' \in \hat{U}}{aggr}(r_{u',i}) \tag{2.2}$$

where $\hat{U}$ denotes the set of N users u' that are the most similar to user u and who have rated item i (N can range anywhere from 1 to the number of all users). In the simplest case, the aggregation can be a simple average, as defined by equation (2.3).

$$r_{u,i} = \frac{1}{N} \sum_{u' \in \hat{U}} r_{u',i} \tag{2.3}$$

$$r_{u,i} = k \sum_{u' \in \hat{U}} sim(u,u') \times r_{u',i} \tag{2.4}$$

$$r_{u,i} = \overline{r}_c + k \sum_{u' \in \hat{U}} sim(u,u') \times (r_{u',i} - \overline{r}_{u'}) \tag{2.5}$$

However, the most common aggregation approach is to use the weighted sum, shown in equation (2.4). The similarity measure between the users u and u', $sim(u, u')$, is essentially a distance measure and is used as a weight, i.e., the more similar users u and u' are, the more weight rating $r_{u',i}$ will carry in the prediction of $r_{u,i}$. Multiplier k serves as a normalizing factor and is usually selected as:

$$k = \frac{1}{\sum_{u' \in \hat{U}} |sim(u,u')|} \tag{2.6}$$

Note that *sim(x,y)* is a heuristic artifact that is introduced in order to be able to differentiate between levels of user similarity (i.e., to be able to find a set of "closest peers" or "nearest neighbors" for each user) and at the same time simplify the rating estimation procedure. As shown in equation (2.4), different recommendation applications can use their own user similarity measure, as long as the calculations are normalized using the normalizing factor k, as shown above. The two most commonly used similarity measures will be described below.

One problem with using the weighted sum, as in equation (2.4), is that it does not take into account the fact that different users may use the rating scale differently. The adjusted weighted sum, shown in equation (2.5), has been widely used to address this limitation. In this approach, instead of using the absolute values of ratings, the weighted sum uses their deviations from the average rating of the corresponding user. In equation (2.5), average rating of user u, $\overline{r}_u$ is defined as:

$$\overline{r}_u = \frac{1}{|I_u|}\sum_{i \in I_u} r_{u,i} \quad where \quad I_u = \{i \in I \mid r_{u,i} \neq \varnothing\} \tag{2.7}$$

Various approaches have been used to compute the similarity *sim(u,u')* between users in collaborative recommendation systems. In most of these approaches, the similarity between two users is based on their ratings of items that both of them have rated. The two most popular approaches are *correlation-based* and *cosine-based*. To present them, let I_{xy} be the set of all items co-rated by both users x and y, i.e. $I_{xy} = \{i \in I \mid r_{x,i} \neq \varnothing \,\&\, r_{y,i} \neq \varnothing\}$. In collaborative recommendation systems I_{xy} is used mainly as an intermediate result for calculating the "nearest neighbors" of user x and is often computed in a straightforward manner, i.e., by computing the intersection of sets I_x and I_y. However, some methods, such as the graph-theoretic approach to collaborative filtering [Aggarwal et al., 1999], can determine the nearest neighbors of x without computing I_{xy} for the whole user base. In the correlation-based approach, the Pearson correlation coefficient is used to measure the similarity between users [Resnick et al., 1994; Shardanand and Maes, 1995]:

$$sim(x,y) = \frac{\sum_{i \in I_{xy}} (r_{x,i} - \overline{r}_x)(r_{y,i} - \overline{r}_y)}{\sqrt{\sum_{i \in I_{xy}} (r_{x,i} - \overline{r}_x)^2 \sum_{i \in I_{xy}} (r_{y,i} - \overline{r}_y)^2}} \tag{2.8}$$

In the cosine-based approach [Breese et al., 1998; Sarwar et al., 2001], the two users x and y are treated as two vectors in m-dimensional space, where $m = |I_{xy}|$. Then, the similarity between two vectors can be measured by computing the cosine of the angle between them:

$$sim(x,y) = \cos(\vec{x},\vec{y}) = \frac{\vec{x}\cdot\vec{y}}{\|\vec{x}\|_2 \times \|\vec{y}\|_2} = \frac{\sum_{i \in I_{xy}} r_{x,i} r_{y,i}}{\sqrt{\sum_{i \in I_{xy}} r_{x,i}^2}\sqrt{\sum_{i \in I_{xy}} r_{y,i}^2}} \tag{2.9}$$

where $\vec{x}\cdot\vec{y}$ denotes the dot-product between the vectors $\vec{x}$ and $\vec{y}$. Still another approach to measuring similarity between users uses the *mean squared difference* measure and is described in [Shardanand and Maes, 1995]. Note that different recommendation systems may take different approaches in order to implement the user similarity calculations and rating estimations as efficiently as possible. One common strategy is to calculate all user similarities *sim(x,y)* (including the calculation of I_{xy}) in advance and recalculate them only once in a while (since the network of peers usually does not change dramatically in a short time). Then, the ratings can be efficiently calculated on demand (using precomputed similarities), e.g., whenever the user asks for a recommendation.

Many performance-improving modifications, such as *default voting*, *inverse user frequency*, *case amplification* [Breese et al., 1998], and *weighted-majority prediction* [Nakamura and Abe 1998; Delgado and Ishii, 1999], have been proposed as extensions to these standard correlation-based and cosine-based techniques. For example, the default voting [Breese et al., 1998] is an extension to the memory-based approaches described above. It was observed that whenever there are relatively few user-specified ratings, these methods would not work well in computing similarity between users x and y since the similarity measure is based on the intersection of the item sets, i.e., sets of items rated by both users x and y. It was empirically shown that the rating prediction accuracy could improve if we assume some default rating value for the missing ratings [Breese et al., 1998].

Also, while the above techniques traditionally have been used to compute similarities between users, in [Sarwar et al., 2001] it was proposed to use the same correlation-based and cosine-based techniques to compute similarities between items instead and obtaining the ratings from them. Furthermore, [Sarwar et al., 2001] presents empirical evidence that item-based algorithms can provide better computational performance than traditional user-based collaborative methods, while at the same time also providing better quality than the best available user-based algorithms.

2.3.2.2 Model-based Collaborative Filtering Algorithms

In contrast to memory-based methods, model-based algorithms [Breese et al., 1998; Billsus and Pazzani, 1998; Ungar and Foster, 1998; Chien and George, 1999; Getoor and Sahami, 1999; Goldberg et al., 2001] use the collection of ratings to learn a model, which is then used to make rating predictions. For example, [Breese et al., 1998] proposes a probabilistic approach to collaborative filtering, where the unknown ratings are calculated as:

$$r_{u,i} = E(r_{u,i}) = \sum_{c=0}^{n} c \times \Pr(r_{u,i} = c \mid r_{u,i'}, i' \in I_u) \tag{2.10}$$

and it is assumed that rating values are integers between 0 and n, and the probability expression is the probability that user u will give a particular rating to item i given user's ratings of the previously rated items. To estimate this probability, [Breese et al., 1998] proposes two alternative probabilistic models: *cluster models* and *Bayesian networks*. In the first model, likeminded users are clustered into classes. Given the user's class membership, the user ratings are assumed to be independent, i.e., the model structure is that of a naïve Bayesian model. The number of classes and the parameters of the model are learned from the data. The second model represents each item in the domain as a node in a Bayesian network, where the states of each node correspond to the possible rating values for each item. Both the structure of the network and the conditional probabilities are learned from the data. One limitation of this approach is that each user can be clustered into a single cluster, whereas some recommendation applications may benefit from the ability to cluster users into several categories at once. For example, in a book recommendation application, a user may be interested in one topic (e.g., programming) for work purposes and a completely different topic (e.g., fishing) for leisure.

Moreover, [Billsus and Pazzani, 1998] proposed a collaborative filtering method in a machine learning framework, where various machine learning techniques (such as artificial neural networks) coupled with feature extraction techniques (such as singular value decomposition – an algebraic technique for reducing dimensionality of matrices) can be used. Both [Breese et al., 1998] and [Billsus and Pazzani, 1998] compare their respective model-based approaches with standard memory-based approaches and report that in some applications model based methods outperform memory-based approaches in terms of accuracy of recommendations. However, the comparison in both cases is purely empirical and no underlying theoretical evidence supporting this claim is provided.

There has been several other model-based collaborative recommendation approaches proposed in the literature. A statistical model for collaborative filtering was proposed in [Ungar and Foster, 1998], and several different algorithms for estimating the model parameters were compared, including *K*-means clustering and Gibbs sampling. Other methods for collaborative filtering include a Bayesian model [Chien and George, 1999], a probabilistic relational model [Getoor and Sahami 1999], and a linear regression [Sarwar et al., 2001]. Furthermore, [Kumar et al., 2001] use a simple probabilistic model to demonstrate that collaborative filtering is valuable with relatively little data on each user, and that, in certain restricted settings, simple collaborative filtering algorithms are almost as effective as the best possible algorithms in terms of utility.

As in the case of content-based techniques, the main difference between collaborative model-based techniques and memory-based approaches is that the model-based techniques calculate utility (rating) predictions based not on some ad-hoc heuristic rules, but rather based on a model learned from the underlying data using statistical learning and machine learning techniques. A method combining both memory-based and model-based approaches was proposed in [Pennock and Horvitz, 1999], where it was empirically demonstrated that the use of this combined approach can provide better recommendations than pure memory-based and model-based collaborative approaches.

A different approach to improving the performance of existing collaborative filtering algorithms was taken by [Yu et al., 2002], where the input set of user-specified ratings is carefully selected using several techniques that exclude noise, redundancy, and exploit the sparsity of the ratings' data. The empirical results demonstrate the increase in accuracy and efficiency for model-based collaborative filtering algorithms. It is also suggested that the proposed input selection techniques may help the model-based algorithms to address the problem of learning from large databases [Yu et al., 2002].

2.3.3 Hybrid Methods

Several recommendation systems use a hybrid approach by combining collaborative and content-based methods, which helps to avoid certain limitations of content-based and collaborative systems [Balabanovic and Shoham, 1997; Basu et al., 1998; Ungar and Foster, 1998; Claypool et al., 1999; Soboroff and Nicholas, 1999; Pazzani, 1999;

Schein et al., 2002]. Different ways to combine collaborative and content-based methods into a hybrid recommendation system are described in this section.

Many hybrid recommendation systems, including Fab [Balabanovic and Shoham, 1997] and the "collaboration via content" approach, described in [Pazzani, 1999], combine collaborative and content-based approaches by (1) learning and maintaining user profiles based on content analysis using various information retrieval methods and other content-based techniques, and (2) directly comparing the resulting profiles to determine similar users in order to make collaborative recommendations. This means that users can be recommended items when items either score highly against the user's profile or are rated highly by a user with a similar profile. Moreover, [Balabanovic and Shoham, 1997] observes that content-based and collaborative approaches can be considered as special cases of the hybrid approach. If the content analysis component does not extract any features (e.g., keywords) from items and just deals with a unique item identifier, then the hybrid approach reduces to pure collaborative recommendation. Moreover, if there is only a single user, the hybrid method reduces to pure content-based recommendation. [Basu et al., 1998; Melville et al., 2002] follow similar approach and proposes to use additional sources of information (e.g., the age or gender of users or the genre of movies) to aid collaborative filtering predictions, i.e., [Basu et al., 1998; Melville et al., 2002] adds some content-based elements into the collaborative filtering framework. Similarly, [Soboroff and Nicholas, 1999; Schein et al., 2002] propose to implement the hybrid approach by incorporating some elements of collaborative filtering into the content-based recommendation framework using latent semantic indexing technique.

Another approach to building hybrid recommendation systems is to implement separate collaborative and content-based recommendation systems. Then, we can have two different scenarios. First, we can combine the outputs (ratings) obtained from individual recommendation systems into one final recommendation using either a linear combination of ratings [Claypool et al., 1999] or a voting scheme [Pazzani, 1999]. Alternatively, we can use one of the individual recommendation systems, at any given moment choosing to use the one that is "better" than others based on some recommendation "quality" metric. For example, the DailyLearner system [Billsus and Pazzani, 2000] selects the recommendation system that can give the recommendation with the higher level of confidence, while [Tran and Cohen, 2000] chooses the one whose recommendation is more consistent with past ratings of the user. Yet another hybrid approach to recommendations is used by [Condliff et al., 1999; Ansari et al., 2000], where instead of combining collaborative and content-based methods the authors propose to use all the available information in a single recommendation model. Both [Condliff et al., 1999] and [Ansari et al., 2000] use Bayesian mixed-effects regression models that employ Markov chain Monte Carlo methods for parameter estimation and prediction. In particular, [Ansari et al., 2000] use the profile information of users and items in a single statistical model that estimates unknown ratings.

Hybrid recommendation systems can also be augmented by knowledge-based techniques [Burke, 2000], such as case-based reasoning, in order to improve recommendation accuracy and to address some of the limitations (e.g., new user, new item problems) of traditional recommendation systems. For example, knowledge-based recommendation system Entrée [Burke, 2000] uses some domain knowledge about restaurants, cuisines, and foods (e.g., that "seafood" is not "vegetarian") to recommend restaurants to its users. However, the main drawback of knowledge-based systems is a need for knowledge acquisition – a well-known bottleneck for many artificial intelligence applications. Moreover, it was empirically demonstrated in [Balabanovic and Shoham, 1997; Pazzani, 1999] that hybrid methods can provide more accurate recommendations than pure collaborative and content-based methods.

2.3.4 Summary

To summarize, there has been much research done on recommendation technologies over the past several years that have used a broad range of statistical, machine learning, information retrieval and other techniques and that significantly advanced the state-of-art in comparison to early recommendation systems that utilized collaborative- and content-based heuristics. As was discussed in this section, recommendation systems can be categorized as being (a) content-based, collaborative, or hybrid, based on the recommendation approach used, and (b) memory-based or model-based based on the types of recommendation techniques used for the rating estimation. We use these two orthogonal dimensions to classify the recommendation systems research in Table 2-2.

Table 2-2. Classification of Recommendation Approaches

<table>
<tr><th rowspan="2">Recomm. Approach</th><th colspan="2">Recommendation Technique</th></tr>
<tr><th>Memory-based</th><th>Model-based</th></tr>
<tr><td>Content-based</td><td>Commonly used techniques:
• TF-IDF (information retrieval)
• Clustering
Representative research:
• Lang, 1995
• Balabanovic and Shoham, 1997
• Pazzani and Billsus, 1997</td><td>Commonly used techniques:
• Bayesian classifiers
• Clustering
• Decision trees
• Artificial neural networks
Representative research:
• Pazzani and Billsus, 1997
• Mooney et al., 1998
• Mooney, 1999
• Billsus and Pazzani, 1999, 2000</td></tr>
<tr><td>Collaborative</td><td>Commonly used techniques:
• Nearest neighbor (cosine, correlation)
• Clustering
• Graph theory
Representative research:
• Resnick et al. 1994
• Hill et al. 1995
• Shardanand and Maes 1995
• Breese et al. 1998
• Nakamura and Abe 1998
• Aggarwal et al. 1999
• Delgado and Ishii 1999
• Pennock and Horwitz 1999
• Sarwar et al. 2001</td><td>Commonly used techniques:
• Bayesian networks
• Clustering
• Artificial neural networks
• Linear regression
Representative research:
• Billsus and Pazzani 1998
• Breese et al. 1998
• Ungar and Foster 1998
• Chien and George 1999
• Getoor and Sahami 1999
• Pennock and Horwitz 1999
• Goldberg et al. 2001
• Kumar et al. 2001</td></tr>
<tr><td>Hybrid</td><td>Combining content-based and collaborative components by:
• Linear combination
• Various voting schemes
Representative research:
• Balabanovic and Shoham 1997
• Claypool et al. 1999
• Pazzani 1999
• Billsus and Pazzani 2000
• Tran and Cohen 2000</td><td>Combining content-based and collaborative components by:
• Incorporating one component as a part of the other
• Building one unifying model
Representative research:
• Basu et al. 1998
• Condliff et al. 1999
• Soboroff and Nicholas 1999
• Ansari et al. 2000
• Melville et al. 2002
• Schein et al. 2002</td></tr>
</table>

2.4 Shortcomings of Recommendation Approaches

Content-based and collaborative filtering based methods have been successfully adopted by research and industrial applications in order to filter out and personalize information according to user interests. However, these methods encompass certain shortcoming that may have a negative impact on the effectiveness of the recommendation algorithms. In this paragraph we describe these shortcomings and discuss recent research that has been conducted to get over them.

2.4.1 Challenges and Limitations of Content-based Methods

As was observed in [Shardanand and Maes, 1995; Balabanovic and Shoham 1997], content-based recommendation systems have several limitations such as *limited content analysis*, *over-specialization* and the *new user problem*.

2.4.1.1 Limited content analysis.

Content-based techniques are limited by the features that are explicitly associated with the objects that these systems recommend. Therefore, in order to have a sufficient set of features, the content must either be in a form that can be parsed automatically by a computer (e.g., text), or the features should be assigned to items manually. While information retrieval techniques work well in extracting features from text documents, some other domains have an inherent problem with automatic feature extraction. For example, automatic feature extraction methods are much harder to apply to the multimedia data, e.g., graphical images, audio and video streams. Moreover, it is often not practical to assign attributes by hand due to limitations of resources [Shardanand and Maes, 1995].

Another problem with limited content analysis is that if two different items are represented by the same set of features, they are indistinguishable. Therefore, since text-based documents are usually represented by their most important keywords, content-based systems cannot distinguish between a well-written article and a badly written one, if they happen to use the same terms [Shardanand and Maes, 1995].

