# THÈSE DE DOCTORAT

de

**L'UNIVERSITÉ DE NICE – SOPHIA ANTIPOLIS**

Spécialité

**SYSTÈMES INFORMATIQUES**

présentée par

**Arnd Kohrs**

pour obtenir le grade de

DOCTEUR de l'UNIVERSITÉ DE NICE – SOPHIA ANTIPOLIS

Sujet de la thèse:

# Collaborative Filtering on the Internet

Soutenu le 18 Juillet 2001 devant le jury composé de:

| | | |
|---|---|---|
| Président | Dr. Peter Sander | Professeur, Université De Nice, Sophia Antipolis |
| Rapporteurs | Dr. Marie-France Bruandet | Professeur, Université Joseph Fourier, Grenoble |
| | Dr. Pascal Faudemay | Ingénieur de Recherche, Université Pierre et Marie Curie, Paris |
| Examinateur | Dr. Liming Chen | Professeur, École Centrale de Lyon |
| Membre Invité | Mr. Christophe Ruelle | Directeur Technique, Société Echo, Sophia Antipolis |
| Directeur de Thèse | Dr. Bernard Mérialdo | Professeur, Institut Eurécom, Sophia Antipolis |

# Acknowledgments

I wish to express my sincere appreciation to the people at Institute Eurécom, where this dissertation was written. I would also like to thank all the colleagues and friends for their a warm welcome and the productive atmosphere they created.

Special gratitude is due to Bernard Mérialdo, my research advisor, who accompanied me during this research. His guidance was invaluable for me. I am grateful for the knowledge and experience which I acquired during the work with him.

I would like to express my sincere gratitude to for the members of the jury for taking the time and interest in reading and reviewing this thesis.

I thank my beautiful future wife Carine for her patience and love during the past few years. My close family members, Käte, Richard, Uta and Christa receive my thanks for their encouragement and support.

I thank all the new and old friends, my new in-laws family, my house-mates at "Pitou", and fellow and long-gone doctoral students at Eurécom, who attributed a busy life as a researcher with many diversions and helped me to feel at home at the Côte d'Azur. In particular, I want to thank my friend Jakob for directing guiding me to France for my master thesis work a few years ago, which led my life in an exciting and new direction.

I dedicate this dissertation to the memory of my mother, Anita, who passed away, before she got to see the completion of this work, and without whom this work would not have been possible. Her love, support, dedication, motivation and positive attitude were her precious gifts to me.

# Abstract

Collaborative filtering is a recent software technology that provides personalized recommendations. It is used for recommender systems, and in particular for the personalization of Web sites. Users indicate their preferences (i.e. by rating objects) which the collaborative filtering system uses to match users with similar interests and then predict unknown preferences. This thesis focuses on improving collaborative filtering algorithms and their application in Web sites.

The lack of preference information (a.k.a. sparsity) may lead to failure of collaborative filtering algorithms. We propose a new collaborative filtering algorithm based on hierarchical clustering, which is designed to better address situations with limited preference information.

For a better understanding of Web-based applications which apply collaborative filtering for personalization (user-adapted Web sites), we implemented a prototype user-adapted Web site: the Active WebMuseum is a public online Web museum for art paintings.

In order to improve collaborative filtering with the help of weak content-based information which can automatically be indexed, such as color or texture of paintings, we propose two variations for the combination of collaborative filtering with the conceptually different content-based filtering.

When collaborative filtering is applied in an on-line application, such as the Active WebMuseum, it is important to define how the predictions are used to personalize the application, and how the personalization is valued by the user. We propose the Multi-Corridor-Access-Paradigm as a model for formalizing the user interaction with a user-adapted Web site and therefore allowing the derivation of performance metrics related to the personalization performance.

To improve the training phase when new users start using a collaborative filtering system, we propose several approaches for selecting objects for which preferences should be queried from new users.

# Contents

# List of Figures

# Chapter 1

# Introduction to Collaborative Filtering

While today's media society is a blessing in terms of facilitating access to an abundance of information, it is also a curse in terms of loosing focus on the important considerations and wasting time on the insignificant information issues. New technologies such as the Internet and personal computers everywhere create efficient information channels and virtually bury the consumers of information under a metaphorical *information overload*. While this information overload existed in other forms before the new technologies and electronic format of information arrived, the flow of information was however limited by the physical constraints such as paper and delivery costs. This thesis addresses collaborative filtering, one of the most promising techniques to have been proposed to cope with the overload of electronic information.

Most everybody experiences information overload daily as they are bombarded with information from many sources. At work, people are faced with emails, business newsletters, books, technical articles, manuals, bulletin board discussions, infomercials, or just plain advertisement. More information is needed in today's culture in the form of news articles, stock quotes, weather reports, consumer journals, shopping specials, on-line auctions, automatic notification services, on-line music and video material, etc.

This flow of information presents a challenge to individuals such as grasping the content of all the information in a way useful that is useful to them. Techniques such as skimming or intuitively scrolling and browsing are used to parse the contents of the information. The trade-off is often between the time spent going through the information and the amount or quality of useful information discovered. Sometimes we are looking for a specific piece

of information, for example a technical article about a problem that troubles us, and the task lies in locating the information. On other occasions, we want to keep abreast of the latest events in a certain domain. As indicated, the tasks of parsing or locating information can be quite annoying, repetitive and inefficient if done without automatic support.

Research in the fields of information retrieval and its offspring, information filtering, has been directed at developing technologies that automate the tasks of locating and filtering information. These efforts are mostly based on content indexing. In recent years, a conceptually new technology which does not rely on content indexing, collaborative filtering, has been introduced and is the subject of very active research. Collaborative filtering is used in many systems to address the problems of information overload.

In this dissertation, we first describe the state of the art of collaborative filtering. Then we identify typical problems related to collaborative filtering and suggest algorithms and improvements. We also describe the Active WebMuseum, a personalized Web-based museum for art painting, which is used for the experimentation and evaluation of the new ideas that we have proposed.

The remaining chapter is organized as follows. First, we will briefly define and describe content-based filtering, which has been the prevalent approach to information filtering before the development of collaborative filtering. Second, we will describe automated collaborative filtering and its advantages. Third, we will present a list of the research challenges that we have addressed. Fourth, we will briefly describe the research methodology used in the pursuit of these challenges. We will conclude with an overview of the following chapters of this dissertation.

## 1.1 Content-Based Filtering and Retrieval

Concerned researchers have been addressing information overload related problems by designing technology that automatically extract features from information, i.e. indexing. These features, for example keywords occurring in a text document, are then used to determine if information is relevant to a user. Information is relevant if it satisfies a user's interests or information need. Users may describe their information need in terms of a query, e.g. containing significant keywords. The user's interests may be captured in a user profile containing preferred features contained in information.

These techniques are referred to as content-based since the algorithms rely

on information conveyed in the content of the filtered or retrieved objects. Internet search engines, like *Google*[1], are prominent applications of content-based, here text-based, retrieval. Research in the field of text-based retrieval has matured in many directions to a variety of powerful techniques. For the boolean retrieval the queries are formulated as boolean expressions that describe which keywords do or do not occur in the documents. Information retrieval based on the finer-grained vector-space model constructs frequency vectors of the keywords occurring in a document. In the same way queries or user-profiles can be encoded. By comparing the query vector with the document vectors (for example by measuring the angle between these vectors) relevance can be determined [103, 93]. Other examples of text-based retrieval are probabilistic approaches, where probabilistic models are used to derive retrieval decisions, linguistic strategies that use syntactic properties of the retrieval domain to enhance the results, and cognitive approaches that create semantic models about the information contained in documents.

Similarly, other content types such as image retrieval are addressed, yet not as exhaustively, using similar principles [71, 102, 23, 101]. For most content types other than text, such as images, audio, motion video, and newer multimedia content, information retrieval is less advanced as for text-retrieval due to complexity of content-indexing technologies for these formats.

## 1.2 Filtering vs Retrieval

In the previous section we used the terms filtering and retrieval indifferently because they both serve the previously stated goal to help users with information overload. The distinction between information filtering and information retrieval is that filtering addresses a long term information need, determined by the user's interests, while retrieval focuses on short-term information needs, like specific or immediate problems. Therefore, information filtering can be considered as a variation of information retrieval. Belkin and Croft argue that filtering and retrieval are *two sides of the same coin* [10] and therefore should be approached similarly.

Short-term information needs tend to be volatile and precise, while long-term information needs are broad and evolve slowly. In retrieval the user describes the information need in terms of a query. The retrieval system then matches the query against a generally static corpus of information. In filtering the long-term information need is usually captured in a user profile,

[1] `http//www.google.com`

for example user-defined keywords which occur typically in documents that are of interest to the user. While the keeping of a user profile distinguishes filtering from retrieval, one could argue that the provision of storing a profile is a convenience of a retrieval system to better serve users with long term information needs. An inconvenient filtering system can be envisioned which requires the user to define his user profile (similarly to entering a query) each time the filtering system is used. On the other hand, most retrieval systems could probably benefit from considering long-term information needs when responding to short-term information needs specified by a user's query. For example, the results of a query submitted to a search engine could be reranked by lowering the scores of documents that contain words which do not match the context of previous queries by the same user, especially with ambiguous query terms. So if information need is a fluid concept which at least varies between long and short term (but probably should be modeled even beyond this dimension) then it cannot be a discriminating criteria between filtering and retrieval. Therefore a distinction between retrieval and filtering by this criteria is at the least arbitrary in most cases.