2.4.1.2 Over-specialization.

When the system can only recommend items that score highly against a user's profile, the user is limited to being recommended items similar to those already rated. For example, a person with no experience with Greek cuisine would never receive a recommendation for even the greatest Greek restaurant in town. This problem, which has also been studied in other domains, is often addressed by introducing some randomness. For example, the use of genetic algorithms has been proposed as a possible solution in the context of information filtering [Sheth and Maes, 1993]. In addition, the problem with over-specialization is not only that the content-based systems cannot recommend items that are different from anything the user has seen before. In certain cases, items should not be recommended if they are too similar to something the user has already seen, such as different news article describing the same event.

Therefore, some content-based recommendation systems, such as DailyLearner [Billsus and Pazzani, 2000], filter out items not only if they are too different from user's preferences, but also if they are too similar to something the user has seen before.

2.4.1.3 New user problem.

The user has to rate a sufficient number of items before a content-based recommendation system can really understand user's preferences and present the user with reliable recommendations. Therefore, a new user, having very few ratings, would not be able to get accurate recommendations. To address some of these issues, collaborative filtering approach to recommendation systems [Resnick et al., 1994; Hill et al., 1995; Shardanand and Maes, 1995] has been proposed.

2.4.2 Challenges and Limitations of Collaborative Filtering

Collaborative recommendation algorithms do not have certain shortcomings that content-based systems encompass. Since collaborative systems use other users' suggestions (ratings), they can deal with any kind of content and recommend any item, even the ones that are dissimilar to those seen by a user in the past. However, collaborative systems have their own limitations [Balabanovic and Shoham, 1997;

Lee, 2001] that expand to three key dimensions, commonly identified as *the scalability problem*, *the sparsity problem* and *the cold-start problem*.

2.4.2.1 The Scalability Problem

Collaborative Filtering seems to be efficient in filtering in items that are interesting to users. However, it requires computations that are very expensive and grow non-linearly with the number of users and items in a database. Therefore, in order to bring recommendation algorithms successfully on the web, and succeed in providing recommendations with acceptable delay, sophisticated data structures and advanced, scalable architectures are required. In [Cosley et al., 2002], authors describe an open framework for practical testing of recommendation systems in an attempt to provide a standard, public test bed to evaluate recommendation algorithms in real-world conditions.

The collaborative method generates recommendations based on a subset of users that are most similar to the active user. The formulation of a single recommendation is a two-step computation. First, the algorithm needs to compute the similarity between the active user and all other users, based on their co-rated items, so as to pick the ones with similar behavior. Subsequently, the algorithm recommends to the active user items that are highly rated by his or her most similar users. In order to compute the similarities between users, a variety of similarity measures have been proposed, such as Pearson correlation, cosine vector similarity, Spearman correlation, entropy-based uncertainty measure and mean-square difference.

Collaborative filtering fails to scale up its computation with the growth of both the number of users and items in the database. To deal with the scalability problem [Breese et al, 1998] and [Ungar and Foster, 1998] utilize Bayesian network and clustering approaches, while [Sarwar et al, 2001] apply folding in Singular Value Decomposition (SVD) to reduce the dimensionality of the user-item matrix. It is also possible to address these scaling issues by data reduction or data focusing techniques. [Yu et al., 2002] and [Zeng et al., 2003] adopt instance selection for removing the irrelevant and redundant instances. Moreover, content-boosted Collaborative Filtering approaches reduce the number of items examined, by partitioning the item space according to item category or subject classification. Finally, more greedy approaches concentrate on randomly sampling users, discarding users with few ratings or discarding very popular or unpopular items.

Unfortunately, even when these methods achieve improved performance, they also reduce recommendation quality in several ways. Bayesian networks may prove practical for environments in which user preferences change slowly with respect to the time needed to build the model, but are not suitable for environments in which user preference models must be updated frequently. Clustering-based methods suffer from poor accuracy. It is possible to improve their quality by using numerous fine-grained segments [Jung and Kim, 2001], but then online user segment classification becomes almost as expensive as finding similar users using the classic Collaborative Filtering. SVD-based work focuses mainly on accuracy rather than efficiency. Data focusing and reduction approaches, such as instance selection or item-space partitioning, experience reduced accuracy due to loss of information. If an algorithm discards the most popular or unpopular items, there may be items that will never be recommended to some users. Obviously, to gain in computation one needs to lose in recommendation quality and vice versa. Appropriate trade-offs must be considered.

2.4.2.2 The Sparsity Problem

The number of users and items in major e-commerce recommendation systems is very large [Linden et al., 2003]. Even users that are very active result in rating just a few of the total number of items available in a database and respectively, even very popular items result in having been rated by only a few of the total number of users available in the database. This problem, commonly referred to as the sparsity problem, has a major negative impact on the effectiveness of a collaborative filtering approach. Because of sparsity, it is possible that the similarity between two users cannot be defined, rendering collaborative filtering useless. Even when the evaluation of similarity is possible, it may not be very reliable, because of insufficient information processed.

There are several methods that have been proposed to deal with the sparsity problem. Most of them succeed in providing better recommendations, but fail to introduce a general model for dealing with sparsity. One way to overcome the problem of rating sparsity is to employ user profile information when calculating user similarity. That is, two users could be considered similar not only if they rated the same movies similarly, but also if they belong to the same demographic segment. For example, [Pazzani, 1999] uses gender, age, area code, education, and employment information of users in the restaurant recommendation application. This extension of traditional collaborative filtering techniques is sometimes called "demographic filtering" [Pazzani, 1999].

A different approach for dealing with sparse rating matrices is dimensionality reduction. The dimensionality reduction approach addresses the sparsity problem by removing unrepresentative or insignificant users or items so as to condense the user-item matrix. For example, in [Billsus and Pazzani, 1998; Sarwar et al., 2000], Singular Value Decomposition (SVD) was used to reduce dimensionality of sparse ratings matrices. SVD is a well-known method for matrix factorization that provides the best lower rank approximations of the original matrix [Sarwar et al., 2000]. More advanced techniques to achieve dimensionality reduction have been proposed as well. Examples include statistical techniques such as Principle Component Analysis (PCA) [Goldberg et al., 2001] and information retrieval techniques such as Latent Semantic Indexing (LSI) [Billsus and Pazzani, 1998; Deerwester et al., 1990; Hoffmann, 2003].

Furthermore, in [Huang et al., 2004], authors propose to deal with sparsity problem by applying an associative retrieval framework and related spreading activation algorithms to explore transitive associations among consumers through their past transactions and feedback. Other approaches include the utilization of item-based similarity instead of user-based similarity and content-boosted collaborative filtering.

However, potentially useful information might be lost during the dimensionality reduction process. Transitive associations of the associative retrieval technique [Huang et al., 2004], even if they have been successfully employed to deal with the sparsity problem, fail to express the subjective notion of the user-to-user similarity. Item-based [Sarwar et al., 2001; Popescul et al., 2001] in addition to Content-boosted Collaborative Filtering [Melville et al. 2002] approaches require additional information regarding items as well as a metric to compute meaningful similarities among them [Papagelis and Plexousakis, 2005], but in practice, such item information may be difficult or expensive to acquire.

2.4.2.3 The Cold-start Problem

The cold-start problem emphasizes the importance of sparsity problem. *Cold-start* [Schein et al., 2002] refers to the situation in which an item cannot be recommended unless it has been rated by a substantial number of users. This problem applies to new and obscure items and is particularly detrimental to users with eclectic taste. Likewise, a new user has to rate a sufficient number of items before the recommendation algorithm be able to provide reliable and accurate recommendations.

New user problem

In order to make accurate recommendations, the system must first learn the user's preferences from the ratings that the user makes. Several techniques have been proposed to address this problem. Most of them use hybrid recommendation approach, which combines content-based and collaborative techniques. An alternative approach is presented by [Rashid et al., 2002], who explore various techniques for determining the best (i.e., most informative to a recommendation system) items for a new user to rate. These techniques use strategies that are based on item popularity, item entropy, user personalization, and combinations of the above [Rashid et al., 2002].

New item problem

New items are added regularly to recommendation systems. Collaborative systems rely solely on users' preferences to make recommendations. Therefore, until the new item is rated by a substantial number of users, the recommendation system would not be able to recommend it. This problem is usually addressed by using hybrid recommendation approaches.

Chapter 3

Qualitative Analysis of Prediction Algorithms

"If you're not failing every now and again, it's a sign you're not doing anything very innovative"

-W. Allen

Contents

3.1 Introduction

Recommendation systems need to employ efficient prediction algorithms so as to provide users with items that match their interests. Considering a prediction as a value that expresses the likelihood that a user will like an item, a recommendation is defined as the list of the top-N predictions from the set of items available. Improved prediction algorithms indicate better recommendations.

In this chapter, several prediction algorithms are described and evaluated, some of which are novel in that they combine *user-based* and *item-based* similarity measures derived from either *explicit* or *implicit* ratings. To evaluate the accuracy performance of the algorithms we employ both *statistical* and *decision-support* accuracy metrics against different levels of data sparsity and different operational thresholds. The first metric evaluates the accuracy in terms of average absolute deviation, while the second evaluates how effectively predictions help users to select high-quality items.

3.2 Similarity Measures

In this section, a set of similarity measures are presented based on the Pearson correlation coefficient, a metric of relevance between two vectors [Pearson, 1900]. When the values of these vectors are associated with a user's model then the similarity is called *user-based similarity*, whereas when they are associated with an item's model then it is called *item-based similarity*. The similarity measure can be effectively used to balance the ratings significance in a prediction algorithm and therefore to improve accuracy.

There are several similarity algorithms that have been used: cosine vector similarity, Pearson correlation, Spearman correlation, entropy-based uncertainty measure and mean-squared difference. In [Breese et al., 1998] authors suggest that Pearson correlation performs better than cosine vector similarity, while in [Herlock et al., 1999] is indicated that Pearson's correlation performs better than Spearman's correlation, entropy-based uncertainty and mean-squared difference for collaborative filtering. According to these remarks Pearson correlation is selected to compute item-based and user-based similarities taking advantage of both explicit and implicit ratings.

An *explicit rating* identifies the preference of a user to a specific item. A user is prompted by the system's interface to provide ratings for items so as to improve user's model. The more ratings the user provides, the more accurate the recommendations provided are. Ratings range from 1 to 10 with 1 expressing greatest aversion to the item and 10 expressing greatest liking to the item. Explicit ratings are logged by the system and are employed to construct the user's model.

An *implicit rating* [Nichols, 1997; Kleinberg et al., 2001] identifies the preference of a user to specific categories[1]. The term "implicit" is used here somewhat excessively, so as to express that a user is never actually prompted to express a degree

[1] Items in the database belong to categories

of preference to categories. Taking advantage of the fact that an item belongs to a number of categories, it is possible to develop a user model based on category preferences. If the explicit rating of a user to a specific item that belongs to a set of categories is considered "good" then user's model is updated so as to include the preference and vice versa. A rating is considered as "good" when it is greater than or equal to a threshold.

Before describing the algorithms the following definitions are introduced to facilitate the explanation process:

- A set of m users $U=\{u_x: x=1,2,\ldots,m\}$
- A set of n items $I=\{i_x: x=1,2,\ldots,n\}$
- A set of p categories $C=\{c_x: x=1,2,\ldots,p\}$
- A set of q explicit ratings $R=\{r_x:x:1,2,\ldots,q \wedge q\leq m*n\}$
- A set of t implicit ratings $R'=\{r_x': x=1,2,\ldots,t \wedge t\leq m*p\}$
- The explicit rating of a user u_x with reference to an item i_h as r_{u_x,i_h}
- The average explicit rating of a user u_x as $\overline{r}_{u_x}$

In the sequence, three matrices are defined that derive from user's rating activity: the user-item matrix, the user-category matrix and the item-category bitmap matrix.

- *User-item matrix* is a matrix of users against items that have as elements the explicit ratings of users to items. Some of the user-matrix cells are not filled, as there are items that are not rated by any user.
- *User-category matrix* is a matrix of users against item categories that have as elements, values that show the number of times a user has rated positively or negatively for a category. For each category two columns are kept, one for positive ratings and one for negative ratings.
- *Item-category bitmap matrix* is a matrix of items against categories that have as elements the value 1 if the item belongs to the specific category and the value 0 otherwise.

Similarity is computed over the parts of the two vectors that derive from one of these matrices, as it is depicted by the shadowed parts of Figure 3-1.

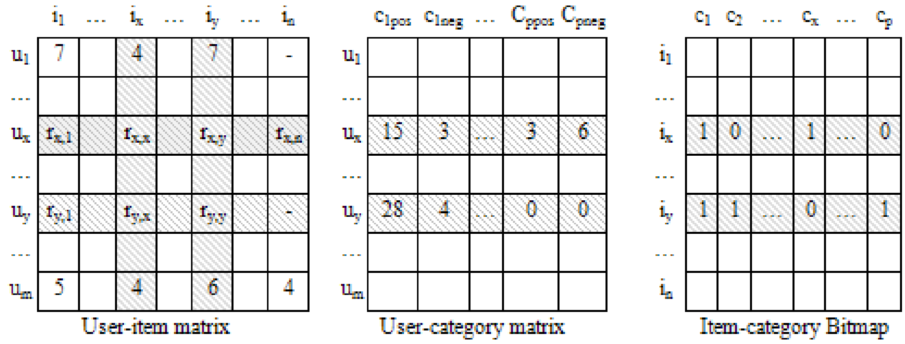

Figure 3-1. User similarities and Item similarities that derive from the user-item, the user-category or the item-category matrix by applying vector similarity measures.

3.2.1 User-based Similarity

3.2.1.1 Based on Explicit Ratings

If the set of items that users u_x and u_y have co-rated is defined as $I' = \{i_x : x = 1, 2, ..., n' \wedge n' \leq n\}$, where n is the total number of items in the database, then the similarity between two users is defined as the Pearson correlation coefficient of their associated rows in the user-item matrix and is given by equation 3.1.

$$\kappa_{x,y} = sim(u_x, u_y) = \frac{\sum_{h=1}^{n'} (r_{u_x,i_h} - \overline{r_{u_x}})(r_{u_y,i_h} - \overline{r_{u_y}})}{\sqrt{\sum_{h=1}^{n'} (r_{u_x,i_h} - \overline{r_{u_x}})^2} \sqrt{\sum_{h=1}^{n'} (r_{u_y,i_h} - \overline{r_{u_y}})^2}} \tag{3.1}$$

3.2.1.2 Based on Implicit Ratings

Whenever an explicit rating is submitted by a user for a specific item, the respective values of the user-category matrix elements are incremented to include the new rating. Thus, it is possible to infer the preference of a user $u_x \in U$ to the category $c_x \in C$ by the user-category matrix. This preference, which is considered as an implicit rating $r'_{u_x,c_x} \in R'$ to that category is computed as $r'_{u_x,c_x} = \frac{c_{x_{pos}}}{c_{x_{pos}} + c_{x_{neg}}} * 10$, where $c_{x_{pos}}$, $c_{x_{neg}}$ are respectively the number of positive and negative ratings[2] that user u_x has implicitly given to category x. Implicit ratings range from 1 to 10, with 1 expressing greatest

[2] Ratings are considered as positive or negative when they are greater or lower than a threshold respectively

aversion to the category and 10 expressing greatest liking to the category. The similarity between the two users is defined as the Pearson correlation coefficient of their implicit ratings to all categories $c \in C$ and is given by equation 3.2, where p is the number of available categories.

$$\lambda_{x,y} = sim(u_x, u_y) = \frac{\sum_{h=1}^{p} (r'_{u_x,c_h} - \overline{r'_{u_x}})(r'_{u_y,c_h} - \overline{r'_{u_y}})}{\sqrt{\sum_{h=1}^{p} (r'_{u_x,c_h} - \overline{r'_{u_x}})^2} \sqrt{\sum_{h=1}^{p} (r'_{u_y,c_h} - \overline{r'_{u_y}})^2}} \tag{3.2}$$

3.2.2 Item-based Similarity

3.2.2.1 Based on Explicit Ratings

If the set of users that have rated both items i_x and i_y is defined as $U' = \{u_x : x = 1, 2, ..., m' \wedge m' \leq m\}$, where m is the total number of users in database, then the similarity between two items is defined as the Pearson correlation coefficient of their associated columns in the user-item matrix and is given by equation 3.3.

$$\mu_{x,y} = sim(i_x, i_y) = \frac{\sum_{h=1}^{m'} (r_{u_h,i_x} - \overline{r_{i_x}})(r_{u_h,i_y} - \overline{r_{i_y}})}{\sqrt{\sum_{h=1}^{m'} (r_{u_h,i_x} - \overline{r_{i_x}})^2} \sqrt{\sum_{h=1}^{m'} (r_{u_h,i_y} - \overline{r_{i_y}})^2}} \tag{3.3}$$

3.2.2.2 Based on Item-category Bitmap

It is also possible to compute the correlation between two items by taking into account the categories in which they belong. In this case, the similarity between two items is defined as the Pearson correlation coefficient of their associated rows in the item-category bitmap matrix and is given by equation 3.4, where p is the number of categories and v_{c_h,i_x} is a Boolean value that equals to 1 if the item x belongs to the category h or 0 otherwise.

$$v_{x,y} = sim(i_x, i_y) = \frac{\sum_{h=1}^{p} (v_{c_h,i_x} - \overline{v_{i_x}})(v_{c_h,i_y} - \overline{v_{i_y}})}{\sqrt{\sum_{h=1}^{p} (v_{u_h,i_x} - \overline{v_{i_x}})^2} \sqrt{\sum_{h=1}^{p} (v_{u_h,i_y} - \overline{v_{i_y}})^2}} \tag{3.4}$$

3.3 Prediction Algorithms

Prediction algorithms [Breese et al., 1998] try to guess the rating that a user is going to provide for an item. This user will be referred as *active user* u_a and this item as *active item* i_a. These algorithms take advantage of the logged history of ratings and of content associated with users and items in order to provide predictions.

3.3.1 Random Prediction Algorithms

The random prediction algorithm represents the worst case of prediction algorithm[3], since instead of applying a sophisticated technique to produce a prediction it generates a random one. The random prediction algorithm serves as a reference point that helps to define how much better results are obtained by the utilization of more sophisticated algorithms.