Information filtering systems decide for new pieces of information from a stream of information if they are relevant to the user's information needs. Therefore, filtering and retrieval systems fundamentally differ in their application domains: static corpus vs. information stream. The difference between these two concepts is fluid and most of the time the distinction is difficult: Is the World Wide Web a static corpus or rather an information stream? In most domains where filtering is applied, the information stream is artificially created and really founded on a regularly updated static corpus and repetitive queries. An example is the filtering service *Infobeat*[2] which creates a personalized daily newsletter by querying several underlying newspaper sites. Therefore the distinction between static corpus and information stream is only a weak discriminator between filtering and retrieval.

In view of the above considerations, for the purposes of this dissertation about collaborative filtering, we do not make a distinction between filtering and retrieval. Collaborative filtering is a useful technology for typical filtering applications, such as news filtering [58], but also for typical retrieval application such as bibliographic research [28].

[2]Infobeat: `http://www.infobeat.com`

## 1.3 Limitations of Content-Based Filtering

The classic method used to address the problem of information overload is content-based filtering. The content-based filtering system selects information objects, based on the comparison of the user's profile with the descriptions of the information objects. Content-based systems function well when the information objects (objects for short) and information needs can be effectively described so that filtering models can be developed based on these descriptions. In general, user profiles and objects are described in terms of features. These features are obtained through classification and through indexing the content either automatically or, more expensively, manually.

Therefore the need for indexing and classification of the content leads to a natural limitation of content-based filtering. While for domains such as text-filtering the content can be effectively and efficiently described through automatic keyword indexing, for more complex types of information in the domain of multimedia automatic content indexing, the techniques are not as advanced and therefore less effective. Examples of such content are photographs, music, video and even lyrics. When automatic indexing is not available, content has to be described manually, which introduces a number of difficulties: cost, consistency, coverage, availability of experts, etc. Content-based filtering is limited by the automatic indexing technology available for a particular content type, or the limitations of manual or assisted indexing.

Content-based filtering follows the assumption that the filtering model in use is able to identify relevant material by comparing user profiles with the objects. Content-based filtering will always find relevant objects that relate directly to the user profile. Categories of objects which the user is unaware of and therefore which do not relate to the user profile, but which might be relevant to the user, cannot be identified by content-based filtering. Content-based filtering cannot produce serendipitous finds, therefore limits the user to known categories, and does not allow the exploration for objects which users do not yet know that they like. Generally, content-indexing techniques can only extract superficial attributes from objects, e.g. keywords for text documents. Therefore, content-based filtering cannot compare objects based on more complex attributes such as quality or style. Yet human inspection of objects usually allows for the assessment, either by intuition or through expert knowledge context, of more complex attributes.

Collaborative filtering is a filtering technique which does not rely on content-based indexing but rather on opinions of peer users, and therefore does not limit the space to search for relevant objects to the objects known by the user.

These particularities suggest collaborative filtering as a good complimentary technology for content-based filtering.

## 1.4 Related Work

Collaborative filtering exploits the similarities between different users and relies on the fact that their tastes and opinions are not randomly distributed. Based on this compelling assumption, research in collaborative filtering has led to powerful recommender systems, for all kinds of content types:

- Usenet News article through GroupLens [89].
- Music recommendations by Ringo [99, 98].
- Books at Amazon.com[3].
- Movies at MovieCritic[4].
- Jokes by Jester [36].
- etc ...

In this section, we draw an outline of the history of the developments in the field of collaborative filtering, followed by a presentation of the related research fields.

### 1.4.1 History of Collaborative Filtering

One of the earliest mentions of the term *collaborative filtering* can be found in an article by Goldberg et al. [33]. The article describes *Tapestry*, an email filtering system with a collaborative nature. Tapestry was developed to manage email overload, caused by internal email within the Xerox organization. In order to address huge amounts of email sent to each employee, mailing lists were put into place. While mailing lists gave the users the means to reduce the amount of email by choosing relevant lists, they bore the risk that mails were published on the wrong list so that some would miss relevant information. While content-based filtering techniques for selecting email were

[3]Amazon.com: `http://www.amazon.com`
[4]MovieCritic: `http://www.moviecritic.com`

already in existence [65], Xerox followed the notion that email delivery could be improved by including human judgment in the process. Tapestry filters from a huge amount of email messages which are relevant for a user. Through their email programs, users could annotate messages and attribute personal assessments about relevance. The user-added information was then made available to other users of the system through a proprietary query language similar to SQL. For example, users could ask the system to retrieve all emails which were deemed relevant by a certain peer user whose judgment they trust or whose interests are similar. The technology developed was named "Collaborative Filtering". While Tapestry is one of the first instances of the application of collaborative filtering, it is very different from later developments in the field. The main difference is that the Tapestry system relied on the fact that users of the system knew each other. Relationships in interest between the users could not be exploited automatically and therefore Tapestry or similar systems are applicable for smaller groups. Tapestry and later developments of generalized collaborative applications such as Grassroots [49] refine this notion of closed collaborative groups, but do not provide a viable solution for information filtering which scales well with the amount of users.

While relevant literature often cites the Tapestry system as the first collaborative filtering system, older claims exist. In a 1989 U.S. patent, Hey claims inventions for a system and methods which use sampled user reactions to predict unsampled user reactions [43]. In a later patent [42] he extents his claims that such a system can be used for recommending items. Based on his inventions he also developed a recommender system software "Likeminds"[60, 35, 24]. A movie recommender system has been developed as a showcase for the Likeminds software: MovieCritic [61].

A lot of attention was given to collaborative filtering during the "Human Factors in Computing Systems 1995 (CHI'95)"conference: Maltz and Ehrlich [66] identify a distinction between active and passive collaborative filtering. They observed the existing praxis in organization that some people are more active than others in disseminating information and for others, e.g. by forwarding calls for papers to the appropriate researchers. This led to the notion of *active*. *Passive* means in this context that the collaborative filtering system is passive and does not recommend unless solicited by the user. This is in contrast to "Active"collaborative filtering where recommendations are triggered by other peer actions, e.g. forwarding of an email containing a joke to friends. Also at CHI'95 the term *Virtual Community* was coined in the context of an email-based collaborative filtering system for video recommendations [44]. Virtual communities provide synergies similar to real communities without

the social relationship requirement.

Another important contribution to CHI'95 was an article describing *Ringo*, an email-based music recommender system based on collaborative filtering [99]. While describing interesting aspects about the algorithm used and proving the effectiveness of collaborative filtering, the authors described the task of collaborative filtering as "automating the word-of-mouth". Due to the social nature of the word-of-mouth they referred to collaborative also as *Social Information Filtering* [98].

Collaborative filtering was broadly acknowledged as a viable technology for recommender systems through March, 1997, issue of the *Communications of the ACM* which was dedicated to recommender systems [90]. Here one of the most remarkable articles presented was about *GroupLens* [58]. GroupLens uses collaborative filtering to furnish recommendations for Usenet articles. The beauty of GroupLens lies in the integration of the prediction results within the news reader software. Some news reader applications, such as Gnus for Emacs, were extended so that the user was not necessarily aware that a collaborative filtering system recommended articles. The recommendations of collaborative filtering were used to augment the appearance of the Usenet articles in the reader software, for example by arranging the order in which articles are read. The technique used in GroupLens is referred to as *Automated Collaborative Filtering*, since the server-based collaborative filtering system determines similarities in interest between users automatically without the requirement that users know each other.

GroupLens is also the name of the research group at the University of Minnesota dedicated to the field of collaborative filtering. Research by GroupLens led to a remarkable body of work in collaborative filtering[89, 58, 97, 34, 40, 41, 69]. GroupLens also led to the formation of the software company NetPerceptions, which develops on-line marketing software based on collaborative filtering.

Also presented in the dedicated issue of the Communications of the ACM is the bookmark recommender system *Siteseer* [91]. In contrast to other collaborative filtering systems presented until now which use explicit actions by the user through rating or annotation to evaluate objects, Siteseer collects bookmark files of participating users as implicit evidence for the preferences of the users. Based on similarities with other users' bookmark files, the system provides personalized recommendations for URLs.

Another bookmark recommender system also presented is this journal is PHOAKS [104, 45]. Similarly to Siteseer, the PHOAKS system uses implicit information about the relevance of URLs by mining archives of old Usenet

articles for the mentions of URLs, following the reasonable assumption that important Web pages are mentioned more frequently in the appropriate forum. PHOAKS, however, can not provide personalized recommendation but provides non-personalized recommendations for relevant URLs related to topics discussed in news groups. While not personalized, the system is very useful for finding relevant material for a subject with which one is not yet familiar, for example for locating Web pages containing FAQs related to a programming language which is discussed in a dedicated Usenet group.