3.3.2 User-based Prediction Algorithms Description

User-based prediction algorithms are based on user's average rating and an adjustment to it, as given by equation 3.5.

$$prediction = user\_average + adjustment \tag{3.5}$$

The adjustment is most often a weighted sum that integrates user-based or item-based similarity measures. Since prediction arises as the sum of the two, improvements can be considered in both operators. Next, the classic user-based collaborative filtering prediction algorithm is presented and some improvements are suggested that take advantage of the different user-based and item-based similarity measures described in the earlier section.

3.3.2.1 User-based with Explicit Ratings (CF_{UB-ER})

This prediction algorithm represents the classic user-based collaborative filtering prediction algorithm and comes up as the sum of the active user's average rating, regarding the whole set of items that the active user has rated, and an adjustment. The

[3] Actually, this is not absolutely true. Worse prediction algorithms than the random-based one can be artificially produced but in order to do this some kind of logged information is needed

adjustment is a weighted sum of the other users' ratings concerning the active item and their similarity with the active user. The prediction algorithm is given by the equation 3.6, where m' is the number of users that have rated the item i_a and $\overline{r}_{u_a}$ is the user's average rating over the set of items that the active user has rated.

$$CF_{UB-ER} = p_{u_a,i_a} = \overline{r}_{u_a} + \frac{\sum_{h=1}^{m'} k_{a,h}(r_{u_h,i_a} - \overline{r}_{u_h})}{\sum_{h=1}^{m'} |k_{a,h}|} \tag{3.6}$$

3.3.2.2 User-based with Explicit Ratings and Category Boosted ($CF_{UB-ER-CB}$)

Instead of computing the active user's average rating over the total number of rated items, it may be preferable to take into account the active user's average rating over the subset of rated items that belong to the same categories as the active item. This seems reasonable, since user's ratings may be higher for specific item categories and lower for others. The prediction algorithm is given by the equation 3.7, where m' is the number of users that have rated the active item i_a and $\overline{r}_{u_a}$ is user's average rating over the set of items that have been rated by the active user and belong to at least one of the categories that active item i_a belongs to.

$$CF_{UB-ER-CB} = p_{u_a,i_a} = \overline{r}_{u_a} + \frac{\sum_{h=1}^{m'} k_{a,h}(r_{u_h,i_a} - \overline{r}_{u_h})}{\sum_{h=1}^{m'} |k_{a,h}|} \tag{3.7}$$

3.3.2.3 User-based with Implicit Ratings (CF_{UB-IR})

Instead of using the user-based explicit ratings similarity κ, it is possible to use the user-based implicit ratings similarity λ in order to compute the similarity between the active user and the other users. The prediction algorithm is given by the equation 3.8, where m' is the number of users that have rated the item i_a and $\overline{r}_{u_a}$ is the user's average rating over the set of items that the active user has rated.

$$CF_{UB-IR} = p_{u_a,i_a} = \overline{r}_{u_a} + \frac{\sum_{h=1}^{m'} \lambda_{a,h}(r_{u_h,i_a} - \overline{r}_{u_h})}{\sum_{h=1}^{m'} \left|\lambda_{a,h}\right|} \tag{3.8}$$

3.3.3 Item-based Prediction Algorithms Description

Item-based prediction algorithms refer to algorithms that are based on item's average rating and an adjustment to it, as given by equation 3.9.

$$prediction = item\_average + adjustment \tag{3.9}$$

The adjustment is most often a weighted sum that integrates user-based or item-based similarity measures. Since prediction arises as the sum of the two, improvements can be considered in both operators. Next, two item-based algorithms are suggested; an item-based collaborative filtering prediction algorithm based on explicit ratings and an item-based collaborative filtering prediction algorithm based on implicit ratings. Both cases employ the item-based similarity measures described in the earlier section.

3.3.3.1 Item-based with Explicit Ratings

This algorithm can be considered as the reverse of the classic user-based collaborative filtering. First, the item's average rating is computed and then an adjustment is added. The item-based collaborative filtering prediction algorithm comes up as the sum of the active item's average rating, regarding the whole set of users that have rated it, and an adjustment. The adjustment is a weighted sum of the ratings that the active user has given to other items and their similarity with the active item. The prediction algorithm is given by equation 3.10, where n' is the number of items that the active user u_a has rated and $\overline{r}_{i_a}$ is the item's average rating based on all the ratings that have been submitted for it.

$$CF_{IB-ER} = p_{u_a,i_a} = \overline{r}_{i_a} + \frac{\sum_{h=1}^{n'} \mu_{a,h}(r_{u_a,i_h} - \overline{r}_{u_a})}{\sum_{h=1}^{n'} \left|\mu_{a,h}\right|} \tag{3.10}$$

3.3.3.2 Item-based with Implicit Ratings

Instead of using the item-based explicit ratings similarity μ, it is possible to use the item-based implicit ratings similarity ν in order to compute the similarity between the active item and the other items. The prediction algorithm is given by equation 3.11, where n' is the number of items that the active user u_a has rated and $\overline{r}_{i_a}$ is the item's average rating based on all the ratings that have been submitted for it.

$$CF_{IB-IR} = p_{u_a,i_a} = \overline{r}_{i_a} + \frac{\sum_{h=1}^{n'} \nu_{a,h}(r_{u_a,i_h} - \overline{r}_{u_a})}{\sum_{h=1}^{n'} |\nu_{a,h}|} \tag{3.11}$$

3.4 Experimental Evaluation and Results

3.4.1 Data Set

The experimental data comes from an in-house movie recommendation system named MRS. The MRS database currently consists of 2068 ratings provided by 114 users to 641 movies, which belong to at least 1 of 21 categories. Therefore the lowest level of sparsity for the tests is defined as $\frac{114\times641-2068}{114\times641} \simeq 0.9717$. The prediction algorithms are tested over a pre-selected 300-ratings set extracted randomly by the set of 2068 actual ratings. The interested user is strongly encouraged to visit the web site of the system and obtain a more detailed view.

3.4.2 Metrics

Coverage and *accuracy* are two key dimensions on which the quality of a prediction algorithm is usually evaluated. The metrics that are employed to evaluate coverage and accuracy are discussed below.

3.4.2.1 Coverage Metric

Coverage is a measure of the percentage of items for which a recommendation system can provide predictions. A basic coverage metric is the percentage of items for which predictions are available. Coverage can be reduced by defining small neighbourhood sizes or by sampling users to compute predictions. All experimental results demonstrated in this paper had coverage slightly less than perfect for typical level of sparsity (Coverage: 99%, Sparsity: 97.17%, in these experiments). Obviously a prediction cannot be computed in case that the active user has zero correlations with other users.

3.4.2.2 Accuracy Metrics

Several metrics have been proposed for assessing the accuracy of collaborative filtering methods. They are divided into two main categories: *statistical accuracy metrics* and *decision-support accuracy metrics*.

Statistical Accuracy Metrics

Statistical accuracy metrics evaluate the accuracy of a prediction algorithm by comparing the numerical deviation of the predicted ratings from the respective actual user ratings. Some of them frequently used are *Mean Absolute Error (MAE)*, *Root Mean Squared Error (RMSE)* and *Correlation* between ratings and predictions [Herlocker et al., 1999]. All of the above metrics were computed on result data and generally provided the same conclusions.

As statistical accuracy measure, Mean Absolute Error (MAE) [Hofmann, 2003] is employed. Formally, if *n* is the number of actual ratings in an item set, then MAE is defined as the average absolute difference between the *n* pairs $< p_h, r_h >$ of predicted ratings p_h and the actual ratings r_h and is given by equation 3.12.

$$MAE = \frac{\sum_{h=1}^{n} |p_h - r_h|}{n} \tag{3.12}$$

The lower the MAE, the more accurate the predictions would be, allowing for better recommendations to be formulated. MAE has been computed for different prediction algorithms and for different levels of Sparsity. Table 3-1 provides values for the MAE of the different prediction algorithms presented, while Figure 3-2 illustrates the sensitivity of the algorithms in relation to the different levels of sparsity applied.

Table 3-1. Statistical Accuracy of the different prediction algorithms in terms of Mean Absolute Error (MAE) with respect to different Sparsity levels

		Sparsity Levels						
		0.972	0.975	0.98	0.985	0.99	0.995	0.999
Prediction Algorithms	CF_{UB-ER}	1.385	1.457	1.541	1.637	1.801	1.746	1.865
	$CF_{UB-ER-CB}$	1.34	1.412	1.518	1.606	1.807	1.667	1.771
	CF_{UB-IR}	1.703	1.739	1.796	1.781	1.755	1.863	2.147
	CF_{IB-ER}	0.838	0.91	1.06	1.14	1.28	1.626	1.66
	CF_{IB-IR}	1.35	1.38	1.445	1.45	1.521	1.804	1.665
	Random	3.166	3.515	3.414	3.024	3.256	3.174	3.398

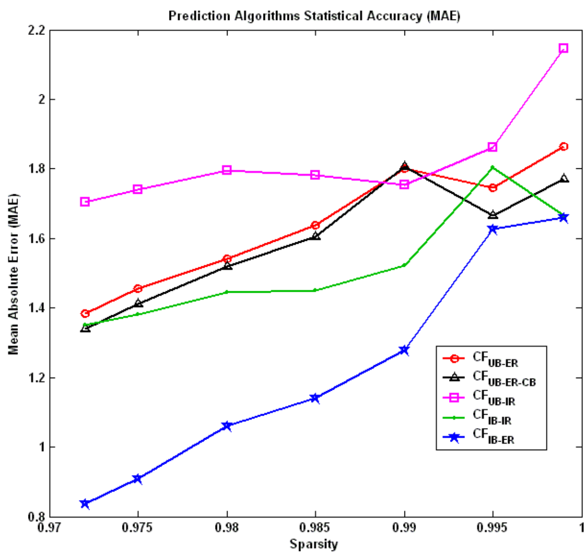

Figure 3-2. Statistical Accuracy of the different prediction algorithms in terms of Mean Absolute Error (MAE) with respect to different Sparsity levels

As far as statistical accuracy is concerned, the following outcomes about the quality performance of the prediction algorithms are reached.

- Performance of item-based prediction algorithms is of superior quality than user-based prediction algorithms
- Performance of implicit rating based algorithms, in the sense that they have been defined for these tests, is of inferior quality than explicit rating based algorithms
- Item-based algorithm, based on explicit ratings (CF_{IB-ER}) seems to be very sensitive to sparsity levels. As sparsity reduces, the MAE of the algorithm decreases, which means that prediction accuracy is increased. CF_{IB-ER} performs as much as 39,5% better than classic Collaborative Filtering prediction algorithm, CF_{UB-ER}, for Sparsity levels close to 97,2%.

- $CF_{UB\text{-}ER\text{-}CB}$ increases the accuracy precision of $CF_{UB\text{-}ER}$, as it calculates the user average based only on the subset of items that belong to the same categories as the active item. However, this increment is insignificant if taking into consideration the extra computation needed to include the category information.

Experimental results indicate that item-based algorithms provide more accurate recommendations than user-based algorithms. In particular, $CF_{IB\text{-}ER}$ behaves much better as data becomes more dense (i.e. sparsity level decreases) in comparison to all other algorithms presented. A prospective recommendation system would provide predictions with an mean absolute error lower than 1 grade (i.e. 0.838). In MRS recommendation system, ratings range from 1 to 10, while in other common systems (e.g. GroupLens, EachMovie dataset) ratings range from 1 to 5. In order to obtain a clear comparative view of presented MAE results, one needs to divide the results with a factor of 2. This consideration leads to MAE of 0,419 in the best case, which is particularly satisfactory for providing high-quality recommendations.

Decision-support Accuracy Metrics

Decision-support accuracy metrics evaluate how effectively predictions help a user to select high-quality items. Some of them frequently used are *reversal rate*, *weighted errors*, *Precision-Recall Curve (PRC) sensitivity* and *Receiver Operating Characteristic (ROC) sensitivity* (Sarwar et al., 1998). They are based on the observation that, for many users, filtering is a binary process. Consequently, prediction algorithms can be treated as a filtering procedure, which distinguishes "good" items from "bad" items.

As decision support accuracy measure, ROC sensitivity is employed. ROC sensitivity is a measure of the diagnostic power of a filtering system. Operationally, it is the area under the receiver operating characteristic (ROC) curve-a curve that plots the sensitivity and the 1-specificity of the test. Sensitivity refers to the probability of a randomly selected "good" item being accepted by the filter. Specificity is the probability of a randomly selected "bad" item being rejected by the filter. The ROC curve plots sensitivity (from 0 to 1) and 1 – specificity (from 0 to 1), obtaining a set of points by varying the quality threshold. The ROC sensitivity range between 0 to 1, where 0.5 is random and 1 is perfect.

If PR denotes the predicted rating, AR denotes the actual rating and the quality threshold as QT, then the following possible cases are defined by the filter for one item

- True Positive (TP) when $PR \geq QT \wedge AR \geq QT$
- False Positive (FP) when $PR \geq QT \wedge AR < QT$
- True Negative (TN) when $PR < QT \wedge AR < QT$
- False Negative (FN) when $PR < QT \wedge AR \geq QT$

For a set of items sensitivity is defined as the True Positive Fraction (TPF) and the 1-specificity as the False Positive Fraction (FPF) where

- $sensitivity = TPF = \frac{tp}{tp + fn}$, where tp, fn is the number of the true positive and the false negative occurrences over the set of items respectively.
- $1 - specificity = FPF = \frac{fp}{fp + tn}$, where tn, fp is the number of the true negative and the false positive occurrences over the set of items respectively.

ROC curve has been computed for different prediction algorithms and for quality thresholds ranging between 1 and 9, while the sparsity level was equal to 0,972. Notation of the form ROC-threshold defines the discrete points on the ROC curve for the specific quality threshold value. The area under the curve represents how much sensitive the prediction algorithm is, so the more area it covers the better for the prediction algorithm. Figure 3-3 illustrates the sensitivity of the different prediction algorithms, while Table 3-2 provides specific values for the ROC-6, ROC-7, ROC-8 and ROC-9 which are of greatest interest. We consider these specific points in ROC curve of greatest interest, because typically an item is considered as "good" if its average rating is over 6, 7, 8, or 9 in a 1-10 rating scale.

Table 3-2. Decision Support Accuracy of the different prediction algorithms in terms of Receiver Operating Curve (ROC) with respect to different Sparsity levels

			Quality Threshold			
			ROC-6	ROC-7	ROC-8	ROC-9
TPF or Sensitivity	Prediction Algorithms	CF_{UB-ER}	0.77	0.55	0.28	0.21
		$CF_{UB-ER-CB}$	0.77	0.59	0.33	0.24
		CF_{UB-IR}	0.75	0.39	0.2	0.1
		CF_{IB-ER}	0.89	0.71	0.53	0.41
		CF_{IB-IR}	0.78	0.53	0.28	0.21

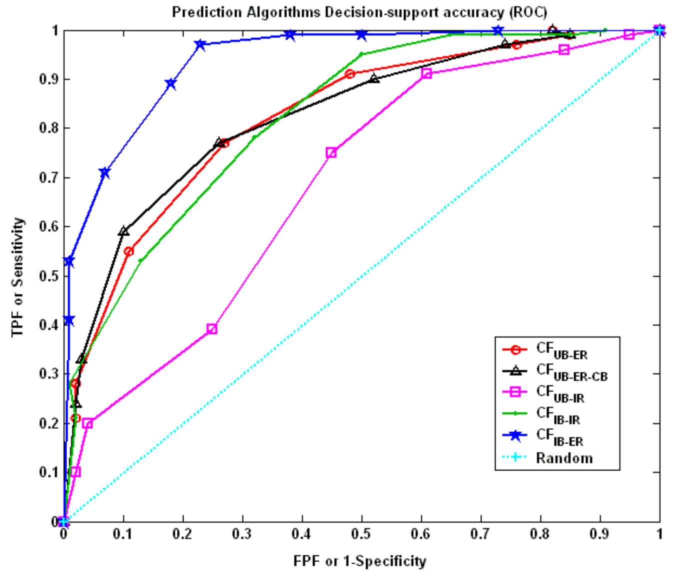

Figure 3-3. Decision Support Accuracy of the different prediction algorithms in terms of Receiver Operating Curve (ROC) with respect to different Sparsity levels

The following remarks can be made about the quality of the prediction algorithms as far as decision-support accuracy is concerned.

- Performance of item-based prediction algorithms is of superior quality than user-based prediction algorithms
- Performance of implicit rating based algorithms, in the context that have been defined in this paper, is of inferior quality than explicit rating based algorithms
- CF_{IB-ER} performs 95% better than classic collaborative filtering, CF_{UB-ER}, for ROC-9, 89% better for ROC-8 and 29% better for ROC-7. This means that if as "good" items are defined the ones that have average rating more than 9 or 8 or 7 respectively and as "bad" items the ones that have average rating less than 9 or 8 or 7 respectively, then CF_{IB-ER} predicts and therefore recommends items with 95% or 89% or 29% respectively more accuracy than classic collaborative filtering CF_{UB-ER}.
- To obtain a clear view of the overall performance of each algorithm one need to compute the area under the ROC curve. It is clear from Figure 3 that CF_{IB-ER} performs much better than every other algorithm examined.

3.5 Concluding Remarks

The vast volume of information flowing on the web has given rise to the need for information filtering techniques. Recommendation systems have been effectively used to filter out excess information and to provide personalized services to users by employing sophisticated, well though-out prediction algorithms. This research work described how explicit ratings can be utilized in order to implicitly obtain user's preference to specific categories. A number of prediction algorithms, based on either user or item similarity, have been designed, implemented and thoroughly evaluated according to their statistical and decision-support accuracy performance. Experimental analysis showed that the performance of item-based prediction algorithms is of superior quality than user-based prediction algorithms. Category-boosted algorithms can lead to slightly better quality when combined with explicit ratings, while performance of prediction algorithms based on implicit ratings is of inferior quality than ones based on explicit ratings.