### 1.4.2 Classification of Collaborative Filtering

The previous section showed many approaches to collaborative filtering. The broad notion is that the individual user benefits from the experiences of other users for the identification of objects and information. Some efforts have been made to classify collaborative filtering approaches. Maltz et al. [66] distinguish between active and passive collaborative filtering:

**Active** collaborative filtering systems allow so-called mediators to annotate items and forward them to appropriate parties and therefore the mediator is required to make filtering decisions. Consequently, an active collaborative filtering system is rather the infrastructure to support the filtering part of collaborative work.

**Passive** collaborative filtering systems, in contrast, let target user retrieve recommendations, such as recommender systems.

The GroupLens group refines the term collaborative filtering to

**Automatic** collaborative filtering [40]. The attribute "automatic"refers to the fact that the system automatically identifies interest similarities between users without the requirement that the users know each other. Therefore, an active collaborative filtering system is scalable for large anonymous groups, for example the visitors of Web sites. The automatic nature has other implications for the system, such as a model for preferences and recommendations which is in a computable form such as numerical ratings.

This dissertation concerns automatic collaborative filtering. Since other forms of collaborative filtering are not discussed, for the sake of simplicity,

the term "collaborative filtering"refers to the automatic form in the following chapters.

Breese et al. [15] further divide automatic collaborative filtering algorithms into two categories: memory-based and model-based:

**Model-based:** Model-based algorithms create a model for the filtering process which is based on previous preferences. The model is then used to calculate the prediction for the filtering process. Examples are Bayesian decision trees [15, 38] and cluster models such as presented in this thesis (see Chapter 2) [51, 13, 74].

**Memory-based:** The second category of algorithms does not use a predictive model but calculates predictions based on all known preference information. Examples of such algorithms are the correlation approaches based on Pearson correlation [89], or approaches based on weighted-majority voting [29, 26, 27].

The advantage of model-based approaches is that models can be created before they are used to fulfill the filtering task, i.e. off-line, and in general model-based approaches provide fast predictions. However, the ability to precalculate the models is also a burden, since the models cannot quickly be updated when new data is available.Memory-based approaches always use all data available by definition so that updating of data is not an issue. However, these algorithms do not scale well with increasing numbers of users since the amount of data to be considered in calculations increases proportionally. Generally, the creation of models also implies the risk of introducing overly broad assumptions, for example default values for undefined preferences.

In this dissertation, the distinction between these two categories of algorithms is not considered to be of high importance. The use of models can be a design decision during the development of a collaborative filtering system considering the trade-off between scalability, accuracy and other factors. In a similar way as document classification is used to reduce the computational complexity of textual information retrieval systems [70], model-based approaches can be used to ease the burden of memory-based algorithms [77].

### 1.4.3 Collaborative Filtering Embodiments

In previous sections, example applications of collaborative filtering have been described. Here we list the applications of collaborative filtering systems in a more general way:

**Recommender Systems:** Recommender systems identify objects from a large set of objects which are most useful or interesting for the user.

This term is often used as an alternative term for collaborative filtering within the collaborative filtering community because most collaborative filtering systems manifest themselves as recommender.

**Intelligent Agents:** While there is no precise definition what an intelligent agent is, some characteristics are commonly associated with agents. The characteristics include adaptability to the user, i.e. the agent learns the user's preferences so that the user may achieve tasks more efficiently. Various approaches to collaborative filtering have considered the agent paradigm. Balabanovic uses a multi-agent system for adapting a URL recommender system and to combine collaborative and content-based filtering [6, 7]. Sarwar and et al. also use a multi-agent approach to combine collaborative and content-based filtering [97, 34] by having agents express artificial preferences based on content-based features.

Other examples exist where the fact that the filtering application intelligently assists the user during filtering tasks leads to the notion of an intelligent agent [30, 72, 20, 59, 103]

**Personalization Services:** Personalization services aim at individualizing offerings. Prominent examples are customized news-letters and targeted advertisement [92]. For example, the bookstore Amazon.com sends emails to recent customers promoting objects which relate to previous purchases, either determined by content or by collaborative filtering based on similar purchasing behavior of other customers [96]. Adaptive Web sites, as discussed in chapter 3 with the example of a personalized museum, are also examples where personalization services are added to Web sites to increase the site's value for the individual user. In contrast Perkowitz [85] et al. use data mining on Web server access logs to augment the structure of a Web site. However, their approach does not lead to personalized Web sites.

The concepts of recommender systems, intelligent software agents, and personalization services, overlap. Often the goals for the use of collaborative filtering vary while the algorithms and technologies used are similar. Very often the task can be abstracted as predicting user preferences. Therefore in this dissertation, the embodiment of collaborative filtering is of only minor interest. The main focus is the underlying algorithms that determine the success or failure of collaborative filtering systems.

### 1.4.4 Reference Systems

In this section two reference collaborative filtering systems are presented.

#### 1.4.4.1 GroupLens

GroupLens is a system that applies collaborative filtering to personalize the service of the Usenet. It provides an open architecture wherein people rate articles and in return their reader software may request recommendations from the GroupLens recommender system.

The developers were guided by specific requirements:

- That the system integrates well with existing news-reader applications, which led to an open protocol specification for the remote interactions between the collaborative filtering service and the Usenet reader (see Figure 1.1).
- That the provisions for user feedback be efficient, which led to a numeric 5-point rating scale so that users may rate articles with just one keystroke.
- That the system predict the ratings which the user might give instead of selecting some articles for the user, so that the system provides advice rather than censorship.

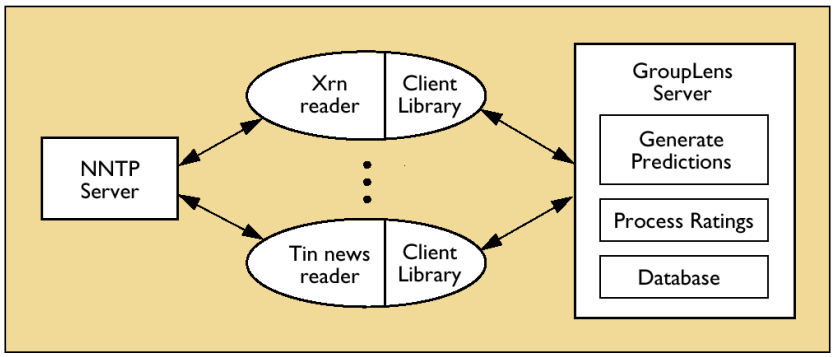



Figure 1.1: The GroupLens architecture overview: The open GroupLens architecture provides client libraries, which allow Usenet reader client software to interact with the GroupLens server. The client software interacts with the Usenet independently of GroupLens [58].

The GroupLens researchers where mainly concerned with the following issues:

- The integration of collaborative filtering into an information filtering environment with existing applications and users.
- Addressing the distributed nature of the Usenet.
- Working with very limited sets of ratings.
- Scalability of the systems performance to number of users and newsgroups.

After pilot field trials, a prototype system was made available for public use over several months. Various news reader user-agents were available for download from the group's Web site (see Figure 1.2 for an example). Due to performance issues of the experimental system, the recommendation service was limited to a small subset of general popular newsgroups. GroupLens provided recommendations for articles within a newsgroup for a target user by comparing the target user's ratings for articles within that newsgroup with other users' ratings in that same newsgroup.

In this dissertation a similar model of collaborative filtering is pursued, such as numeric ratings are used to predict ratings which the user might give.

#### 1.4.4.2 Fab

Fab is a URL recommender system for the world wide Web. The Fab system used collaborative and content-based filtering combined. The target user logs into the system on a regular basis and the system creates a page containing URLs referencing the most recommended pages for the target user (see Figure 1.3 for an example).

Fab is based on a multi-agent model (See Figure 1.4). The systems hosts several classes of entities:

**Collection Agents:** These agents collect pages from the World Wide Web, in order to make a pre-selection for the supported user base. Several strategies of collection have been implemented, for example querying search engines with typically occurring keywords of the supported user base. In general collection agents adapt to special or broad interests. The profile of the collection agent is updated according to the success of the collected documents with the users.

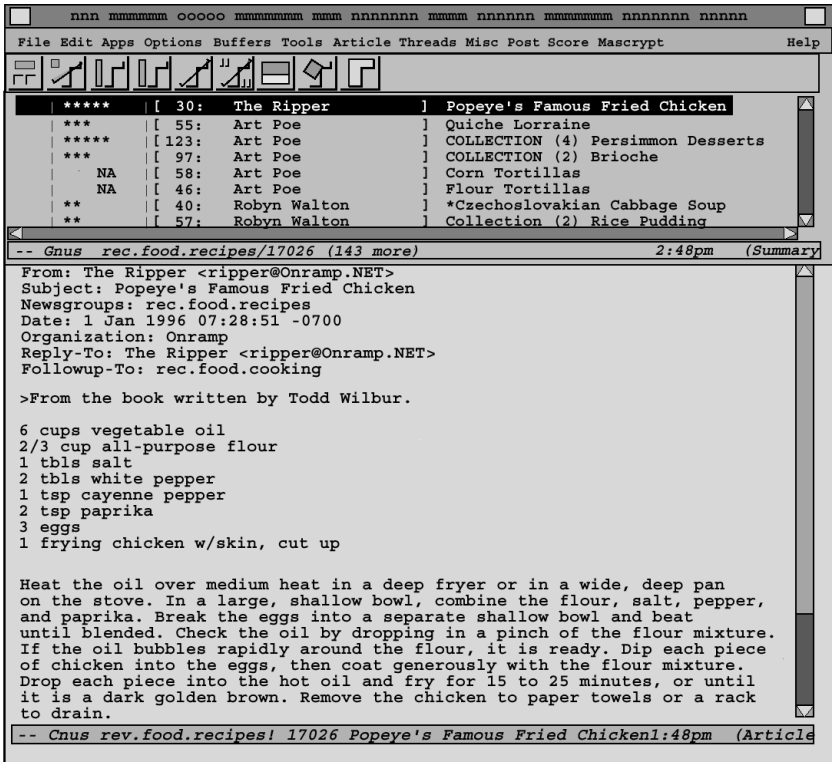



Figure 1.2: GroupLens extended Usenet reader software: This figure depicts the Gnus news reader software which is integrated within the Emacs application. Here the recommendations are used to augment the appearance of the summary lines for the articles of a newsgroup. The number of stars indicates the predicted rating returned by the GroupLens system [58].


Figure 1.3: Fab example: The Fab recommender system provides personalized lists of recommended URLs, which may be rated [7].

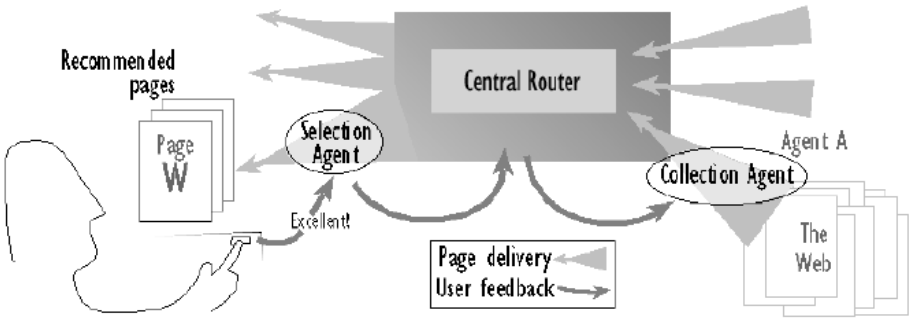



Figure 1.4: Fab architecture: The Fab system consists of collection agents for representing topics, selection agents for representing users and a central router which mediates between the agents [7].

**Selection agents:** Selection agents represent the users within the system. These agents store the user profile in terms of weighted keywords. The user profile is updated when a user rates a document, i.e. URL.

**Central Router:** The central router mediates between collection and selection agents. The central router compares the weighted keyword profiles of the agents and forwards URLs when a high similarity is detected. Feedback, based on user ratings, is forwarded in reverse in order to continuously adapt the agents' profiles.

The Fab system does not store user ratings directly. User ratings are used to adapt the keyword-based user profiles. Furthermore, the Fab system relies heavily on the comparability of documents and user profiles, so that it can be considered a content-based rather than a collaborative filtering system. Fab could even be useful for a single user. Collaborative filtering is introduced through the shared use (and therefore shared training) of collection agents. Furthermore, Fab provides short-cut recommendations: if a document is highly rated by one user, the system also forwards the same document to users with similar user profiles.

However, the use of systems similar to Fab for collaborative filtering would be very limited due to the requirement of a strong underlying content-based filtering technology.

This dissertation pursues a collaborative filtering model which differs greatly from Fab , since the scope is towards broad application for a variety of filtered objects.

## 1.5 Focus of Dissertation

In the graphic: Content-based = Content-Based

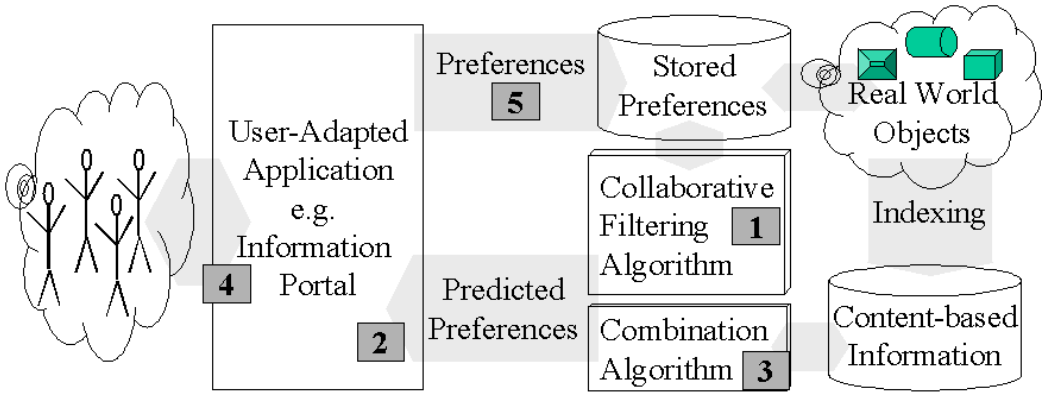



Figure 1.5: Overview of challenges in collaborative filtering: The challenges addressed in this dissertation are indicated in this overview by bold numbers in boxes:

**1** How can collaborative algorithms cope with sparsity?

**2** How can predicted preferences lead to an adapted application?

**3** How can collaborative filtering be combined with content-based filtering for complex media?

**4** How can the usefulness of personalization in user-adapted applications be measured?

**5** How can the querying for preferences be improved?

This dissertation presents significant results related to five challenges in collaborative filtering. Figure 1.5 offers an overview of where these challenges occur.

### 1.5.1 Sparsity of Preference Information

Collaborative filtering predicts preferences which a target user might have for objects or pieces of information. It does so by deducing predicted pref-

erences from other users who have shown agreement with the target user on preferences for other objects.

The lack of preference information leads to failure of collaborative filtering. First, because similarity between users cannot be determined, and second, because other preferences may not be available from which to deduce.

Lack of preferences is very typical in collaborative filtering systems. Typical cases of lack of preferences are listed below:

**New-User Case:** Users for whom no or very few preferences are known are difficult to match against other users.

**New-Object Case:** It is difficult to predict preferences for objects for which very few preferences are known.

**Bootstrap Case:** A combination of both cases arises when a new collaborative filtering system is created. In general users provide preferences to a collaborative filtering system with the incentive to receive predicted preferences in turn. This mechanism leads to a bootstrap dilemma. New collaborative filtering systems with very few known user preferences lack the incentive to attract users.

Sparsity of preference information is addressed in this dissertation in the following ways:

- A collaborative filtering algorithm was designed based on hierarchical clustering which exploits the given preference information more thoroughly than commonly used algorithms.
- A combination of collaborative filtering with content-based filtering is used to combine the benefits of both approaches and therefore improve prediction accuracy.
- Especially for addressing the *New User Case*, techniques are explored to make the gathering of user preferences more effective and efficient by smartly selecting objects to be evaluated.

### 1.5.2 Creating User-Adapted Web sites

Applications which are hosted by a Web site bear the inherent advantage that they may support multiple users and therefore make collaborative filtering

feasible as a supportive technology. A promising use of collaborative filtering is the personalization of a Web site through adaptation to the user's predicted preferences. The following issues need to be addressed.

- General guidelines are necessary for the integration of collaborative filtering into a Web-based application.
- Furthermore, to determine the usefulness of a user-adapted application for a user, evaluation measures are necessary.

This dissertation addresses these issues through the following approaches:

- The Active WebMuseum, a prototype user-adapted Web application was developed and implemented. This application uses collaborative filtering to achieve a high degree of personalization.
- The presentation of objects within the application based on predicted preferences and the user's interaction with them was formalized as the *multi-corridor-access-paradigm* which helps to evaluate the usefulness of personalization efforts.

## 1.6 Research Methodology

A major part of the results produced in this dissertation was mostly achieved through experimentation. Off-line experiments were designed to learn about the effectiveness of enhancements and modifications of real collaborative filtering algorithms using real collaborative filtering datasets.

Furthermore, our research was founded on an on-line prototype collaborative filtering system, which applied algorithms and methods under study.

The experimentation and the prototype required the implementation of a large software library. This library (using C++ classes) allowed the implementation of experiments. Furthermore, the library was used to partially implement the on-line prototype.

In this section we first describe the most important algorithms which are used in this dissertation as well as the mathematical model which is used for the modeling of collaborative filtering. The general approach of the experimentation together with datasets and error measurements is then explained.

## 1.6.1 Algorithms and Models

### 1.6.1.1 The Rating Matrix

Collaborative filtering systems collect the users' preferences as ratings on a numerical scale, which leads to a large matrix $rating(user, object)$ (in short $r_{u,o}$). Each row represents the preferences of one user for all known objects. The rating matrix is in general very sparse due to many unknown preferences. The task of a collaborative filtering algorithm lies in predicting the undefined preferences of the users, i.e. the gaps in the ratings matrix.

### 1.6.1.2 Pearson Prediction

Several algorithms have been proposed on how to use the rating matrix to predict ratings [99, 15].

For the research leading to this dissertation, we derived a collaborative filtering algorithm from a commonly used technique, first proposed for the GroupLens project [89] and also used in Ringo [99], which is based on Pearson vector correlation. In the following we describe the underlying formulas in more detail to provide the reader with a better understanding of the general idea of automatically using other users as expert recommenders.