Chapter 4

Addressing the Scalability Problem

"Not everything that counts can be counted, and not everything that can be counted counts"

-Sign hanging in Einstein's office at Princeton

Contents

4.1 Introduction

Most recommendation systems employ variations of Collaborative Filtering (CF) for formulating suggestions of items relevant to users' interests. However, Collaborative Filtering requires expensive computations that grow polynomially with the number of users and items in the database. Methods proposed for handling this scalability problem and speeding up recommendation formulation are based on approximation mechanisms and, even when performance improves, they most of the time result in accuracy degradation. In this chapter, we describe a method for addressing the scalability problem based on incremental updates of user-to-user similarities. Our Incremental Collaborative Filtering (ICF) algorithm:

i. is not based on any approximation method and gives the potential for high-quality recommendation formulation

ii. provides recommendations orders of magnitude faster than classic Collaborative Filtering and thus, is suitable for online application.

4.2 Incremental Collaborative Filtering

In this section, we present a method to deal with the scalability challenge without compromising recommendation quality. We refer to this method as Incremental Collaborative Filtering (ICF), because it is based on incremental updates of the user-to-user similarities. ICF can be employed to effectively bring on the Web highly scalable and accurate recommendation algorithms.

4.2.1 Methodology

The similarity between user u_x and u_y for the subset of items they have co-rated, is given by equation 4.1.

$$sim(u_x,u_y)=\frac{\sum_{h=1}^{n'}(r_{u_x,i_h}-\overline{r_{u_x}})(r_{u_y,i_h}-\overline{r_{u_y}})}{\sqrt{\sum_{h=1}^{n'}(r_{u_x,i_h}-\overline{r_{u_x}})^2}\sqrt{\sum_{h=1}^{n'}(r_{u_y,i_h}-\overline{r_{u_y}})^2}} \tag{4.1}$$

Whenever a user u_x, submits a new rating or updates the value of an already submitted rating, similarity values between her/him and the rest of the users may need to be re-computed. Our objective is to express the new similarity values between the two users in relation to the old similarity values. This describes an incremental update of their associated similarity. To smoothen the progress of this task we adopt the following notation for the Pearson Correlation similarity measure of equation 4.1:

$$A=\frac{B}{\sqrt{C}\sqrt{D}}\Rightarrow A=sim(u_x,u_y),\quad B=\sum_{h=1}^{n'}(r_{u_x,i_h}-\overline{r_{u_x}})(r_{u_y,i_h}-\overline{r_{u_y}}),\quad C=\sum_{i=1}^{n'}(r_{u_x,i_h}-\overline{r_{u_x}})^2,\quad D=\sum_{i=1}^{n'}(r_{u_y,i_h}-\overline{r_{u_y}})^2$$

Actually, we split the similarity measure into three factors B, C, D, independently calculate the new values of each factor B', C', D' and then combine these values so as to yield the value of the new similarity A' as shown below:

$$A'=\frac{B'}{\sqrt{C'}\sqrt{D'}}\Rightarrow A'=\frac{B+e}{\sqrt{C+f}\sqrt{D+g}},\quad B'=B+e,\quad C'=C+f,\quad D'=D+g$$

where e, f, g are *increments* that need to be computed after either the submission of a new or the update of an existing rating. Next, we split our study, so as to consider the slightly different computations needed for the two special cases. Table 4-1 shows the increments to be computed and Table 4-2 provides proof of equations 4.2 to 4.13.

Table 4-1. Summary of incremental factors that need to be calculated after each rating so as to achieve incremental update of the similarity measure

<table>
<tr><td colspan="4">Submission of a new rating r_{u_a,i_a} for an item i_a by the active user u_a</td></tr>
<tr><td rowspan="3">In case that item i_a has been rated by user u_y</td><td>e</td><td>$e = (r_{u_a,i_a} - \overline{r_{u_a}}')(r_{u_y,i_a} - \overline{r_{u_y}}) - \sum_{h=1}^{n'} d\overline{r_{u_a}}(r_{u_y,i_h} - \overline{r_{u_y}})$</td><td>(4.2)</td></tr>
<tr><td>f</td><td>$f = (r_{u_a,i_a} - \overline{r_{u_a}}')^2 + \sum_{h=1}^{n'} d\overline{r_{u_a}}^2 - 2\sum_{h=1}^{n'} d\overline{r_{u_a}}(r_{u_a,i_h} - \overline{r_{u_a}})$</td><td>(4.3)</td></tr>
<tr><td>g</td><td>$g = (r_{u_y,i_a} - \overline{r_{u_y}})^2$</td><td>(4.4)</td></tr>
<tr><td rowspan="3">In case that item i_a has not been rated by user u_y</td><td>e</td><td>$e = -\sum_{h=1}^{n'} d\overline{r_{u_a}}(r_{u_y,i_h} - \overline{r_{u_y}})$</td><td>(4.5)</td></tr>
<tr><td>f</td><td>$f = \sum_{h=1}^{n'} d\overline{r_{u_a}}^2 - 2\sum_{h=1}^{n'} d\overline{r_{u_a}}(r_{u_a,i_h} - \overline{r_{u_a}})$</td><td>(4.6)</td></tr>
<tr><td>g</td><td>$g = 0$</td><td>(4.7)</td></tr>
<tr><td colspan="4">Update of an existing rating r_{u_a,i_a} for an item i_a by the active user u_a</td></tr>
<tr><td rowspan="3">In case that item i_a has been rated by user u_y</td><td>e</td><td>$e = dr_{u_a,i_a}(r_{u_y,i_a} - \overline{r_{u_y}}) - \sum_{h=1}^{n'} d\overline{r_{u_a}}(r_{u_y,i_h} - \overline{r_{u_y}})$</td><td>(4.8)</td></tr>
<tr><td>f</td><td>$f = dr_{u_a,i_a}^{\ 2} + 2dr_{u_a,i_a}(r_{u_a,i_a} - \overline{r_{u_a}}') + \sum_{h=1}^{n'} d\overline{r_{u_a}}'^2 - 2\sum_{h=1}^{n'} d\overline{r_{u_a}}'(r_{u_a,i_h} - \overline{r_{u_a}})$</td><td>(4.9)</td></tr>
<tr><td>g</td><td>$g = 0$</td><td>(4.10)</td></tr>
<tr><td rowspan="3">In case that item i_a has not been rated by user u_y</td><td>e</td><td>$e = -\sum_{h=1}^{n'} d\overline{r_{u_a}}(r_{u_y,i_h} - \overline{r_{u_y}})$</td><td>(4.11)</td></tr>
<tr><td>f</td><td>$f = \sum_{h=1}^{n'} d\overline{r_{u_a}}'^2 - 2\sum_{h=1}^{n'} d\overline{r_{u_a}}'(r_{u_a,i_h} - \overline{r_{u_a}})$</td><td>(4.12)</td></tr>
<tr><td>g</td><td>$g = 0$</td><td>(4.13)</td></tr>
</table>

Table 4-2. Proof of Equations 4.2 to 4.13

Eq.	Proof
4-2	$B' = \sum_{h=1}^{n''}(r_{u_a,i_h} - \overline{r_{u_a}}')(r_{u_y,i_h} - \overline{r_{u_y}}) \Leftrightarrow B' = (r_{u_a,i_a} - \overline{r_{u_a}}')(r_{u_y,i_a} - \overline{r_{u_y}}) + \sum_{h=1}^{n'}(r_{u_a,i_h} - \overline{r_{u_a}}')(r_{u_y,i_h} - \overline{r_{u_y}}) \Leftrightarrow$ $B' = (r_{u_a,i_a} - \overline{r_{u_a}}')(r_{u_y,i_a} - \overline{r_{u_y}}) + B - \sum_{h=1}^{n'} d\overline{r_{u_a}}(r_{u_y,i_h} - \overline{r_{u_y}})$ $\Rightarrow e = (r_{u_a,i_a} - \overline{r_{u_a}}')(r_{u_y,i_a} - \overline{r_{u_y}}) - \sum_{h=1}^{n'} d\overline{r_{u_a}}(r_{u_y,i_h} - \overline{r_{u_y}})$
4-3	$C' = \sum_{h=1}^{n''}(r_{u_a,i_h} - \overline{r_{u_a}}')^2 \Leftrightarrow C' = (r_{u_a,i_a} - \overline{r_{u_a}}')^2 + \sum_{h=1}^{n'}(r_{u_a,i_h} - \overline{r_{u_a}}')^2 \Leftrightarrow$ $C' = (r_{u_a,i_a} - \overline{r_{u_a}}')^2 + C + \sum_{h=1}^{n'} d\overline{r_{u_a}}^2 - 2\sum_{h=1}^{n'} d\overline{r_{u_a}}(r_{u_a,i_h} - \overline{r_{u_a}}) \Rightarrow$ $f = (r_{u_a,i_a} - \overline{r_{u_a}}')^2 + \sum_{h=1}^{n'} d\overline{r_{u_a}}^2 - 2\sum_{h=1}^{n'} d\overline{r_{u_a}}(r_{u_a,i_h} - \overline{r_{u_a}})$
4-4	$D' = \sum_{h=1}^{n''}(r_{u_y,i_h} - \overline{r_{u_y}})^2 \Leftrightarrow D' = (r_{u_y,i_a} - \overline{r_{u_y}})^2 + \sum_{h=1}^{n'}(r_{u_y,i_h} - \overline{r_{u_y}})^2 \Leftrightarrow$ $D' = (r_{u_y,i_a} - \overline{r_{u_y}})^2 + D \Rightarrow g = (r_{u_y,i_a} - \overline{r_{u_y}})^2$
4-5, 4-6, 4-7	In the case that user u_y has not rated the item i_a, the values of B, C and D are proved in way similar to equations 2, 3 and 4 respectively. In this case the increments e, f and g equal to: $e = -\sum_{h=1}^{n'} d\overline{r_{u_a}}(r_{u_y,i_h} - \overline{r_{u_y}}),\quad f = \sum_{h=1}^{n'} d\overline{r_{u_a}}^2 - 2\sum_{h=1}^{n'} d\overline{r_{u_a}}(r_{u_a,i_h} - \overline{r_{u_a}}),\quad g = 0$
4-8	$B' = \sum_{h=1}^{n'}(r_{u_a,i_h} - \overline{r_{u_a}}')(r_{u_y,i_h} - \overline{r_{u_y}}) \Leftrightarrow B' = (r_{u_a,i_a}' - \overline{r_{u_a}}')(r_{u_y,i_a} - \overline{r_{u_y}}) + \sum_{h=1}^{n'-1}(r_{u_a,i_h} - \overline{r_{u_a}}')(r_{u_y,i_h} - \overline{r_{u_y}}) \Leftrightarrow$ $B' = dr_{u_a,i_a}(r_{u_y,i_a} - \overline{r_{u_y}}) + (r_{u_a,i_a} - \overline{r_{u_a}}')(r_{u_y,i_a} - \overline{r_{u_y}}) + \sum_{h=1}^{n'-1}(r_{u_a,i_h} - \overline{r_{u_a}}')(r_{u_y,i_h} - \overline{r_{u_y}}) \Leftrightarrow$ $B' = dr_{u_a,i_a}(r_{u_y,i_a} - \overline{r_{u_y}}) + \sum_{h=1}^{n'}(r_{u_a,i_h} - \overline{r_{u_a}}')(r_{u_y,i_h} - \overline{r_{u_y}}) \Leftrightarrow$ $B' = dr_{u_a,i_a}(r_{u_y,i_a} - \overline{r_{u_y}}) + B - \sum_{h=1}^{n'} d\overline{r_{u_a}}(r_{u_y,i_h} - \overline{r_{u_y}})$ $\Rightarrow e = dr_{u_a,i_a}(r_{u_y,i_a} - \overline{r_{u_y}}) - \sum_{h=1}^{n'} d\overline{r_{u_a}}(r_{u_y,i_h} - \overline{r_{u_y}})$
4-9	$C' = \sum_{h=1}^{n'}(r_{u_a,i_h} - \overline{r_{u_a}}')^2 \Leftrightarrow C' = (r_{u_a,i_a}' - \overline{r_{u_a}}')^2 + \sum_{h=1}^{n'-1}(r_{u_a,i_h} - \overline{r_{u_a}}')^2 \Leftrightarrow$ $C' = dr_{u_a,i_a}{}^2 + 2dr_{u_a,i_a}(r_{u_a,i_a} - \overline{r_{u_a}}') + (r_{u_a,i_a} - \overline{r_{u_a}}')^2 + \sum_{h=1}^{n'-1}(r_{u_a,i_h} - \overline{r_{u_a}}')^2 \Leftrightarrow$ $C' = dr_{u_a,i_a}{}^2 + 2dr_{u_a,i_a}(r_{u_a,i_a} - \overline{r_{u_a}}') + \sum_{h=1}^{n'}(r_{u_a,i_h} - \overline{r_{u_a}}')^2 \Leftrightarrow$ $C' = dr_{u_a,i_a}{}^2 + 2dr_{u_a,i_a}(r_{u_a,i_a} - \overline{r_{u_a}}') + C + \sum_{h=1}^{n'} d\overline{r_{u_a}}^2 - 2\sum_{h=1}^{n'} d\overline{r_{u_a}}(r_{u_a,i_h} - \overline{r_{u_a}}) \Leftrightarrow$ $\Rightarrow f = dr_{u_a,i_a}{}^2 + 2dr_{u_a,i_a}(r_{u_a,i_a} - \overline{r_{u_a}}') + \sum_{h=1}^{n'} d\overline{r_{u_a}}^2 - 2\sum_{h=1}^{n'} d\overline{r_{u_a}}(r_{u_a,i_h} - \overline{r_{u_a}})$

4-10	$D' = \sum_{h=1}^{n'} (r_{u_y,i_h} - \overline{r_{u_y}'})^2 \Leftrightarrow D' = D \Rightarrow g = 0$
4-11, 4-12, 4-13	In the case that user u_y has not rated the item i_a, the values of B, C and D are proved in way similar to equations 8, 9 and 10 respectively. In this case the increments e, f and g equal to: $e = -\sum_{h=1}^{n'} d\overline{r_{u_a}}(r_{u_y,i_h} - \overline{r_{u_y}}),\quad f = \sum_{h=1}^{n'} d\overline{r_{u_a}'}^2 - 2\sum_{h=1}^{n'} d\overline{r_{u_a}'}(r_{u_a,i_h} - \overline{r_{u_a}}),\quad g = 0$

4.2.1.1 Case 1: Submission of a new rating

To calculate the similarity of u_a and u_y, when the active user u_a submits a *new rating* for the active item i_a, we need to distinguish between two cases:

i. u_y had rated i_a: B, C, D are updated due to the new average of u_a, the new rating of u_a to i_a and the new number of co-rated items

ii. u_y had not rated i_a: B, C are updated due to the new average of u_a.

4.2.1.2 Case 2: Update of an existing rating

To calculate the similarity of u_a and u_y, when the active user u_a updates an existing rating for the active item i_a, we need to distinguish between two cases:

i. u_y had rated i_a: B, C are updated due to the new average of u_a and the new rating of u_a to i_a

ii. u_y had not rated i_a: B, C are updated due to the new average of u_a

4.2.2 Caching

In the previous paragraph, we managed to express B', C' and D' using the former values of B, C and D and the respective increments e, f, g. However, to compute the increments with trivial operations we need to cache the values of B, C and D for all pairs of users, the average rating of each user and the number of items that each user has rated. Part of the cached information needs to be updated after the submission of a new or the update of an existing rating. Table 4-3 explains how each factor that appears in increments e, f, g is computed.

Table 4-3. Computation of the factors that appear in increments *e*, *f* and *g*

Factors	Calculation
B, C, D	Cached Information (For all pairs of users)
m	Cached Information (The number of the items that a user has rated)
$\overline{r_{u_a}}, \overline{r_{u_y}}$	Cached information (Average ratings of all users in database)
$\sum_{h=1}^{n'} r_{u_y,i_h}$, $\sum_{h=1}^{n'} r_{u_a,i_h}$	Cached Information (For each pair of users, the sum of their ratings to co-rated items is cached)
$\overline{r_{u_a}}'$	New average rating of active user: • Submission of a new rating: $\overline{r_{u_a}}' = \frac{r_{u_a,i_a}}{m+1} + \frac{m}{m+1}\overline{r_{u_a}}$ • Update of existing rating: $\overline{r_{u_a}}' = \frac{dr_{u_a,i_h}}{m} + \overline{r_{u_a}}$
r_{u_a,i_a}	Interface (Actual rating of the active user u_a to the active item i_a)
$d\overline{r_{u_a}}$	$d\overline{r_{u_a}} = \overline{r_{u_a}}' - \overline{r_{u_a}} \Leftrightarrow \overline{r_{u_a}}' = \overline{r_{u_a}} + d\overline{r_{u_a}}$ (The difference of user's previous and current average rating)
r_{u_y,i_a}	Database query. (The rating of the user u_y to the item i_a)

4.3 Complexity Issues

In this section, we discuss the computational complexity of the classic Collaborative Filtering and ICF algorithms. We initially present the worst cases and then try to give approximations of the algorithms under real conditions. For each case, our study spans in two directions, the one refers to the complexity of maintaining the user similarities matrix and the other refers to the complexity of formulating a single recommendation to an active user.