Usually, the task of a collaborative filtering system is to predict the rating of a particular user $u$ for an object $o$. The system compares the user $u$'s ratings with the ratings of all other users, who have rated the considered object $o$. Then a weighted average of the other users' ratings is used as a prediction.

If $O_u$ is the set of objects that a user $u$ has rated, then we can define the mean rating of user $u$ as:

$$\overline{r_u} = \frac{1}{|O_u|} \sum_{o \in O_u} r_{u,o}$$

Collaborative filtering algorithms predict the ratings based on the ratings of similar users. When *Pearson* correlation is used, similarity is determined from the correlation of the rating vectors of user $u$ and the other users $u'$:

$$\rho(u,u') = \frac{\sum_{o \in O_u \cap O_{u'}} (r_{u,o} - \overline{r_u})(r_{u',o} - \overline{r_{u'}})}{\sqrt{\left(\sum_{o \in O_u \cap O_{u'}} (r_{u,o} - \overline{r_u})^2\right)\left(\sum_{o \in O_u \cap O_{u'}} (r_{u',o} - \overline{r_{u'}})^2\right)}}$$

It can be noted that $\rho \in [-1, +1]$.

The correlation coefficient $\rho$ measures the similarity between the two users' rating vectors. A high absolute value signifies high similarity and a low absolute value dissimilarity.

The prediction formula is based on the assumption that the unknown ratings can be expressed as the average rating of the target user plus a deviation which can be expressed as a weighted sum of the deviations of similar users from their respective means. The weights refer to the amount of similarity between the target user $u$ and the other users.

$$p^{collab}(u,o) = \overline{r_u} + k \sum_{u' \in U_o} \rho(u,u')(r_{u',o} - \overline{r_{u'}})$$

$$\text{with} \begin{cases} U_o & \text{: Users, who rated object } o. \\ k & = \frac{1}{\sum_{u' \in U_o} \rho(u,u')} \end{cases}$$

(The factor $k$ normalizes the weights.)

Sometimes the correlation coefficient between two users is undefined because the users have not rated common objects, i.e. $O_u \cap O_{u'} = \varnothing$. We found in our experiments that assuming a default value for the correlation between the rating vectors is helpful when the dataset is very small. For example, we measured in experiments $\rho_{default} = 0.2$ as the mean of typically occurring correlation coefficients in our dataset. The application of a default correlation biases the prediction toward the mean, which might stabilize the prediction results for very small datasets. To avoid this bias, we did not use default correlation in our later described experiments, instead, if correlation between a target and a peer user can not be measured, the peer user is ignored for the prediction for the target users.

In the rest of this document we refer to this approach as the *Pearson* algorithm, since it is based on the Pearson correlation coefficient.

A detailed discussion of this algorithm, its various parameters, and variations has been contributed by Shardanand [98, 99] and also more recently by Herlocker [41]. Breese et al. [15] compared Pearson prediction with an algorithm based on vector similarity, an algorithm based on Bayesian networks and an algorithm based on decision trees. Two different collaborative filtering databases, one based on movie ratings and one based on Web site interactions, were used to evaluate the prediction performance of the algorithms under study. The performance was measured in terms of mean absolute error (MAE) as well as another measure which focuses on the ordering of the

objects. The results showed that the Pearson prediction was in all experiments one of the best performing, if not best, algorithms. Other authors also showed a superior performance of Person prediction [99, 41]. Therefore, the Pearson algorithm is used as a reference algorithm throughout this thesis.

#### 1.6.1.3 Base Prediction

Sometimes the above Pearson algorithm fails to produce a prediction. This happens when it is not possible to determine the similarity of the target user with other users who have rated the target object, i.e. when $\forall u' \neq u : O_u \cap O_{u'} = \varnothing$. In such cases a backup strategy is necessary to produce a prediction.

The Base algorithm uses the mean rating of all users who have rated for the target object as a prediction.

$$p^{base}(u, o) = \frac{1}{|U_o|} \sum_{u' \in U_o} r_{u',o} \tag{1.1}$$

In its essence this formula also performs collaborative filtering because the ratings of other users are used for the prediction.

The Base prediction is not as sophisticated as the Pearson prediction since it only leads to non-personalized results. However, in some applications it is used as an ad-hoc solution to find popular objects. Examples include

- the online bookseller *Amazon.com* which provides access to best-seller lists (see `http://www.amazon.com/exec/obidos/subst/lists/best/amazon-bestsellers.html`) and
- the community service for financial issues, *The Motley Fool*, with its service for discovering the highest rated messages on their message boards (see `http://boards.fool.com/TopBoards.asp`).

Base prediction (or similar approaches) has long been used before the notion of collaborative filtering appeared. Studies have indicated [92, 41] that newer collaborative filtering approaches, such as Pearson prediction, generally outperform Base prediction. However, because of its broad use for recommending popular objects we use Base prediction in most experiments as a reference algorithm.

### 1.6.2 Experiments

Often off-line experiments are used to simulate the behavior of algorithms in the on-line application. Most of the time the experiments are concerned with evaluating the prediction accuracy of collaborative filtering algorithms or modifications of such. In general the datasets were split into training and test sets. The training set is the part of the dataset which represents the known user preferences, and the test set represents the unknown preferences, which a collaborative filtering system is to predict. To measure the precision of the prediction, the actual ratings in the test set are compared with the predicted ratings.

To ensure that the experiments do not lead to artificial results due to the experimentational constraints, several aspects were considered:

- The data in the test set is **never** used by the prediction algorithms, neither to create models nor to calculate means.
- The splitting of the training and the test set is performed randomly.
- When complexity allows, the experiments are repeated many times with different random seeding, and the precision is averaged over all runs.

#### 1.6.2.1 Datasets

At the beginning of the research for this dissertation in 1998, there was only one large collaborative filtering dataset publicly available, the EachMovie dataset. This lack of datasets led also to the decision to implement an on-line prototype in order to collect a collaborative filtering dataset.

**1.6.2.1.1 EachMovie** The EachMovie dataset was collected by *Digital Research* in an 18-month collaborative filtering movie recommendation study. The dataset is made available to interested parties. It contains ratings of roughly $70,000$ users who gave 2.8 million ratings for $1,600$ different movies. The original dataset contains three tables: users, movies, ratings. The original ratings were on a scale between 0 and 1 with 0.2 intervals leading to 6 different possible ratings.

For our experiments the scale of the ratings was transposed to the scale $\{0, 1, 2, 3, 4, 5, \}$ so that the data could be represented in whole numbers and therefore occupy less memory. The dataset originally contained ratings where

the users did not know the movies but they concluded their rating solely from the title. These ratings (approximately 10%) were removed from the dataset, since the accuracy of these ratings is less certain. For some experiments we reduced the size of this dataset for computational purposes. However, in such cases the reduction is done in an unbiased way, so that the reduced dataset is representative of the whole dataset.

**1.6.2.1.2 Active WebMuseum** The on-line museum, the Active WebMuseum, which provides personalized visits to art paintings, contains roughly $1,300$ paintings which can be rated. Originally the rating scale was from 0 to 10. However, due to some feedback, it was realized that the resolution of the ratings scale was too fine, so that it was difficult for users to provide quick ratings. After several months of service, the rating scale was reduced to five ratings from 0 to 4 { 0=terrible 1=bad 2=neutral 3=good 4=excellent }.

Further to advertisement in related newsgroups and mailing list, some users were attracted to the Web site. The number of ratings keeps increasing as the site is always available. There are currently about $12,000$ ratings by 400 users in the dataset.

### 1.6.2.2 Measurements

In order to evaluate various approaches of collaborative filtering, we divides the rating dataset into test set ($r^{test}$) and training set ($r^{training}$). The training set is used to predict ratings in the test set. The predictions are then compared to the ratings that users provided in the test set using the commonly used error measure:

**Mean Absolute Error (MAE):** The mean absolute error is calculated as follows:

$$\begin{aligned} MAE(u) &= \frac{\sum_{o \in O_u/r^{test}} |r_{u,o} - p(u,o)|}{|O_u/r^{test}|} \\ O_u/r^{test} &= \{o : r_{u,i} \in r^{test}\} \end{aligned}$$

The MAE for several users is then averaged as follows:

$$MAE = \frac{1}{\sum_{u \in U} |O_u/r^{test}|} \sum_{u \in U} MAE(u) \cdot |O_u/r^{test}|$$

We use MAE to measure the performance of a collaborative filtering system, a metric typically used to measure the precision of collaborative filtering

results [99, 97, 15]. Unfortunately, it is difficult to relate this metric to the performance a user would experience while using a collaborative filtering system. Later in Chapter 5 a category-based performance metric is proposed. This metric is more integrated with the recommender system and considers how the collaborative filtering prediction results benefit the user when used in the application. However, the category-based metric is very specific and not generally applicable, i.e. it requires the knowledge of categories of objects. Other commonly used metrics include root squared error, correlation between predictions and ratings, precision and recall, etc. Our experience and related research on performance measurements of collaborative filtering algorithms has shown that the MAE is a good proxy measure, since other measures correlate with the MAE measurements [94].

In summary, we chose mean absolute error (MAE) as a reference metric for assessing the performance of collaborative filtering algorithms for the following reasons:

- First, MAE is a well understood measure used as well by other researchers concerned with collaborative filtering.
- Second, MAE has been shown to correlate with other metrics, thus indicating that it is representative for the performance of collaborative filtering algorithms.
- Finally, contrary to more specialized metrics, for example the ones which are presented in Chapter 5.2.1, MAE can be used to evaluate collaborative filtering algorithms without knowing how the prediction results are applied later in an application.

## 1.7 Organization of Dissertation

This dissertation is organized as follows:

- In Chapter 2 an original collaborative filtering algorithm is presented which uses hierarchical clustering to create a model of the collaborative filtering dataset, a method that produces better predictions in cases of sparsity of preference data.
- In Chapter 3 the prototype user-adapted Web site is described in detail. Further, general observations and guidelines are given for the creation of user-adapted Web sites that rely on collaborative filtering to achieve personalization.

- Then in Chapter 4, algorithms for the combination of collaborative and content-based filtering are studied. Here the focus is object domains such as multimedia for which indexing techniques are less advanced and lead in general to rather weak indexes.
- In Chapter 5 the multi-corridor-access- paradigm is presented which models the use of predicted preferences in a user-adapted Web site and at the same time allows for an evaluation of the personalization quality produced.
- Following, in Chapter 6 strategies are described and evaluated for effectively and efficiently querying new users for preferences.
- Finally in Chapter 7 this dissertation is concluded with a summary of contributions as well as an outlook on important issues that should be addressed in future work.

# Chapter 2

# A Clustering Collaborative Filtering Approach

The performance of collaborative filtering systems depends heavily on the amount of available ratings. Most collaborative filtering algorithms perform poorly when only few ratings are available, i.e. the collaborative filtering database is very sparse. It is reasonable to assume that the choice of algorithm should depend on the amount of ratings available. Particularly for small databases of ratings, the algorithms chosen should make optimal use of the information contained in a collaborative filtering database, while for large databases, algorithms that focus on runtime efficiency are more suitable.

In this chapter we focus on the efficient exploitation of collaborative filtering databases with respect to their problematic states, i.e. the Bootstrap, New-User, and New-Object cases. We present a novel algorithm for collaborative filtering, based on hierarchical clustering, which is aimed at balancing robustness and accuracy of predictions. At the end of this chapter we experimentally show that this clustering algorithm is especially efficient in dealing with very sparse collaborative filtering databases.

## 2.1 Clustering: Motivation

Several algorithms are classically used for collaborative filtering. In particular, algorithms based on Pearson correlation, or variations, are used to perform predictions in several recommender systems and projects [58, 99, 15].

These algorithms build predictions by comparing the rating vectors of users

in order to determine the similarity in ratings of previously rated objects. The prediction is based on a weighted sum of ratings where the weights are determined by the similarity of users.

While this approach has a widespread use, it has some drawbacks in that it does not consider all the information given by the rating database.

### 2.1.1 Transitive Similarity

The Pearson algorithm recognizes users as similar, if the vectors composed of the two users' ratings from objects have a high degree of correlation. In order to calculate this correlation it is necessary that ratings for the same objects by both users be known. However, it is possible that two similar users have not rated the same objects so that the correlation between these users can not be identified and therefore these users are not identified as similar.

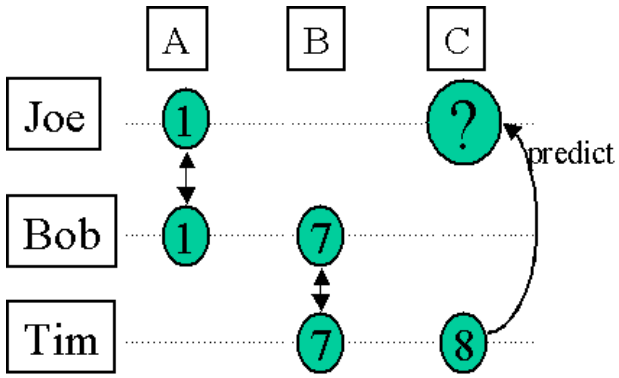



Figure 2.1: Transitive Similarity: The recommender system is to predict how much Joe likes object $C$ using the ratings of two other users Bob and Tim. Joe and Bob rated object $A$ alike and Bob and Tim rated object $B$ alike. Transitive similarity leads to the conclusion that Joe and Tim would also rate similarly on object $C$ and a prediction for Joe could be made. On the other hand, a Pearson algorithm would be unable to make a prediction for Joe since no directly similar user can be identified.

If a third user *3* is similar to two users *1* and *2* it can be assumed that the two users *1* and *2* are more likely to be similar than to be dissimilar. We call the similarity between two users which can only be proven transitively through other users, *transitive similarity*. Figure 2.1 illustrates the use of transitive similarity in an example.

Of course transitive similarity is a vague concept and it does not provide the same certitude about similarity between users as direct correlation of rating vectors. However, when very few ratings are known and it is difficult to base assumption about similarity on factual ratings by users, transitive similarity should be considered to obtain more but less certain information.

### 2.1.2 Object to Object Similarity

In general, collaborative filtering algorithms, especially the Pearson algorithm, form predictions based on similarity of users. Predictions for the target user are inferred from other users' ratings for the target object. In general, no direct conclusions are drawn from the target user's previous ratings for other objects other than to determine similarity with other users. It can be assumed that some objects are similar and therefore receive similar ratings from most users. Similarity or dissimilarity between objects can be determined by the fact that the objects receive similar or dissimilar ratings by the same users. See Figure 2.2 for an example of object-object-similarity.

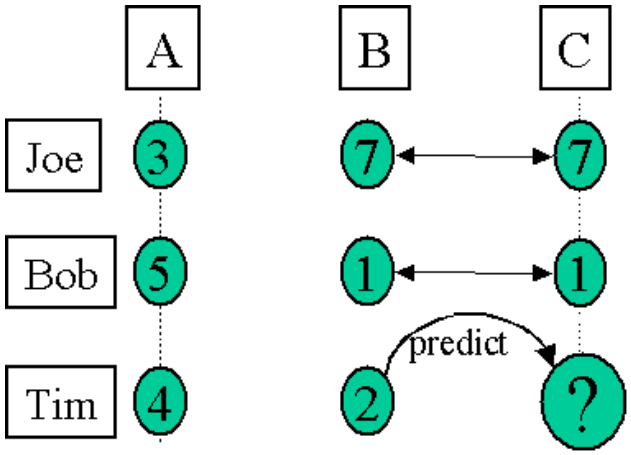



Figure 2.2: Object-to-Object Similarity: The recommender system is to predict how much Tim likes object $C$. Objects $B$ and $C$ were rated the same by Joe. Further Bob gave the same ratings to the objects $B$ and $C$. Object-to-object similarity leads to the conclusion that Tim would also rate object $C$ similarly as object $B$ and his rating for object $B$ can be used for the prediction for his rating of object $C$

Again as with transitive similarity, object-to-object similarity is a very vague concept and should lead to rather uncertain conclusions. However, in cases

of a shortage of ratings, object-to-object similarity should lead to predictions where otherwise no prediction is possible.

### 2.1.3 Using Hierarchical Clustering

We address the sparsity of ratings by a new collaborative filtering algorithm, which takes optimal advantage of the data available. Primarily, transitive and object-to-object similarity are exploited. We propose clustering of rating data to obtain better results for sparse collaborative databases. Our algorithm is based on a hierarchical clustering of users and rated objects. Both user and object rating vectors (users and objects for short) are clustered independently into two cluster hierarchies with the following structural constraints:

- A cluster is either a node or a leaf.
- Each user (respective object) belongs to exactly one leaf of the user (respective object) cluster hierarchy.
- Each node cluster contains exactly two child clusters and therefore all the child clusters' users (respective objects).

Following this definition, the root of the user(respective object) hierarchy contains all users (respective object). The hierarchies are constructed, so that clusters contain similar objects or users, and the degree of similarity increases from the root toward the leaves of the hierarchy.

Row vectors of the rating matrix $score(user, object)$ (the ratings for all objects by a given user) are clustered in the user cluster hierarchy, and column vectors (the ratings to a given object by all users) are clustered in the object cluster hierarchy.

Unknown ratings are predicted by traversing the paths in the hierarchy designated by the target user and target object leading to a smoothed weighted sum (more detail later).

The goal of using the above identified concepts of transitive and object-to-object similarity is achieved as follows by the hierarchical clustering:

- The cluster hierarchy divides users into subgroups(clusters). Each subgroup determines users with rating vectors which are closer to a cluster's center vector than the rating vectors of all the other users which are outside the cluster. Therefore users which are transitively similar are likely to be clustered into the same cluster.