In case of Classic Collaborative Filtering, the computation complexity of maintaining the user similarities matrix in worst case is $O(m^2n)$, since we need to compute the similarity between each pair of users according to the subset of their co-rated items. In order to deal with this task, major e-commerce systems prefer to carry out expensive computations offline and feed the database with updated information periodically [Linden et al, 2002]. In this way, they succeed to provide quick recommendations to users, based on pre-computed similarities. These recommendations however, are not of the highest accuracy, because ratings submitted between two offline computations are not considered. Thus, the offline computation method may be detrimental to new or obscure users and items due to their almost undeveloped profile. Alternatively, if user similarities are not pre-computed offline, they may be computed at the time a recommendation is requested. In this case, instead of computing the whole user similarities matrix, only the similarities between the active user and all the rest or a set of training users are computed. The cost of this computation is of the order *O(mn)*. The cost of generating a single recommendation using the Classic Collaborative Filtering is the cost of finding the most similar users to the active user and then scanning their rated items to find the ones that better match with the active user's interests. In the worst case, this computation costs *O(n)* when similarities are pre-computed offline, or *O(mn)* (based on *O(mn)+O(n)*) when similarities are not pre-computed.

In the case of ICF algorithm user-to-user similarities are computed incrementally at the time of rating activity and not at the time that a recommendation is requested. The complexity of this operation is *O(mn)* at worst, as at most *m-1* similarities need to be updated and at most *n* items need to be examined for each user. Since user similarities are considered pre-computed, the cost of generating a single recommendation using the ICF is of the order of *O(n)* in the worst case, as *n* items need to be examined.

Table 4-4. Worst case and Approximation complexities of Classic Collaborative Filtering and ICF

<table>
<tr><td rowspan="2"></td><td colspan="2">Classic CF</td><td colspan="2">Incremental CF</td></tr>
<tr><td>Worst</td><td>Approximation</td><td>Worst</td><td>Approximation</td></tr>
<tr><td>Complexity for maintaining the Similarity Matrix</td><td>$O(m^2 n)$</td><td>$O(mm'n'')$</td><td>$O(mn)$</td><td>$O(m'n')$</td></tr>
<tr><td rowspan="3">Complexity for providing a recommendation to active user</td><td>$O(mn)$</td><td>$O(m'n'')+O(n')$</td><td rowspan="3">$O(n)$</td><td rowspan="3">$O(n')$</td></tr>
<tr><td colspan="2">Pre-computed Offline</td></tr>
<tr><td>$O(n)$</td><td>$O(n')$</td></tr>
</table>

Since sparsity levels are very high in recommendation systems, it is essential to also consider approximations of the complexities in order to estimate the algorithm's performance under real conditions. In order to compute the approximation complexities we define: *m'*, where *m'<<m*, as the number of users with whom the active user has at least one co-rated item; *n'*, where *n'<<n,* as the number of items that have not been rated by the active user and have been rated by at least one of its similar users; *n''*, where *n''<<n*, as the number of co-rated items of the active user and another user. According to these definitions, we can set up the approximations of

the complexities following the discussion of the previous paragraph. Worst case and approximation complexities of Classic Collaborative Filtering and ICF are summarized in Table 4-4.

4.4 Experimental Evaluation

As complexity computation fails to give real time performance and behavior of the algorithms described, we set up an experimental scenario for evaluating the performance of our ICF algorithm as opposed to the Classic Collaborative Filtering. We evaluate the recommendation algorithms presented according to *response time* and *accuracy* metrics as defined below:

response time: Time required by the algorithm to find out the items to recommend.

Accuracy: The fraction of the number of items an algorithm recommends, to the number of items that are recommended by an algorithm that takes into consideration the whole dataset available.

The assumption made here is that recommendations based on the whole dataset are of the highest quality, which is not necessarily true. Indeed, we define this to demonstrate the potential that ICF gives for formulating recommendations based on the complete information in a database and not only a part of it.

Table 4-5. Performance comparison of Classic Collaborative Filtering and ICF

User-item matrix size	Classic CF (Based on sampling)			Incremental CF	
	Samples (#users)	Time (sec)	Accuracy	Time (sec)	Accuracy
100 users x 100 items	10	0.17	22%	0.045	100%
	30	0.55	49.5%		
	50	0.765	67.5%		
	99	1.38	100%		
1000 users x 1000 items	100	6.81	26,7%	0.46	100%
	300	20	53,8%		
	500	33	66.8%		
	999	66	100%		

The experimental scenario is set up so as to depict the level of scalability that both algorithms demonstrate when the active user requests a single recommendation. We employ sparsity level of 92% consider a user to be similar to the active user if their associated Pearson correlation coefficient is greater than 0.65 (in a range of -1 to 1). The values selected represent typical values for recommendation systems and do not influence the results of the experiments. Table 4-5 presents the results of our experiments for user-item matrix of size 100x100 and 1000x1000 respectively. Experiments have been carried out on a 2.80 Mhz, 1G RAM PC.

The following remarks derive from Table 4-5, about the performance of Collaborative Filtering and ICF.

- The trade-off between performance and accuracy in case of Classic Collaborative Filtering is confirmed. Indeed, Classic Collaborative Filtering is very sensitive to the size of samples used. As the sample size increases, accuracy is improved, but the response time also increases and vice versa. Large sample sizes are impractical for online applications due to the slow response time, while small sample sizes are impractical due to accuracy degradation. On the other side, the accuracy of ICF remains as high as 100%, since it is applied to the whole information available.
- ICF proves to be highly-scalable as its response time remains acceptable even for a very large data set. E.g. it provides recommendation in 0.46 seconds for a matrix size of 1000x1000. Classic Collaborative Filtering requires extremely disproportional time to reach a satisfactory accuracy level for large matrix sizes. E.g. when an accuracy level of 66.8% is intended, using a sample of 500 users, in a 1000x1000 matrix Classic Collaborative Filtering performs 71 times slower than ICF
- ICF's performance grows linearly with the number of items, thus in cases of *very* large number of items ICF will probably need to employ some approximation methods

4.5 Concluding Remarks

High dimensionality seems to be the "Achilles' heel" for most of the Collaborative Filtering based recommendation systems. For dealing with this scalability problem, we proposed an incremental method that replaces expensive vector operations with a scalar operation, able to speed-up computations of high dimensional user-item matrices. We named this method Incremental Collaborative Filtering (ICF). ICF is not based on any approximation method and thus, provides the potential of formulating high-quality recommendations. Moreover, pre-computed user to user similarities permit for recommendations to be delivered orders of times faster than with classic Collaborative Filtering. ICF appears to be suitable for online applications, while the methodology described is general and may probably be easily adopted to develop incremental collaborative filtering with the utilization of similarity measures other than Pearson correlation.

Chapter 5

Addressing The Sparsity Problem

"Truth is ever to be found in the simplicity and not in the multiplicity and confusion of things"

-Isaac Newton

Contents

5.1 Introduction

Collaborative Filtering (CF), the prevalent recommendation approach, has been successfully used to identify users that can be characterized as "similar" according to their logged history of prior transactions. However, the applicability of Collaborative Filtering is limited due to the sparsity problem, which refers to a situation that transactional data are lacking or are insufficient. In this chapter, in an attempt to provide high-quality recommendations even when data are sparse, we propose a method for alleviating sparsity using trust inferences. Trust inferences are transitive associations between users in the context of an underlying social network and are valuable sources of additional information that help dealing with the sparsity and the cold-start problems. A trust computational model has been developed that permits to define the subjective notion of trust by applying confidence and uncertainty properties to network associations. Finally, we compare our method with the classic Collaborative Filtering that does not consider any transitive associations. Our experimental results indicate that our method of trust inferences significantly improves the quality performance of the classic Collaborative Filtering method.

5.2 Methodology

In this section we present a method for alleviating the sparsity problem in collaborative filtering based on trust inferences.

5.2.1 Social Networks in Recommendation Systems

Collaborative Filtering has been successfully employed to express the "word-of-mouth" paradigm in a computational context [Shardanand and Maes, 1995]. Common interactions that take place in a typical recommendation system include ratings, transactions, feedback data etc. For the rest of the paper we assume without loss of generality that interactions are based on rating activity. Based on these interactions, it is possible to express similarity conditions between pairs of users, according to the subset of their co-rated items. We view these similarity conditions as associations between users. It is then possible to consider these associations as links of a *social network*. If we define as user-item matrix the matrix having as elements the ratings of users to items, then a user's model [Allen, 1990] is represented in this matrix as an *n*-dimensional vector, where *n* is the number of items in the database. Figure 5-1 illustrates the process of the network construction, where a user's rating activity is used to define network associations.

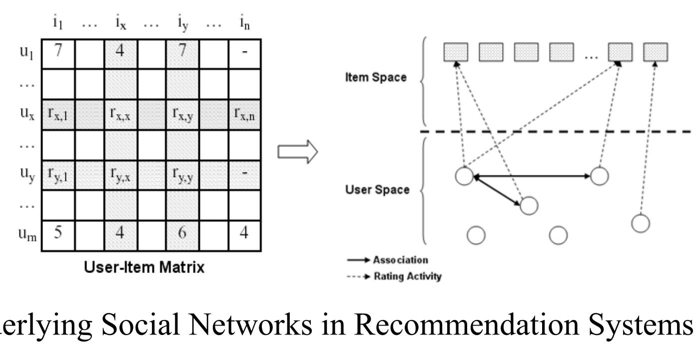

Figure 5-1. Underlying Social Networks in Recommendation Systems

As theories on social networks find application in completely diverse research areas, we need to properly describe their particularities in our context and most importantly identify the process of *membership* and *evolution*.

Membership: A user joins the underlying social network by submitting at least one rating to an item that has previously been rated by another user.

Evolution: Users' ratings to items are enabling the construction of new associations between users and thus new links in the underlying network are considered.

5.2.2 Trust Through User-to-User Similarity

We think of the associations between users as an expression of established *trust* between each other, as far as the specific application area is concerned. Since trust is defined in the context of similarity conditions, the more similar the two users are the greater their established trust would be considered [Ziegler and Lausen, 2004a]. In order to compute the similarity between users, a variety of similarity measures have been proposed, such as Pearson correlation, cosine vector similarity, Spearman correlation, entropy-based uncertainty and mean-square difference. However, in [Breese et al., 1998] and in [Herlocker et al., 1999] it is suggested that Pearson correlation performs better than all the rest.

If we define the subset of items that users u_x and u_y have co-rated as I={i_x: x=1, 2, …, n}, r_{u_x,i_h} as the rating of user u_x to item i_h and $\overline{r_{u_x}}$, $\overline{r_{u_y}}$ as the average ratings of users u_x and u_y respectively, then the established trust between two users is defined as the Pearson correlation [Pearson, 1900] of their associated rows in the user-item matrix (Eq. 5.1).

$$T_{x \to y} = sim(u_x, u_y) = \frac{\sum_{h=1}^{n'} (r_{u_x,i_h} - \overline{r_{u_x}})(r_{u_y,i_h} - \overline{r_{u_y}})}{\sqrt{\sum_{h=1}^{n'} (r_{u_x,i_h} - \overline{r_{u_x}})^2} \sqrt{\sum_{h=1}^{n'} (r_{u_y,i_h} - \overline{r_{u_y}})^2}} \tag{5.1}$$

5.2.3 Trust Inferences

Due to the number of ratings that exist in recommendation systems, underlying social networks are very sparse. There are cases in which insufficient or loss of information is detrimental for the recommendation algorithms. Consider, for example, the case in which associations between users are based on very few data or the case in which there aren't any *k* users to employ in a *k*-nearest neighborhood algorithm. A motivating example is illustrated in Figure 5-2(a). Suppose that users *S*, *N* have rated item I_1 and users *N*, *T* have rated I_2. Classic Collaborative Filtering will associate user *S* with user *N* and user *N* with user *T*, but not user *S* with user *T*. However, a more sophisticated approach that incorporates transitive interactions would recognize the associative relationship between user *S* and user *T* and infer this indirect association. To deal with this problem, we adopt a method of inferring trust between users that are not directly associated to each other. Thus, in the example, it is possible to infer trust between the source user *S* and the target user *T* through the intermediate user *N*. According to this process, trust is propagated in the network and associations between users are built, even if they have no co-rated item.

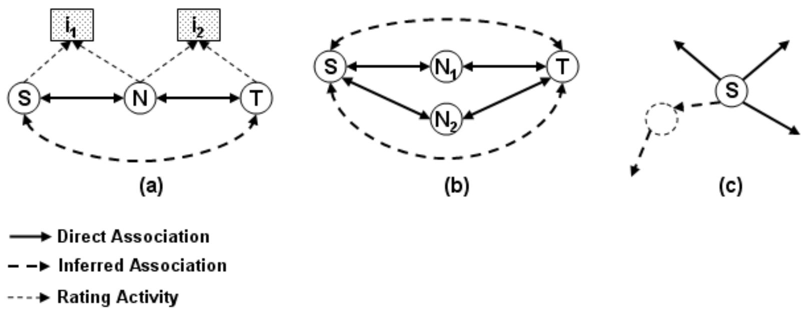

Figure 5-2. Trust Inferences

5.2.3.1 Trust Paths

Propagation of trust [Guha et al., 2004; Ziegler and Lausen, 2004b] implies the existence of *trust paths* in the network. Combination of consecutive direct associations between all intermediate users creates a trust path from a source user to a target user. Trust paths can be of variable *length*, depending on the number of associations that one needs to traverse in order to reach the target user. If *k* associations need to be traversed then the path is considered to be of length *k*. Direct associations are of length *1*, while when the target user is not accessible from the source user, the length of the supposed path is considered infinite.

While computation of trust in direct associations is based on user-to-user similarity, for length-k associations we need to adopt a transitivity rule that facilitates the computation of the inferred trust between the source user and the target user. If we define as N={Ni: i=1, 2, …,k} the set of all intermediate nodes in a trust path that connects user S and user T, then their associated inferred trust is given by Equation 5.2.

$$T_{S \underset{N_1 \to \ldots \to N_k}{\to} T} = \left(\left(\left(\left(T_{S \to N_1} \oplus T_{N_1 \to N_2} \right) \oplus \ldots \right) \oplus T_{N_{k-1} \to N_k} \right) \oplus T_{N_k \to T} \right) \tag{5.2}$$

For example, in order to compute to what degree user S trusts user T in the example of Figure 5-2(a), we need to compute the inferred trust $T_{A \underset{B}{\to} C} = T_{A \to B} \oplus T_{B \to C}$. In Equation 2, we employ the symbol $\oplus$ to denote that we need to apply a special operation in order to compute the inferred trust in the path. If I_x is the set of items that user u_x has rated, and $n(I_x)$ is the cardinality of the set I_x, then Equation 5.3 interprets the special operation employed.

$$T_{S \underset{N}{\to} T} = T_{S \to N} \oplus T_{N \to T} = \oplus \left(\frac{n(I_S \cap I_N)}{n(I_S \cap I_N) + n(I_N \cap I_T)} \left| T_{S \to N} \right| + \frac{n(I_N \cap I_T)}{n(I_S \cap I_N) + n(I_N \cap I_T)} \left| T_{N \to T} \right| \right)$$

$$where \quad \oplus = \begin{cases} +, & if \;\; T_{S \to N} > 0 \;\; and \;\; T_{N \to T} > 0 \\ -, & if \;\; T_{S \to N} > 0 \;\; and \;\; T_{N \to T} < 0 \\ -, & if \;\; T_{S \to N} < 0 \;\; and \;\; T_{N \to T} > 0 \\ \infty, & if \;\; T_{S \to N} < 0 \;\; and \;\; T_{N \to T} < 0 \end{cases} \quad and \quad n(I_{S \to T}) = \left\lceil \frac{n(I_S \cap I_N) + n(I_N \cap I_T)}{2} \right\rceil \tag{5.3}$$

In plain words, in order to compute the inferred trust in a trust path that associates a source user S with a target user T through one intermediate node N, we first compute the weighted sum of the two direct trust associations of S, N and N, T using as weights the number of co-rated items of each direct association, and then apply a sign to the weighted sum according to table 5-1.

Table 5-1. Definition of the sign of the inferred trust in a trust path

	$T_{S \to N} \geq 0$	$T_{S \to N} < 0$
$T_{N \to T} \geq 0$	+	-
$T_{N \to T} < 0$	-	∞

The intuition behind this computation is that:

- If user *S* trusts user *N* and user *N* trusts user *T* then it is inferred that user *S* trusts user *T*
- If user *S* does not trust user *N* and user *N* trusts user *T* then it is inferred that user *S* does not trust user *T*

- If user *S* trusts user *N* and user *N* does not trust user *T* then it is inferred that user *S* does not trust user *T*
- If user *S* does not trust user *N* and user *N* does not trust user *T* then inference is not applicable and the length of the supposed path between user *S* and user *T* is considered infinite

The computed value of the inferred trust is a value that lies between the values of the two direct trust associations as indicated in Equation 5.4 and it is biased towards the value of the direct trust association with the most co-rated items. For example, if $T_{S\to N}=0,7$ based on 5 co-rated items and $T_{N\to T}=0,35$ based on 2 co-rated items, then $T_{\underset{N}{S\to T}}=0,6$. In the same context, if $T_{S\to N}=0,7$ and $T_{N\to T}=-0,35$, then $T_{\underset{N}{S\to T}}=-0,6$.

$$\min\{T_{S\to N},T_{N\to T}\}\leq T_{\underset{N}{S\to T}}\leq \max\{T_{S\to N},T_{N\to T}\} \tag{5.4}$$

5.2.4 Confidence and Uncertainty Properties of Trust Associations

Network evolution is based on individual rating behavior, thus it is reasonable to consider that available structural information defines multiple personalized webs of trust [Ziegler and Lausen, 2004b]. The *personal web of trust* or *local trust* for a user *S* is given through the set of trust paths originating from *S* and passing through users he or she trusts *directly* or *indirectly*. Figure 5-2(c) depicts the notion of personal web of trust. Consequently, a user *S* that interacts with other users in the system develops a *subjective belief* of the network. By subjective belief, we mean that probably what a user in the network believes about *S* is different from what another user in the network believes about user *S*. In order to express this subjective notion of trust we set up a confidence model able to respond to the following interrelated questions:

Q1: How confident user *S* feels of his or her opinion about user *T*?

Q2: What is the uncertainty enclosed in user's *S* opinion about user *T*?

5.2.4.1 Confidence Property

We define as *confidence*, a property assigned to each direct association of the network that expresses the reliability of the association. We make the assumption that confidence is directly related to the number of co-rated items between two users. This assumption indicates that (a) a user's opinion becomes more reliable as additional co-rated items become available and that (b) the reliability of an association between two users may be influenced by the change of the number of co-rated items between other users in the system. For that reason, the more items two users have co-rated, the higher the degree of confidence their association would have. Confidence is applied to each one of a user's direct associations and it is based exclusively on the user's rating activity. In order to compute the confidence of all direct associations of a user, we initially identify the most confident association in an individual's personal web and then express all confidence values of the remaining direct associations in relation to the identified most confident association. We denote the user with which the most confident association has been created as u_{MAX_CONF}. If I_x is the set of items that user u_x has rated, and $n(I_x)$ is the cardinality of the set I_x, then the confidence $C_{S \to T}$ of the association between the source user S and the target user T is given by equation 5.5.

$$C_{S \to T} = \frac{n(I_S \cap I_T)}{n(I_S \cap I_{u_{MAX\_CONF}})} \tag{5.5}$$

Figures 5-3(a) and 5-3(b) show how confidence values of direct associations derive from the number of co-rated items between the source user S and the remaining users in the system. The value of the most confident direct association is always equal to 1, while all other direct associations are equal to or less than 1 as depicted in Figure 5-3(b).

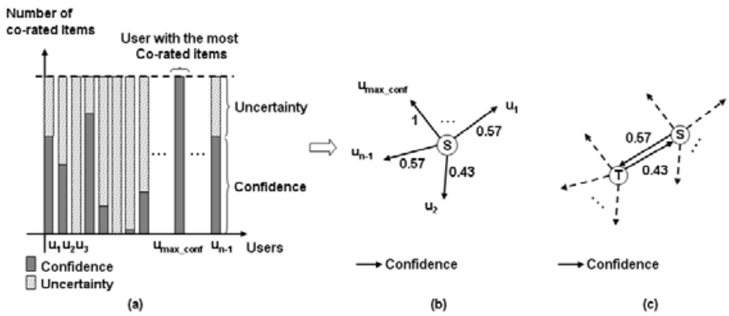

Figure 5-3. Confidence Model to Define Uncertainty and Subjectiveness of Trust

5.2.4.2 Uncertainty Property

The confidence model described earlier can be employed to define *uncertainty* [Josang, 2001]. We define as uncertainty, a property assigned to each direct association of the network that expresses the unreliability of the association. Uncertainty, just like confidence is directly related to the number of co-rated items between two users. This assumption indicates that (a) the uncertainty enclosed to a user's opinion is greater when the number of co-rated items is small and that (b) the uncertainty of an association between two users may be influenced by the change of the number of co-rated items between other users in the system. It becomes obvious that in our model, confidence and uncertainty are contradictory and complementary. Consequently, the more confident one feels about his or her opinion of a user, the less uncertainty is enclosed in his or her opinion of that user and vice versa. Uncertainty $U_{S\to T}$ of the association between the source user *S* and the target user *T* is given by equation 5.6.

$$U_{S\to T} = 1 - C_{S\to T} \tag{5.6}$$

5.2.4.3 Confidence and Uncertainty in Trust Paths

Confidence and uncertainty properties may also be assigned to trust paths. We adopt a transitivity rule that facilitates the computation of the confidence between a source user and a target user through a trust path [Guha et al., 2004; Ziegler and Lausen, 2004b]. If we define the set of intermediate nodes in a trust path that associate a source user *S* with a target user *T* as $N=\{N_i: i=1, 2, ...,k\}$, then the confidence of the trust path is given by Equation 5.7. Accordingly, the uncertainty assigned to the trust path is given by equation 5.8.

$$C_{S \underset{N_1\to\ldots\to N_k}{\to} T} = \left(\left(\left(\left(C_{S\to N_1}\cdot C_{N_1\to N_2}\right)\cdot\ldots\right)\cdot C_{N_{k-1}\to N_k}\right)\cdot C_{N_k\to T}\right) \tag{5.7}$$

$$U_{S \underset{N_1\to\ldots\to N_k}{\to} T} = 1 - C_{S \underset{N_1\to\ldots\to N_k}{\to} T} \tag{5.8}$$

5.2.4.4 Subjectiveness

Since the evolution of personal webs is based on individual rating behavior one would expect that confidence and uncertainty are defined from a user's perspective. Indeed, confidence and uncertainty are *bidirectional* properties. This means that even if two users trust each other as much as what a similarity measure indicates, they do not necessarily have the same confidence in this association. Consider for example, the illustration of Figure 3(c) where there is a direct trust association between user *S* and user *T*. Since computation of trust is based on user similarities their associated trust would be the same for both users. However, user *S* is as much as 0.57 confident about this association, while user T is as much as 0.43 confident about this association. Therefore, our approach is in accordance with the widely accepted position that trust has a subjective notion [Josang, 2001] and reflects the way in which trust is raised in real world social networks.

5.2.5 Managing Multiple Trust Paths

Since trust inferences are based on traversal paths in a network, it is possible to find *multiple paths* that connect two users. Figure 5-4 depicts an example in which a source user S is connected to a target user T through two alternative trust paths P_A and P_B. Path P_A passes through users N_1, N_2, while path P_B through user N_3. The inferred trust in each of these trust paths is independent of the other. Thus, our trust model needs to define a rule that decides which of these inferred trusts to take into consideration. We describe two approaches for inferring trust when there are multiple trust paths available; the first approach is based on *path composition*, while the other is based on *path selection*. For the following approaches we assume that there are p discrete paths between user S and user T.

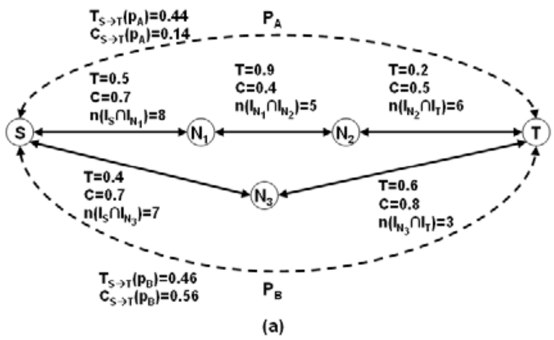

Figure 5-4. Illustrating Example of Multiple Trust Paths

5.2.5.1 Path Composition

The path composition approach tries to combine the values that are inferred by the multiple paths to one single trust value. We distinguish between two methods of composition; *Average Composition* and *Weighted Average Composition*.

Average Composition: We compute the average of all the trust values that are inferred by each of the alternative paths according to Equation 5.9. Despite the fact that this approach is very cost effective it is considered too naive, because it doesn't take into consideration the confidence of each path.

$$T_{S\to T} = \frac{\sum_{i=1}^{p} T_{\underset{P_i}{S\to T}}}{p} \tag{5.9}$$

- *Weighted Average Composition*: We compute the weighted average of the trust inferred by the alternative paths, using for weights the propagated confidence of each inferred association between user S and user T, according to Equation 5.7. This approach is more sophisticated since path confidence is taken into consideration. The final computed trust would be biased to the trust inferred by the most confident path.

$$T_{S\to T} = \sum_{i=1}^{p} \frac{C_{\underset{P_i}{S\to T}}}{\sum_{i=1}^{p} C_{\underset{P_i}{S\to T}}} T_{\underset{P_i}{S\to T}} \tag{5.10}$$

5.2.5.2 Path Selection

The path selection approach tries to identify the most confident path among the paths available. We employ two methods of selection, one based on *Maximum Path Confidence* and one based on *Minimum Mean Absolute Deviation (MAD)*.

- *Selection Based on Path Maximum Confidence*: Based on the confidence of direct association we can compute the confidence of a path in the network according to Equation 5.7. Thus, it is possible to compute the confidence of all discrete paths and then to select the one with the highest degree of confidence. Then, we can use only this path to compute the inferred trust between user *S* and user *T*.

$$T_{S\to T} = \max\{C_{\underset{P_i}{S\to T}} : i = 1,2,...,p\} \tag{5.11}$$

- *Selection Based on Minimum Mean Absolute Deviation (MAD)*: It is possible to order the discrete paths that connect user *S* and user *T*, according to the Mean Absolute Deviation of their direct associations. We consider absolute deviation to be the difference between the confidence values of two consecutive associations. Once all MAD values are computed for each of the paths available we select the one with the minimum MAD as indicated by Equation 5.12, where N is the cardinality of nodes in the path p. This path selection method requires that the path comprises of at least 3 users (i.e. $N \geq 3$). The assumption of this approach is that a path would be more confident when consecutive values of confidence introduce smaller instability.

$$T_{S \to T} = \min\{MAD(P_i) : i = 1, 2, ..., p\}, \quad where \quad MAD(P_i) = \frac{\sum_{k=1}^{N-2} \left\| C_{N_k \to N_{k+1}} \right| - \left| C_{N_{k+1} \to N_{k+2}} \right\|}{N-2} \tag{5.12}$$

5.3 Experimental Evaluation and Results

In this section we evaluate our method for alleviating the sparsity problem using trust inferences. Our evaluation scenario spans across two dimensions. We first evaluate the *impact of trust inferences* to the sparsity problem and then evaluate the *quality of the recommendations* that are based on the underlying network of direct and inferred associations. The experimental data come from our movie recommendation system named *MRS*. The lowest level of sparsity introduced by the system is 0.972 which is a typical sparsity level for recommendation systems, while ratings range from 1 to 10.

5.3.1 Trust Inference Impact

Our first objective was to introduce a method that would lead to additional information accessible for recommendation purposes. We have run tests to discover how much more informative or "dense" is the user-item matrix after applying our method of trust inferences. However, since inferences are dependent on user rating activity we first provide an allocation of ratings that correspond to each user. This helps understanding the peculiarities of our network. Figure 5-5 illustrates the user rating activity in our recommendation system, which seems to follow a *power law distribution* (Zipf distribution) [Faloutsos et al., 1999]. There are a few users that have submitted many ratings, some users with normal number of ratings and many users with a few or even no ratings. It is essential to mention that 38% of users have no rating. This means that in our system there are some users for which no information is available, and therefore recommendations are not possible. However, for the rest users, which are members of the underlying social network, our methodology seems to be beneficial.

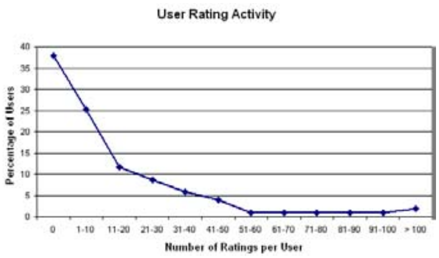

Figure 5-5. User Rating Activity

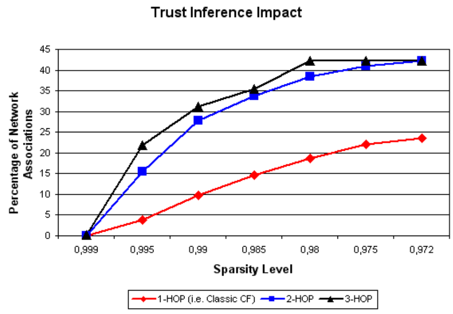

Figure 5-6. Impact of Trust Inference for Different Sparsity Levels

For our experiments, we define as *k*-HOP Collaborative Filtering the method that employs neighbor users that are *k* hops away from the active user. We compute the percentage of user pairs that are feasible in the network when *1*-HOP, *2*-HOP and *3*-HOP Collaborative Filtering algorithms are employed and for different sparsity levels. *1*-HOP Collaborative Filtering represents the classic Collaborative Filtering algorithm, while *2*-HOP Collaborative Filtering and *3*-HOP Collaborative Filtering represent our trust inference based transitive method for 2 and 3 hops away respectively. According to Figure 5-6, the percentage of network associations considered by the Classic Collaborative Filtering are fewer than these considered by our transitive method. This is consistent with our theory, since Classic Collaborative Filtering (*1*-HOP) employs only direct associations, while *2*-HOP Collaborative Filtering and *3*-HOP Collaborative Filtering apply transitive properties in the network. In addition, it is shown that for sparsity level of 0.972, the *1*-HOP Collaborative Filtering considers approximately 24% of the total user pairs, while *2*-HOP and *3*-HOP consider approximately 43% of the total user pairs. It is also demonstrated that after a while *1*-HOP, *2*-HOP and *3*-HOP Collaborative Filtering algorithms reach an *upper limit*. This limit is defined by the percentage of users that are inactive in the system, and therefore are not connected to the underlying network. Furthermore, it is depicted that 3-HOP Collaborative Filtering has similar results to 2-HOP CF, thus for the recommendation quality experiments we only consider the 2-HOP Collaborative Filtering algorithm, which has better time performance.

5.3.2 Recommendation Quality

If a *prediction* is defined as a value that expresses the predicted likelihood that a user will "like" an item, then a *recommendation* is defined as the list of *n* items with respect to the top-*n* predictions from the set of items available. Thus we can reduce the problem of recommendation quality to the problem of prediction quality for our experiments. More accurate prediction algorithms indicate better recommendations. *Statistical accuracy* and *decision-support accuracy* are the key dimensions on which the quality of a prediction algorithm is usually evaluated.

5.3.2.1 Statistical Accuracy Metrics

Statistical accuracy metrics evaluate the accuracy of a prediction algorithm by comparing the numerical deviation of the predicted ratings from the respective actual user ratings. Some of them frequently used are *Mean Absolute Error (MAE)*, *Root Mean Squared Error (RMSE)* and *Correlation* between ratings and predictions [Herlocker et al., 1999]. As statistical accuracy measure, Mean Absolute Error (MAE) is employed. Formally, if *n* is the number of actual ratings in an item set, then MAE is defined as the average absolute difference between the *n* pairs $< p_h, r_h >$ of predicted ratings p_h and the actual ratings r_h and is given by equation 5.13.

$$MAE = \frac{\sum_{h=1}^{n} |p_h - r_h|}{n} \tag{5.13}$$

The lower the *MAE*, the more accurate the predictions are, allowing for better recommendations to be formulated. *MAE* has been computed for Classic Collaborative Filtering and for the four variations of our *2*-HOP Collaborative Filtering method based on trust inferences. The prediction algorithms are tested for different levels of sparsity over a pre-selected *300-ratings set* extracted randomly by the set of actual ratings. Figure 5-7 illustrates the sensitivity of the algorithms in relation to the different levels of sparsity applied.

As far as statistical accuracy is concerned *2*-HOP Collaborative Filtering algorithm outperforms the *1*-HOP Classic Collaborative Filtering for all sparsity levels. For typical sparsity levels of recommendation systems, such as 0.975 and 0.98, *2*-HOP Collaborative Filtering performs as much as 10.1% and 13.1% better than *1*-HOP Collaborative Filtering respectively. In cases that data is extremely sparse, for example when it is equal to 0.99, *2*-HOP Collaborative Filtering performs as much as 17% better than *1*-HOP Collaborative Filtering. Considering that most of the alternative methods proposed for dealing with the sparsity problem result in recommendation quality degradation, the quality performance of our prediction algorithms is very satisfactory.

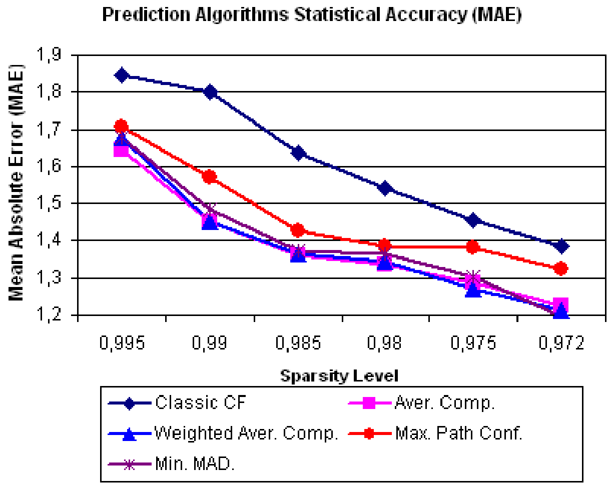

Figure 5-7. MAE of the Classic Collaborative Filtering and the variations of our Collaborative Filtering method of trust inferences for different Sparsity levels

5.3.2.2 Decision-support Accuracy Metrics

Decision-support accuracy metrics evaluate how effectively predictions help a user to select high-quality items. Some of them frequently used are *reversal rate*, *weighted errors*, *Precision-Recall Curve (PRC) sensitivity* and *Receiver Operating Characteristic (ROC) sensitivity*. They are based on the observation that, for many users, filtering is a binary process. Consequently, prediction algorithms can be treated as a filtering procedure, which distinguishes "good" items from "bad" items.

As decision support accuracy measure, ROC sensitivity is employed. ROC sensitivity is a measure of the diagnostic power of a filtering system. Operationally, it is the area under the receiver operating characteristic (ROC) curve, a curve that plots the sensitivity and the 1-specificity of the test. Sensitivity refers to the probability of a randomly selected "good" item being accepted by the filter. Specificity is the probability of a randomly selected "bad" item being rejected by the filter.