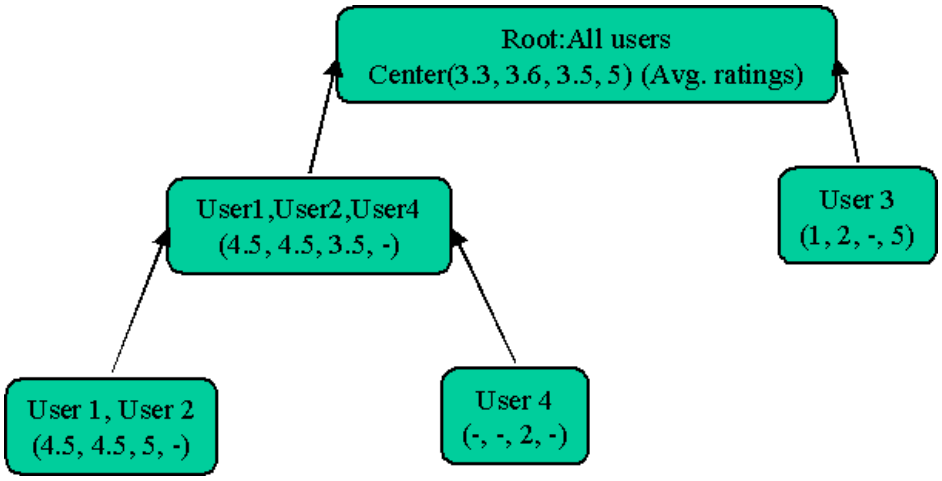



Figure 2.3: Cluster Hierarchy

- Two hierarchies are created: one for user rating vectors and one for object rating vectors. Each hierarchy is exploited for predicting ratings in the same way. Therefore the prediction algorithm takes advantage of object-to-object similarity in the same way as user-to-user similarity.
- In order to avoid the inherent instability of the predictions especially when based on very sparse databases, the predictions are smoothed along the hierarchies. Towards the roots the hierarchies expose information which has more certitude since it is based on more ratings, while towards the leaves the exposed information is less certain but more precise. Basing the predictions on the paths from the leaf to the root leads to more robustness.

In the following section the clustering algorithm is explained in more detail.

## 2.2 Top-Down Strategy

The goal of clustering vectors is to group similar data points. In our case the data points are the rating vectors of users (or in the case of a rated-object cluster the vectors of ratings for an object).

$$u = (u_1, u_2, \ldots, u_n) \quad \text{vector of user ratings for objects } 1, 2, \ldots, n$$
$$o = (o_1, o_2, \ldots, o_m) \quad \text{vector of object ratings by users } 1, 2, \ldots, m$$

The center $c = (c_1, c_2, \ldots, c_n)$ of a cluster $C$ is a vector which is determined by the mean of all vectors contained in cluster $C$.

$$U_i = \{u : r_{u,i} \neq \text{undefined}\} \tag{2.1}$$

$$c_i = \begin{cases} \frac{1}{|C \cap U_i|} \sum_{u \in C \cap U_i} u_i & \text{if } C \cap U_i \neq \emptyset \\ \text{undefined} & \text{else} \end{cases} \tag{2.2}$$

The distance between a vector $u$ and a cluster $C$ is determined by the mean-squared-difference. (When either $u_i$ or $v_i$ are undefined in the following equations, they are replaced with estimated values using *default value inheritance*, a concept which is explained in detail later in this section.)

$I_C$ : The set of indexes of the defined components of the center $c$ of cluster $C$.

$I_u$ : The set of indexes of the defined components of vector $u$, i.e. the known ratings by user $u$.

$$d(u, C) = \frac{\sum_{i \in I_C} (u_i - c_i)^2}{|I_C|} \tag{2.3}$$

Distance between vectors (users or objects) is determined similarly:

$$d(u, v) = \frac{\sum_{i \in I_u \cup I_v} (u_i - v_i)^2}{|I_u \cup I_v|} \tag{2.4}$$

Optimally, the center of a cluster is representative of the vectors it contains. In a good cluster hierarchy the distance between vectors and the associated clusters should be minimized.

The quality of a cluster $C$ is measured in distortion, which is the sum of the distances of all contained vectors from its center.

$|C|$ : Size of cluster $C$ (number contained of vectors)

$$D(C) = \sum_{u \in C} \frac{d(u, C)}{|C|} \quad \text{Distortion of a cluster } C \tag{2.5}$$

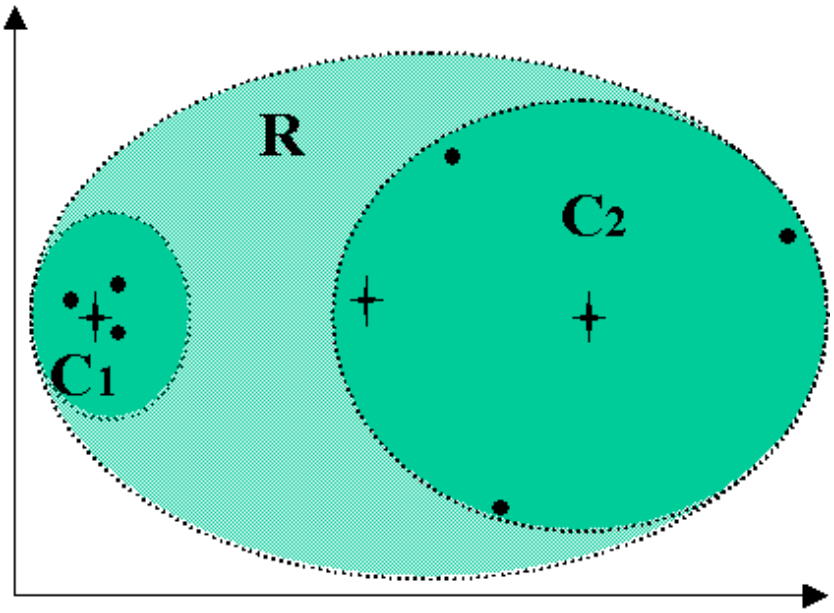



Figure 2.4: Centers in a cluster hierarchy: This figure depicts a root cluster $R$ with two child clusters $C_1$ and $C_2$. The vectors are represented by the dots and the clusters' centers are indicated by crosses. Here it is obvious that the vectors contained in $C_1$ are closer to the center of $C_1$ than the vectors contained in $C_2$ are close to the center of $C_2$. Therefore, the cluster $C_1$ is a better assumption for the location of its vectors than the cluster $C_2$ is for its vectors. Naturally, both clusters are better generalizations for their respective vectors than their parent cluster $R$ is for most vectors.

The goal in constructing a representative cluster hierarchy is to arrange all vectors in a cluster hierarchy so that the distortion for all contained clusters is minimized. For a given distortion, a hierarchy structure is desirable which is as coarse as possible, i.e. the leaves contain as many vectors as possible. The granularity of the hierarchy is controlled by an arbitrarily imposed granularity threshold $t$.

$$\begin{aligned} \mathbf{H} &= \{H : \forall L \in leaves(H) : D(L) < t\} \\ \hat{H} &= \arg\min_{H \in \mathbf{H}} \sum_{C \in H} \mathrm{D}(C) \end{aligned} \tag{2.6}$$

Finding the cluster hierarchy which satisfies the above minimization problem (Equation 2.6) is a hard problem and therefore a heuristic is used to find a sub-optimal solution for the problem.

Cluster hierarchies can be constructed by using either a top-down or a bottom-up approach. The bottom-up approach divides all the vectors into clusters assigning one cluster to each vector. Then find pairs of clusters with minimal distance between the centers and join those clusters to a parent cluster. Continuation of this process would lead to a hierarchy. When using the top-down approach all vectors are initially put into one cluster. Then the cluster is split into two clusters respective to the distortion. The splitting is continued until a desired cluster granularity is achieved.

The bottom-up approach is not feasible in our case due to the enormous computational complexity caused by the size of collaborative filtering databases. Even though this approach might produce the best results in terms of distortion, it is not easily applicable for the domain of collaborative filtering. The amount of users and objects leads to very high dimensional vectors and subsequently to high computational complexity. For example in order to perform one iteration of the bottom-up approach for creating a user hierarchy in the case of the EachMovie database (used for experiments in this chapter) $60,000^2$ distances between vectors with up to $1,600$ components have to be computed. In order to store the distances about 10 GByte of memory would be necessary. Therefore, we decided to construct the hierarchy of clusters using the top-down approach. The steps of our top-down algorithm are listed below:

1. Initially each vector is assigned to the root cluster $R$ of the hierarchy which is initially the only leaf of the hierarchy:

$$\text{leaves}(H) := \{R\}$$

2. Choose a leaf cluster $C \in \text{leaves}(H)$ with a distortion greater than a predefined granularity threshold $\text{D}(C) > t$ which becomes a parent in the hierarchy and is removed from the leaves: leaves := leaves $\setminus \{C\}$
   If no such leaf can be found then the hierarchy has reached its desired granularity and the algorithm ends.

3. Split $C$ into leaves $C_1$ and $C_2$ using the following steps:

   (a) from a random subset of vectors $S \subset C$ the pair of vectors $(\hat{u}, \hat{v})$ with the maximum distance is determined:

   $$(\hat{u}, \hat{v}) = \underset{(u,v)\in(S,S)}{\arg\max}\, d(u, v)$$

   Choosing from a subset reduces the complexity of finding appropriate seed vectors for new child clusters.

   (b) The vectors $\hat{u}$ and $\hat{v}$ are assigned to $C_1$ and $C_2$ respectively. All remaining vectors in $C$ are assigned to the closest cluster of either $C_1$ or $C_2$.

   (c) The centers of $C_1$ and $C_2$ are recalculated.

   (d) When vectors within one child cluster are closer to the other child cluster all these vectors are moved to the other and the algorithm continues with step (c).