If *PR, AR, QT* denote the predicted rating, the actual rating and a quality threshold respectively, then the following possible cases are defined by the filter for one item

- True Positive (TP) when $PR \geq QT \wedge AR \geq QT$
- False Positive (FP) when $PR \geq QT \wedge AR < QT$
- True Negative (TN) when $PR < QT \wedge AR < QT$
- False Negative (FN) when $PR < QT \wedge AR \geq QT$

For a set of items sensitivity is defined as the True Positive Fraction (TPF) and the 1-specificity as the False Positive Fraction (FPF) where

- $sensitivity = TPF = \frac{tp}{tp + fn}$, where tp, fn is the number of the true positive and the false negative occurrences over the set of items respectively
- $1 - specificity = FPF = \frac{fp}{fp + tn}$, where tn, fp is the number of the true negative and the false positive occurrences over the set of items respectively

ROC curve has been computed for different prediction algorithms and for quality thresholds ranging between 1 and 9, while the sparsity level was equal to 0,972. For each prediction we considered a neighborhood of 5 users. The area under the curve represents how much sensitive the prediction algorithm is, so the more area it covers the better for the prediction algorithm. Results are illustrated on Figure 5-8.

As far as decision-support accuracy is concerned the performance of the Collaborative Filtering method based on our method of trust inferences is of superior quality than Classic Collaborative Filtering prediction algorithms, while there is only slight difference between the accuracy performance of the four variations of our Collaborative Filtering method. To obtain a clear view of the overall performance of each algorithm one needs to compute the area under the ROC curve. It is clear from Figure 5-8 that Classic Collaborative Filtering performs much worse than every other algorithm employed based on our method of trust inferences.

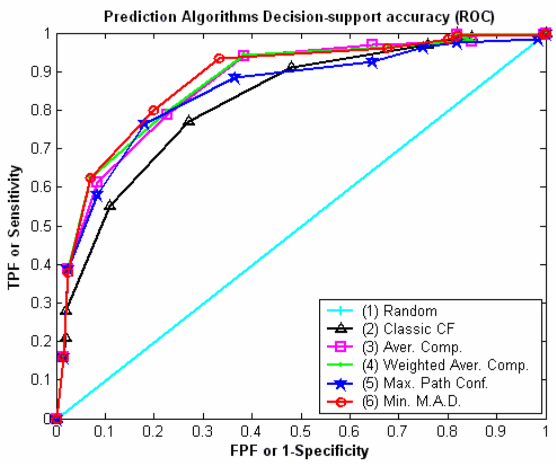

Figure 5-8. ROC for the Classic Collaborative Filtering and the variations of Collaborative Filtering method of trust inferences

5.4 Concluding Remarks

Sparsity is one of the major aspects that limits the application of the Collaborative Filtering method and provokes its success in providing quality recommendation algorithms. In this research, our main objective was to describe a method that is able to provide high-quality recommendations even when information available is insufficient. Our work employs theoretical results of research conducted in areas of social networks and trust management in order to develop a computational trust model for recommendation systems. To deal with the sparsity problem we proposed a method that is based on trust inferences. Trust inferences are transitive associations between users that participate in the underlying social network. Employment of this model provides additional information to Collaborative Filtering algorithm and remarkably relaxes the sparsity and the cold-start problems. Furthermore, our model considers the subjective notion of trust and reflects the way in which it is raised in real world social networks. Subjectiveness is defined in terms of confidence and uncertainty properties that are applied to the network associations. We have experimentally evaluated our method according to the impact that trust inferences have to sparsity and according to recommendation quality. Our experimental results indicate that our method succeeds in providing additional information to the Collaborative Filtering algorithm while it outperforms the quality performance of the classic Collaborative Filtering method. The methodology described is general and may probably be easily adopted to alleviate the sparsity problem in other application areas, especially where underlying social networks can be identified.

Chapter 6

Conclusions

"Those who do not stop asking silly questions become scientists"

-Leon Lederman

Contents

6.1 Summary

The vast volume of information flowing on the web has given rise to the need for development of more sophisticated information retrieval and filtering techniques. Recommendation technologies have been successfully used to filter out excess information and to provide personalized services to users by employing advanced, well though-out architectures. Throughout this research, we tried to obtain a better understanding of the algorithmic foundations of recommendation technologies and to describe methodologies that could overcome certain shortcomings and expand their applicability.

At the beginning, we described how explicit ratings can be utilized in order to implicitly obtain user's preference to specific categories. A number of prediction algorithms have been designed and implemented, based on either user or item similarity and have been thoroughly evaluated according to their statistical and decision-support accuracy performance. Experimental analysis showed that the performance of item-based prediction algorithms is of superior quality than user-based prediction algorithms. Category-boosted algorithms can lead to slightly better quality when combined with explicit ratings, while prediction algorithms based on implicit ratings are of inferior quality than the ones based on explicit ratings.

Subsequently, we focused attention to the scalability problem of recommendation algorithms. High dimensionality seems to be the “Achilles’ heel” for most of the Collaborative Filtering based recommendation approaches. For dealing with the scalability problem, we proposed an incremental method that replaces expensive vector operations with a scalar operation, able to speed-up computations of high dimensional user-item matrices. We named this method Incremental Collaborative Filtering (ICF). ICF is not based on any approximation method and thus, provides the potential of formulating high-quality recommendations. Moreover, pre-computed user to user similarities permit for recommendations to be delivered orders of times faster than with classic Collaborative Filtering. ICF appears to be suitable for online applications, while the methodology described is general and may probably be easily adopted to develop incremental collaborative filtering with the utilization of similarity measures other than Pearson correlation.

Sparsity is one of the major aspects that limits the application of the Collaborative Filtering method and provokes its success in providing quality recommendation algorithms. We described a method that is able to provide high-quality recommendations even when information available is insufficient. Our work employed theoretical results of research conducted in areas of social networks and trust management in order to develop a computational trust model for recommendation systems. To deal with the sparsity problem we proposed a method that is based on trust inferences. Trust inferences are transitive associations between users that participate in the underlying social network. Employment of this model provides additional information to Collaborative Filtering algorithm and remarkably relaxes the sparsity and the cold-start problems. Furthermore, our model considers the subjective notion of trust and reflects the way in which it is raised in real world social networks. Subjectiveness is defined in terms of confidence and uncertainty properties that are applied to the network associations. We have experimentally evaluated our method according to the impact that trust inferences have to sparsity and according to recommendation quality. Our experimental results indicate that our method succeeds in providing additional information to the Collaborative Filtering algorithm while it outperforms the quality performance of the classic Collaborative Filtering method. The methodology described is general and may probably be easily adopted to alleviate the sparsity problem in other application areas, especially where underlying social networks can be identified.

6.2 Extensions of Recommendation Technologies

The current generation of recommendation technologies performed well in several applications, including the ones for recommending books, CDs, and news articles [Mooney, 1999; Schafer et al., 2001]. However, these methods need to be extended for more complex types of applications, such as recommending vacations, financial services, and certain types of movie applications, in order to provide better recommendations.

Recommendation technologies can be extended in several ways that include improving the profiling of users and items, incorporating the contextual information into the recommendation process, supporting multi-criteria ratings, and providing more flexible and less intrusive types of recommendations. Such more comprehensive models of recommendation systems can provide better recommendation capabilities. In the remainder of this section we describe the proposed extensions and also identify various research opportunities for developing them.

6.2.1 Advanced Profiling Techniques

As was pointed out in [Balabanovic and Shoham, 1997; Ungar and Foster, 1998; Konstan et al., 1998; Adomavicius and Tuzhilin 2001a], most of the recommendation methods produce ratings that are based on a limited understanding of users and items as captured by user and item profiles and do not take full advantage of the information in the user's transactional histories and other available data.

For example, classical collaborative filtering methods [Resnick et al., 1994; Hill et al., 1995; Shardanand and Maes, 1995] do not use user and item profiles at all for the recommendation purposes and rely exclusively on the ratings information to make recommendations. Although there has been some progress made on incorporating user and item profiles into some of the methods since the earlier days of recommendation systems, e.g., [Pazzani, 1999; Pennock and Horwitz, 1999; Billsus and Pazzani, 2000], still these profiles tend to be quite simple and do not utilize some of the more advanced profiling techniques.

For example, in addition to using traditional profile features, such as keywords and simple user demographics [Pazzani and Billsus, 1997; Mooney, 1999], more advanced profiling techniques based on *data mining rules* [Fawcett and Provost, 1996; Adomavicius and Tuzhilin, 2001a], *sequences* [Manilla et al., 1995], and *signatures* [Cortes et al., 2000] that describe user's interests can be used to build user profiles. Similar techniques can also be used to build item profiles. Once user and item profiles are built, the most general ratings estimation function can be defined in terms of these profiles and the previously specified ratings.

6.2.2 Extensions for Model-based Recommendations

Some of the model-based approaches provide rigorous rating estimation methods utilizing various statistical and machine learning techniques. However, other areas of mathematics and computer science, such as *mathematical approximation theory* [Powell, 1981; Nurnberger, 1989; Buhmann, 2001], can also contribute to developing better rating estimation methods. One example of an approximation-based approach constitutes *radial basis functions* [Duchon, 1979; Schaback and Wendland, 2001; Buhmann, 2001].

Given a set of points and the values of an unknown function at these points, a radial basis function estimates the values of the function in the whole set of R. One of the advantages of radial basis functions is that they have been extensively studied in the approximation theory, and their theoretical properties and utilization of radial basis functions in many practical applications have been understood very well [Schaback and Wendland, 2001; Buhmann, 2001]. Therefore, it should be interesting to apply them for estimating unknown ratings in recommendation systems.

Therefore, one research challenge is to extend radial basis methods from the real numbers to other domains and apply them to recommendation systems problems. The applicability of other approximation methods for estimating unknown ratings constitutes another interesting research topic.

6.2.3 Multidimensionality of Recommendations

Current generation of recommendation systems operates in the two-dimensional User-Item space. That is, they make their recommendations based only on the user and item information and do not take into the consideration additional contextual information that may be crucial in some applications. However, in many situations the utility of a certain product to a user may depend significantly on time (e.g., the time of the year, such as season or month, the day of the week, or the time of the day, such as morning or evening). It may also depend on the person(s) with whom the product will be consumed or shared and under which circumstances. In such situations it may not be sufficient to simply recommend items to users; the recommendation system must take additional contextual information, such as time, place, and the company of a user, into the consideration when recommending a product.

For example, when recommending a vacation package, the system should also consider the time of the year, with whom the user plans to travel, traveling conditions and restrictions at that time, and other contextual information. As another example, a user can have significantly different preferences for the types of movies she wants to see when she is going out to a movie theater with a boyfriend on a Saturday night as opposed to watching a rental movie at home with her parents on a Wednesday evening.

As was argued in [Herlocker and Konstan, 2001], the inclusion of the knowledge about user's task into the recommendation algorithm in certain applications can lead to better recommendations. In addition, [Adomavicius and Tuzhilin, 2001b] argued that it is important to extend traditional two-dimensional User-Item recommendation methods to multi-dimensional settings, but unfortunately this extension is not always possible.

6.2.4 Multi-criteria ratings

Most of the current recommendation systems deal with single-criterion ratings, such as ratings of movies and books. However, in some applications, such as restaurant recommendations, it is crucial to incorporate multi-criteria ratings into recommendation methods. For example, many restaurant guides, such as Zagat's Guide, provide three criteria for restaurant ratings: food, decor and service.

Although multi-criteria ratings have not yet been examined in the recommendation systems research literature, they have been extensively studied in the OR community over the past few decades [Statnikov and Matusov, 1995; Ehrgott, 2000]. Typical solutions to the multi-criteria optimization problems include (a) finding Pareto optimal solutions, (b) taking a linear combination of multiple criteria and reducing the problem to the single-criterion optimization problem, (c) optimizing only one most important criterion and converting other criteria to constraints, (d) consecutively optimizing one criterion at a time, converting an optimal solution to constraint(s) and repeating the process for other criteria. A typical example of this last approach is the so-called method of successive concessions [Statnikov and Matusov, 1995].

We believe that the problem of finding Pareto-optimal solution set and the iterative method of consecutive single criterion optimizations for multi-criteria recommendation problems mentioned above should also constitute interesting and challenging problems.

6.2.5 Non-intrusiveness

Many recommendation systems are intrusive in the sense that they require explicit feedback from the user and often at a significant level of user involvement. For example, before recommending any newsgroup articles, the system needs to acquire ratings of previously read articles, and often many of them. Since it is impractical to elicit many ratings of these articles from the user, some recommendation systems use non-intrusive rating determination methods where certain proxies are used to estimate real ratings. For example, the amount of time a user spends reading a newsgroup article can serve as a proxy of the article's rating given by this user. Some nonintrusive methods of getting user feedback are presented in [Konstan et al., 1997; Caglayan et al., 1997; Oard and Kim, 1998; Schein et al., 2002]. However, non-intrusive ratings (such as time spent reading an article) are often inaccurate and cannot fully replace explicit ratings provided by the user. Therefore, the problem of minimizing intrusiveness while maintaining certain levels of accuracy of recommendations needs to be addressed by the researchers working on recommendation systems.

One way to explore the intrusiveness problem is to determine an optimal number of ratings the system should ask from a new user. For example, before recommending any movies, MovieLens.org first asks the user to rate a predefined number of movies (e.g., 20). Additional ratings supplied by the user increase the accuracy of recommendations and, therefore, result in certain benefits for the user but also incur certain costs as this process is probably considered unacceptable by the human interaction perspective. Then the intrusiveness problem can be formulated as an optimization problem that tries to find an optimal number of initial rating requests. One interesting intrusiveness-related research problem would be to develop formal models for defining and measuring benefit of supplying *n* initial ratings in terms of the increased accuracy of predictions based on these ratings. Another interesting research opportunity lies in developing marginal cost models that can potentially include cost/benefit analysis of using both implicit and explicit ratings in a recommendation system.

6.2.6 Integration Flexibility

Most of the recommendation methods are inflexible in the sense that they are bound to the systems by the vendors and therefore support only a predefined and fixed set of recommendations. Therefore, the end-user has limited capabilities to customize the types of recommendations according to the user's recommendation needs in real time. This problem has been identified in [Adomavicius and Tuzhilin, 2001b], where a Recommendation Query Language (RQL) and OLAP capabilities have been proposed to address it. RQL is an SQL-like language for expressing flexible user-driven recommendation requests. For example, the request "recommend to each user from Toronto the best three movies that are longer than two hours" can be expressed in RQL as:

RECOMMEND Movie *TO* User
BASED ON Rating *SHOW TOP* 3
FROM MovieRecommender
WHERE Movie.Length > 120 *AND* User.City = "Toronto"

Also, most of the recommendation systems are inflexible since they do not make recommendations at different levels of granularity, i.e., they usually recommend individual items to individual users, and cannot recommend groups of items to groups of users. In some applications it is important to be able to recommend brands or categories of products to certain segments of users [Adomavicius and Tuzhilin, 2001b].

6.2.7 Effectiveness of Recommendations

The problem of developing good metrics to measure effectiveness of recommendations has been extensively addressed in the recommendation systems literature. Some examples of this work include [Mooney, 1999; Herlocker et al., 1999; Yang and Padmanabhan, 2001]. However, in most of the recommendation systems literature, the evaluation of a particular recommendation algorithm is usually limited only to testing its performance in terms of the coverage and accuracy metrics Coverage measures the percentage of items for which a recommendation system is capable of making predictions [Herlocker et al., 1999]. Accuracy measures can be either statistical or decision-support [Herlocker et al., 1999]. Statistical accuracy metrics mainly compare the estimated ratings (e.g., as defined in (16)) against the actual ratings R in the User ×Item matrix, and include Mean Absolute Error (MAE), root mean squared error, and correlation between predictions and ratings. Decision-support measures determine how well a recommendation system can make predictions of high-quality items [Herlocker et al., 1999]. They include classical IR measures of precision (the percentage of ratings classified as positive that are indeed positive) and recall (the percentage of positive ratings classified as positive), F-measure (combined effect of precision and recall), and Receiver Operating Characteristic (ROC) measure demonstrating the tradeoff between true positive and false positive rates in recommendation systems [Herlocker et al., 1999].

Although crucial for measuring accuracy of recommendations, these technical measures often do not capture adequately “usefulness” and “quality” of recommendations. For example, as [Yang and Padmanabhan, 2001] observe for a supermarket application, recommending obvious items, such as milk or bread, that the consumer will buy anyway will produce high accuracy rates; however, it will not be very helpful to the consumer. Therefore, it is also important to develop economics-oriented measures that capture the business value of recommendations, such as return on investments (ROI) and customer lifetime value (CLTV) measures [Schmittlein et al., 1987; Dwyer, 1989; Rosset et al., 2002]. Developing and studying such measures constitutes an interesting research topic in recommendation systems.

6.2.8 Other Extensions

Other important research issues that have been explored in recommendation systems literature include explainability [Billsus and Pazzani, 1999; Herlocker et al., 2000], trustworthiness [Dellarocas, 2002; Xiong and Liu, 2004; Papagelis et al., 2005], scalability [Aggarwal et al., 1999; Goldberg et al., 2001; Schafer et al., 2001; Sarwar et al., 2001], and privacy [Schafer et al., 2001; Ramakrishnan et al., 2001; Kleinberg et al., 2001] issues of recommendation systems.

Bibliography

ADOMAVICIUS, G. AND TUZHILIN, A. (2001A). Expert-driven validation of rule-based user models in personalization applications. Data Mining and Knowledge Discovery, 5(1/2):33-58, 2001.

ADOMAVICIUS, G. AND A. TUZHILIN (2001B). Multidimensional recommender systems: a data warehousing approach. In Proceedings of the Second International Workshop on Electronic Commerce (WELCOM'01). Lecture Notes in Computer Science, vol. 2232, Springer, 2001.

AGGARWAL, C. C., WOLF, J. L., WU, K-L., AND YU, P. S. Horting hatches an egg: A new graphtheoretic approach to collaborative filtering. In Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 1999

ALLEN, R.B. User Models: Theory, method and Practice. International Journal of Man-Machine Studies, 1990

ANSARI, A., ESSEGAIER, S., AND KOHLI, R. Internet recommendations systems. Journal of Marketing Research, pages 363-375, August 2000

BALABANOVIC, M. AND SHOLAM, Y. Combining Content-Based and Collaborative Recommendation. Communications of the ACM, 40 (3), 1997

BAEZA-YATES, R. AND RIBEIRO-NETO, B. Modern Information Retrieval. Addison-Wesley, 1999

BASU, C., HIRSH, H., AND COHEN, W. Recommendation as classification: Using social and content-based information in recommendation. In Recommender Systems. Papers from 1998 Workshop. Technical Report WS-98-08. AAAI Press, 1998

BELKIN, N. AND CROFT, B. Information filtering and information retrieval. Communications of the ACM, 35(12):29-37, 1992

BILLSUS, D. AND PAZZANI, M. Learning collaborative information filters. In International Conference on Machine Learning, Morgan Kaufmann Publishers, 1998

BILLSUS, D. AND M. PAZZANI. A Personal News Agent That Talks, Learns and Explains. In Proceedings of the Third Annual Conference on Autonomous Agents, 1999

BILLSUS, D. AND PAZZANI, M. User modeling for adaptive news access. User Modeling and User-Adapted Interaction, 10(2-3):147-180, 2000

BREESE, J. S., HECKERMAN, D., AND KADIE, C. Empirical analysis of predictive algorithms for collaborative filtering. In Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence, 1998

BUHMANN, M. D. Approximation and interpolation with radial functions. In Multivariate Approximation and Applications. Eds. N. Dyn, D. Leviatan, D. Levin, and A. Pinkus. Cambridge University Press, 2001.

BURKE, R. Knowledge-based recommender systems. In A. Kent (ed.), Encyclopedia of Library and Information Systems. Volume 69, Supplement 32. Marcel Dekker, 2000

CHAUDURY, S. AND DAYAL, U. An overview of data warehousing and OLAP technology. ACM SIGMOD Record, 26(1):65-74, 1997

CHIEN, Y-H. and George, E. I. A bayesian model for collaborative filtering. In Proceedings of the 7th International Workshop on Artificial Intelligence and Statistics, 1999

CLAYPOOL, M., GOKHALE, A., MIRANDA, T., MURNIKOV, P., NETES, D., AND SARTIN, M. Combining content-based and collaborative filters in an online newspaper. In ACM SIGIR'99. Workshop on Recommender Systems: Algorithms and Evaluation, August 1999

CONDLIFF, M., LEWIS, D., MADIGAN, D., AND POSSE, C. Bayesian mixed-effects models for recommender systems. In ACM SIGIR'99 Workshop on Recommender Systems: Algorithms and Evaluation, August 1999

CORTES, C., FISHER, K., PREGIBON, D., ROGERS, A., AND SMITH, HANCOCK, F. A language for extracting signatures from data streams. In Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2000.

COSLEY, D., LAWRENCE, S., PENNOCK, D. M. REFEREE: An open framework for practical testing of recommender systems using ResearchIndex. Proc. of the 28th Very Large Data Bases Conference, 2002

DELLAROCAS, C. The Digitization of Word-of-Mouth: Promise and Challenges of Online Reputation Systems. Working paper, Sloan School of Business, MIT, 2002.

DELGADO, J. AND ISHII, N. Memory-based weighted-majority prediction for recommender systems. In ACM SIGIR'99 Workshop on Recommender Systems: Algorithms and Evaluation, 1999

DEERWESTER, S., DUMAIS, S. T., FURNAS, G. W., LANDAUER, T. K., HARSHMAN, R. “Indexing by Latent Semantic Analysis”. Journal of the American Society for Information Science 41(6), 1990

DUCHON, J. Splines minimizing rotation-invariate semi-norms in Sobolev spaces. In Constructive Theory of Functions of Several Variables, ed. W. Schempp and Zeller, pp. 85-100, Springer 1979

DUDA, R. O., HART, P. E. AND STORK, D. G. Pattern Classification, John Wiley and Sons, Inc., 2001

DWYER, F. R. Customer Lifetime Valuation to Support Marketing Decision Making. Journal of Direct Marketing, Vol 3(4), 1989

EHRGOTT, M. Multicriteria Optimization. Springer Verlag, 2000

FALOUTSOS, M., FALOUTSOS P., FALOUTSOS C. On Power-Law Relationships of the Internet Topology. Proc. of SIGCOMM, 1999

FAWCETT, T., AND PROVOST, F. Combining data mining and machine learning for efficient user profiling. In Proceedings of the Second Intenational Conference On Knowledge Discovery and Data Mining (KDD-96), 1996.

CAGLAYAN, A., SNORRASON, M., JACOBY, J., MAZZU, J., JONES, R., AND KUMAR, K. Learn Sesame: a learning agent engine. Applied Artificial Intelligence, 11:393-412, 1997.

GETOOR, L. AND SAHAMI, M. Using probabilistic relational models for collaborative filtering. In Workshop on Web Usage Analysis and User Profiling (WEBKDD'99), August 1999

GOLDBERG, D., NICHOLS, D., OKI, B. M., AND TERRY, D. Using collaborative filtering to weave an information tapestry. Communications of the ACM, 35(12):61-70, 1992

GOLBECK, J., PARSIA, B., AND HENDLER, J. Trust networks on the Semantic Web. In Proceedings of Cooperative Intelligent Agents, 2003

GOLDBERG, K., ROEDER, T., GUPTA, D., AND PERKINS, C. Eigentaste: A constant time collaborative filtering algorithm. Information Retrieval Journal, 4(2):133-151, July 2001

GUHA,R., KUMAR,R., RAGHAVAN, P., TOMKINS, A. Propagation of Trust and Distrust. Proc. of WWW, 2004

HERLOCKER, J. L. AND J. A. KONSTAN. Content-Independent Task-Focused Recommendation. IEEE Internet Computing, 5(6):40-47, 2001

HERLOCKER, J. L., KONSTAN, J. A., BORCHERS, A., AND RIEDL, J. An algorithmic framework for performing collaborative filtering. In Proc. of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 1999

HERLOCKER, J. L., KONSTAN, J. A., AND RIEDL, J. Explaining Collaborative Filtering Recommendations. Proc. of the ACM Conference on Computer Supported Cooperative Work, 2000

HERLOCKER, J., KONSTAN, J. A., TERVEEN, L. AND RIEDL, J. Evaluating Collaborative Filtering Recommender Systems. ACM Transactions on Information Systems (TOIS), 22 (1), 2004

HILL, W., STEAD, L., ROSENSTEIN, M., AND FURNAS, G. Recommending and evaluating choices in a virtual community of use. In Proceedings of Computer Human Interaction, 1995

HOFMANN, T. Collaborative Filtering via Gaussian Probabilistic Latent Semantic Analysis. Proc. of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 2003

HOLLAND, P. AND LEINHARDT, S. Some evidence on the transitivity of positive interpersonal sentiment. American Journal of Sociology 77, 1205–1209, 1972

HUANG, Z., CHEN, H., ZENG, D. Applying Associative Retrieval Techniques to Alleviate the Sparsity Problem in Collaborative Filtering. ACM Transactions on Information Systems, 22 (1), 2004

JOSANG, A. A Logic for Uncertain Probabilities. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, Vol. 9, No. 3, June, 2001

JUNG, S. Y., AND KIM, T. An Incremental Similarity Computation Method in Agglomerative Hierarchical Clustering. Proc. of the International Symposium on Advanced Intelligent Systems, 2001

KALLES, D., PAPAGELIS, A., ZAROLIAGIS, C. Algorithmic Aspects of Web Intelligent Systems, in: Zhong, N., Liu J., Yao, Y. (eds.), Web Intelligence. Springer-Verlag, pp. 323-345, 2003

KLEINBERG, J., PAPADIMITRIOU, C. H., RAGHAVAN, P. On the Value of Private Information. Proc. of 8th Conference on Theoretical Aspects of Reasoning about Knowledge, 2001

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):77-87, 1997

KONSTAN, J. A., RIEDL, J., BORCHERS, A. AND HERLOCKER, J. L. Recommender systems: a GroupLens perspective. In Recommender Systems. Papers from 1998 Workshop. Technical Report WS-98-08. AAAI Press, 1998

KUMAR, R., RAGHAVAN, P., RAJAGOPALAN, S., AND TOMKINS, A. Recommendation Systems: A Probabilistic Analysis. Journal of Computer and System Sciences, 63(1):42-61, 2001

LANG, K. Newsweeder: Learning to filter netnews. In Proceedings of the 12th International Conference on Machine Learning, 1995

LEE, W. S. Collaborative learning for recommender systems. In Proccedings of the International Conference on Machine Learning, 2001

LILIEN, G. L., KOTLER, P., MOORTHY, K. S. *Marketing Models*, Prentice Hall, 1992

LINDEN, G., SMITH, B., YORK, J. Amazon.com Recommendations: Item-to-Item Collaborative Filtering. IEEE Internet Computing, January, 2003

MANNILA, H., TOIVONEN, H., AND VERKAMO, A. I. Discovering Frequent Episodes in Sequences. In Proceedings of the First International Conference on Knowledge Discovery and Data Mining (KDD-95), 1995.

MCKNIGHT, H. AND CHERVANY N. The meaning of trust. Tech. Rep. MISRC 96-04, Management Informations Systems Research Center, University of Minnesota, 1996

MELVILLE, P., MOONEY, R. J., AND NAGARAJAN, R. Content-Boosted Collaborative Filtering for Improved Recommendations. In Proceedings of the Eighteenth National Conference on Artificial Intelligence, 2002

MOONEY, R. J. Content-based book recommending using learning for text categorization. In ACM SIGIR'99. Workshop on Recommender Systems: Algorithms and Evaluation, 1999.

MOONEY, R. J., BENNETT, P. N., AND ROY, L. Book recommending using text categorization with extracted information. In Recommender Systems. Papers from 1998 Workshop. Technical Report WS-98-08. AAAI Press, 1998

MUI, L., MOHTASHEMI, M., AND HALBERSTADT, A. A computational model of trust and reputation. In Proceedings of the 35th International Conference on System Sciences, 2002

NAKAMURA, A. AND ABE, N. Collaborative filtering using weighted majority prediction algorithms. In Proceedings of the 15th International Conference on Machine Learning, 1998

NICHOLS, D. Implicit Rating and Filtering. Proc. of 5th DELOS Workshop on Filtering and Collaborative Filtering, pp. 31-36, 1997

NURNBERGER, G. Approximation by Spline Functions. Springer-Verlag, 1989

OARD, D. W. AND KIM, J. Implicit feedback for recommender systems. In Recommender Systems. Papers from 1998 Workshop. Technical Report WS-98-08. AAAI Press, 1998.

PAPAGELIS, M. AND PLEXOUSAKIS, D. Recommendation Based Discovery of Dynamic Virtual Communities. In Short Paper Proceedings of the 15th Conference on Advanced Information Systems Engineering, 2003

PAPAGELIS, M. AND PLEXOUSAKIS, D. Qualitative Analysis of User-based and Item-based Prediction Algorithms for Recommendation Agents. Eighth International Workshop on Cooperative Information Agents, 2004

PAPAGELIS, M. AND PLEXOUSAKIS, D. Qualitative Analysis of User-based and Item-based Prediction Algorithms for Recommendation Agents. Journal of Engineering Applications of Artificial Intelligence, 18(4), June, 2005

PAPAGELIS, M., PLEXOUSAKIS, KUTSURAS, T. A method for alleviating the Sparsity Problem in Collaborative Filtering Using Trust Inferences. Proceedings of the 3rd International Conference on Trust Management, 2005

PAPAGELIS, M., PLEXOUSAKIS, D., ROUSIDIS, I., AND THEOHAROPOULOS, E. Qualitative Analysis of User-based and Item-based Prediction Algorithms for Recommendation Systems. 3rd Hellenic Data Management Symposium, 2004

PAPAGELIS, M., ROUSIDIS, I., PLEXOUSAKIS, D., AND THEOHAROPOULOS, E. Incremental Collaborative Filtering for Highly-Scalable Recommendation algorithms. 15th International Symposium on Methodologies of Intelligent Systems, 2005

PAZZANI, M. A framework for collaborative, content-based and demographic filtering. Artificial Intelligence Review, pages 393-408, December 1999

PAZZANI, M. AND BILLSUS, D. Learning and revising user profiles: The identification of interesting web sites. Machine Learning, 27:313-331, 1997

PEARSON, K. Mathematical contribution to the theory of evolution: VII, on the correlation of characters not quantitatively measurable. Phil. Trans. R. Soc. Lond. A, 195,1-47, 1900

PENNOCK, D. M. AND HORVITZ, E., Collaborative filtering by personality diagnosis: A hybrid memory- and model-based approach. In IJCAI'99 Workshop: Machine Learning for Information Filtering, 1999

POPESCUL, A., UNGAR, L. H., PENNOCK, D.M., LAWRENCE, S. Probabilistic Models for Unified Collaborative and Content-Based Recommendation in Sparse-Data Environments. In Proceedings of Uncertainty in Artificial Intelligence, 2001

POWELL, M. J. D. *Approximation Theory and Methods,* Cambridge University Press, 1981

RAMAKRISHNAN, N., B. J. KELLER, B. J. MIRZA, A. Y. GRAMA, AND G. KARYPIS. Privacy Risks in Recommender Systems. IEEE Internet Computing, 5(6):54-62, 2001

RAPOPORT, A. Mathematical models of social interaction. In Handbook of Mathematical Psychology, D. Luce, R. Bush, and E. Galanter, Eds. Vol. 2. Wiley, 1963

RASHID, A. M., ALBERT, I., COSLEY, D., LAM, S. K., MCNEE, S. M., KONSTAN, J. A., AND RIEDL, J. Getting to Know You: Learning New User Preferences in Recommender Systems. In Proceedings of the International Conference on Intelligent User Interfaces, 2002

RESNICK, P., IAKOVOU, N., SUSHAK, M., BERGSTROM, P., AND RIEDL, J. GroupLens: An open architecture for collaborative filtering of netnews. In Proceedings of the 1994 Computer Supported Cooperative Work Conference, 1994

ROSSET, S., NEUMANN, E., EICK, U., VATNIK, N., AND IDAN, Y. Customer Lifetime Value Modeling and Its Use for Customer Retention Planning. In Proceedings The 8th ACM SIGKDD International Conf. on Knowledge Discovery and Data Mining (KDD-2002), 2002.

SALTON, G. *Automatic Text Processing*. Addison-Wesley, 1989

SARWAR, B. M., KARYPIS, G., KONSTAN, J. A., RIEDL, J. T. Application of Dimensionality Reduction in Recommender System: A Case Study. Proc. of ACM SIGKDD, 2000

SARWAR, B., KARYPIS, G., KONSTAN, J., AND RIEDL, J. Item-based Collaborative Filtering Recommendation Algorithms. In Proc. of the 10th International WWW Conference, 2001

SCHABACK, R. AND WENDLAND, H. Characterization and construction of radial basis functions. In Multivariate Approximation and Applications. Eds. N. Dyn, D. Leviatan, D. Levin and A. Pinkus. Cambridge University Press, 2001.

SCHAFER, J. B., J. A. KONSTAN, AND J. RIEDL. E-commerce recommendation applications. Data Mining and Knowledge Discovery, 5(1/2):115-153, 2001.

SCHEIN, A. I., POPESCUL, A., UNGAR, L. H., AND PENNOCK, D. M. Methods and metrics for cold-start recommendations. In Proceedings of the 25th Annual International ACM SIGIR Conference, 2002

SCHMITTLEIN, D. C., MORRISON, D. G., AND COLOMBO, R. Counting Your Customers: Who are they and what will they do next? Management Science, Vol. 33(1), 1987.

SHARDANAND, U. AND MAES, P. Social information filtering: Algorithms for automating 'word of mouth'. In *Proceedings of the Conference on Human Factors in Computing Systems*, 1995

SHETH, B. AND MAES P. Evolving agents for personalized information filtering. In Proceedings of the 9th IEEE Conference on Artificial Intelligence for Applications, 1993

SOBOROFF, I. AND NICHOLAS, C. Combining content and collaboration in text filtering. In IJCAI'99 Workshop: Machine Learning for Information Filtering, 1999

STATNIKOV, R. B. AND MATUSOV, J. B. Multicriteria Optimization and Engineering. Chapman and Hall, 1995.

TERVEEN, L., HILL, W., AMENTO, B., MCDONALD, D., AND CRETER, J. PHOAKS: A system for sharing recommendations. Communications of the ACM, 40(3):59-62, 1997.

TRAN, T. AND COHEN, R. Hybrid Recommender Systems for Electronic Commerce. In Knowledge-Based Electronic Markets, Papers from the AAAI Workshop, Technical Report WS-00-04, AAAI Press, 2000

UNGAR, L. H., AND FOSTER, D. P. Clustering methods for collaborative filtering. In Recommender Systems. Papers from 1998 Workshop. Technical Report WS-98-08. AAAI Press, 1998

XIONG, L. AND LIU, L. Supporting Reputation-based Trust for Peer-toPeer Electronic Communities. IEEE transactions on Knowledge and Data Engineering, 16(7):843-857, 2004

YU, K., XU, X., TAO, J., ESTER, M., AND KRIEGEL, H.-P. Instance Selection Techniques for Memory- Based Collaborative Filtering. In Proceedings of Second SIAM International Conference on Data Mining, 2002

YANG, Y. AND PADMANABHAN, B. On Evaluating Online Personalization, in Proceedings of the Workshop on Information Technology and Systems, pp. 35-41, December 2001.

ZENG, C., XING, C., ZHOU, L. Similarity Measure and Instance Selection for Collaborative Filtering. Proc. of WWW, 2003

ZIEGLER, C.N. AND LAUSEN, G. (2004A). Analyzing Correlation Between Trust and User Similarity in Online Communities. Proc. of the 2nd International Conference on Trust Management, 2004

ZIEGLER, C.N. AND LAUSEN, G. (2004B). Spreading Activation Models for Trust Propagation. Proc. of IEEE International Conference on e-Technology, e-Commerce, and e-Service, 2004