   (e) If no more vectors are moved then a suboptimal separation of the vectors into two clusters is found.

4. Add $C_1$ and $C_2$ as new leaves of $H$:

   $$\text{leaves} := \text{leaves} \cup \{C_1, C_2\}$$

5. Continue algorithm at step 2.

We use the mean-squared-difference as the distance between rating vectors and clusters (see Equation 2.1). However, rating vectors are incomplete, since users have not rated all objects, and objects have not been rated by everyone. Default values have to be assumed for the gaps in the rating vectors. Improving upon assuming arbitrary values for all missing components such as the middle of the rating scale or the average rating of all users, our algorithm takes advantage of the cluster hierarchy and uses the center of a common parent cluster as a vector of default values for undefined components in the vectors contained in the child clusters. If the component of the

center of a parent cluster is also not defined, then all vectors within this cluster have not defined this vector component (otherwise the center would be defined for this dimension), i.e. no user in this parent cluster has rated the object corresponding to the missing component, and therefore this vector component can be ignored for the splitting of this parent cluster. We call this mechanism of inheriting default values from direct parents *default value inheritance*. Default value inheritance avoids the assumption of overly general default values for undefined vector components and should therefore lead to better results.

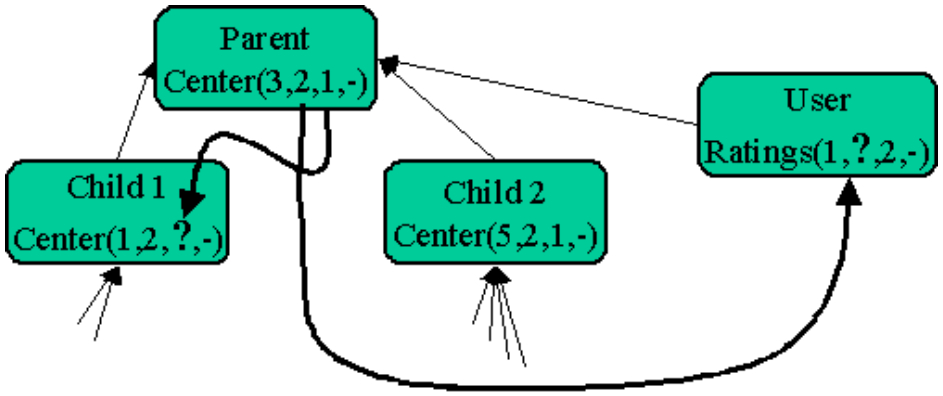



Figure 2.5: Default value inheritance

Figure 2.5 depicts a sample segment of a hierarchy during the clustering. If in this example it should be decided whether *user* has to be assigned to $Child_1$ or $Child_2$, both distances $d(user, Child_1)$ and $d(user, Child_2)$ need to be compared. For the completion of their vectors, *user* inherits 2 as a default for the second component from *parent*, and $Child_2$ inherits 1 as the third component of its center. After assuming the defaults the first three dimensions of $Child_1$, $Child_2$ and *user* are defined and the distances can be determined.

The inheritance of defaults takes advantage of the fact that the neighborhood within a cluster hierarchy contains similar vectors and is therefore more precise than assuming a global default. Consequently less noise is introduced to the calculation of distances.

### 2.2.1 Cluster Prediction

Cluster hierarchies of desired fine granularity can be calculated by using the previously described clustering algorithm. In this section we describe how the cluster hierarchies can be used to obtain predictions. If an unknown rating $\hat{r}_{u,o}$ of a target user $u$ for a target object $o$ should be predicted using the precalculated cluster hierarchies, the following steps are performed:

- Locate the leaf $L_u$ of the user vector hierarchy $H_U$ which contains the rating vector of the user $u$.
- Traverse the path of clusters $P_u$ which is the sequence of all clusters from the root to the leaf $L_u$ of $H_U$ for which the $o$-th component of the center vector $c_o$ is defined.
- These defined centers are used in a weighted sum to predict a rating for this hierarchy.
- The above steps are repeated for the object cluster hierarchy $H_O$.
- Weights in the sums are dependent on the distortion of the corresponding clusters, favoring the clusters with less distortion.
- The combined weighted sum forms the prediction $\hat{r}_{u,o}$.

In a formal way, the prediction algorithm can be expressed as follows:

$$
\begin{aligned}
c_o &: \text{component } o \text{ of center vector of } C \\
H_U &: \text{cluster hierarchy for users } (H_O \text{ for objects}) \\
L_u &: \text{leaf cluster in } H_U \text{ of user } u \\
L_o &: \text{leaf cluster in } H_O \text{ of object } o \\
P_u &: \text{path of all clusters from the root of } H_U \text{ to the leaf } L_u \ (P_o \text{ respectively}) \\
P_{u/o} &= \{C \in P_u | c_o \text{ is defined.}\} \quad \text{respectively for } P_{o/u} \\
w_U(C) &= 1 - \frac{D(C)}{D(H_U)} \quad \text{weights for cluster nodes in } H_U \\
\hat{r}_{u,o} &= \frac{\sum_{C \in P_{u/o}} (c_o * w_U(C)) + \sum_{C \in P_{o/u}} (c_u * w_O(C))}{\sum_{C \in P_{u/o}} w_U(C) + \sum_{C \in P_{o/u}} w_O(C)}
\end{aligned}
\tag{2.7}
$$

The last formula (Equation 2.7) for $\hat{r}_{u,o}$ performs the actual prediction. Summarizing briefly, the prediction is the weighted sum of the defined centers of

all nodes in the cluster hierarchies on the paths from the hierarchy roots to the particular leaves using the respective distortions to derive the weights.

## 2.3 Application and Experimentation

For evaluating our clustering algorithm, we performed simulations on a dataset of movie ratings. We used the data collected by Digital Equipment Research Center from 1995 to 1997 in a collaborative movie recommendation project. The EachMovie dataset contains about 2.8 million ratings of $60,000$ users for $1,600$ movies on a rating scale from 0 to 5. After preprocessing, the dataset contains 2.5 million ratings. For each user, ratings were split into an 80% training set and a 20% test set. From the ratings in the training set, subsets are derived so that desired model parameters are met. Then the suitable subset of the test set is predicted, using various algorithms. The performance of a prediction algorithm is measured in terms of the mean-absolute-error (see section 1.6.2), which measures the difference between the predictions and the ratings in the test set. Even though this measure depends heavily on the rating scale in use and can therefore not be used to compare single results with results of other studies, we found this measure indicative enough for comparing the algorithms in question.

## 2.4 Validation Methodology

In the following our collaborative filtering algorithm, which is based on hierarchal clustering, is validated in the context of sparse collaborative filtering databases. Sparse collaborative filtering databases can be characterized by certain properties such as the number of users and the number of ratings per user. In a first step we define parameters which allow the characterization of collaborative filtering databases. These parameters are then used to artificially derive sparse collaborative filtering databases from larger databases with defined degrees of sparsity. Our algorithm is then compared to other commonly used algorithms for collaborative filtering, Pearson and Base (see section 1.6.1), in prediction experiments. The performance of the algorithms with sparse collaborative filtering databases is compared, measuring the prediction error. Generally speaking, the algorithm that predicts with the lowest error for a given sparse collaborative filtering database should be favored for collaborative filtering systems having databases with a similar degree of sparsity. Therefore, the following experiments indicate general guidelines for the

choice of algorithms for specific degrees of sparsity.

### 2.4.1 Modeling of the Bootstrap Case

When a collaborative filtering application is started for the first time (bootstrapped), it has initially no users but many unrated objects, for which recommendations are required. Eventually, some users will start to add ratings. We captured the number of initial users (modeled as parameter $U$) as a parameter for characterizing a collaborative filtering database. Users, who start using a collaborative filtering application, are confronted with many unrated objects, of which they eventually rate a portion. The number of rated objects increases over time. The amount of ratings (modeled by $R$) that users have provided characterizes the collaborative filtering database. It is important for providers of recommendation services who want to start a new collaborative filtering service to understand the impact of the parameters $U$ and $R$ on the quality of the prediction, so that they can evaluate how much incentive they should give to initial users to build up the collaborative database. These incentives are part of the cost required to set up an efficient collaborative filtering service. This is the reason why in the Bootstrap case, we compare prediction algorithms for low values of $U$ and $R$. In particular, we want to compare strategies where few users provide many ratings each, or many users provide few ratings each.

### 2.4.2 Simulation of the Bootstrap Case

Figure 2.6 depicts measurements of the mean absolute prediction error(MAE) for varying parameters of the Bootstrap case for all algorithms under study.

In this experiment the database is modeled so that it satisfied the varying settings of the parameters for $R$ and $U$. To simulate a database that satisfies settings of the parameters $U$ and $R$, users and ratings are removed or added to the collaborative filtering database. For each measurement the performance of a prediction algorithm is measured in combination with a collaborative filtering database by predicting ratings of a held-out set of test ratings.

It is notable that the Cluster and Pearson algorithms perform equally well for high values of $R$ and $U$. However, the Cluster algorithm converges faster to the maximum performance and is therefore more suitable for sparse collaborative filtering databases than the Pearson algorithm. These graphs also indicate that, for low $R$ and $U$ (very sparse collaborative filtering databases),

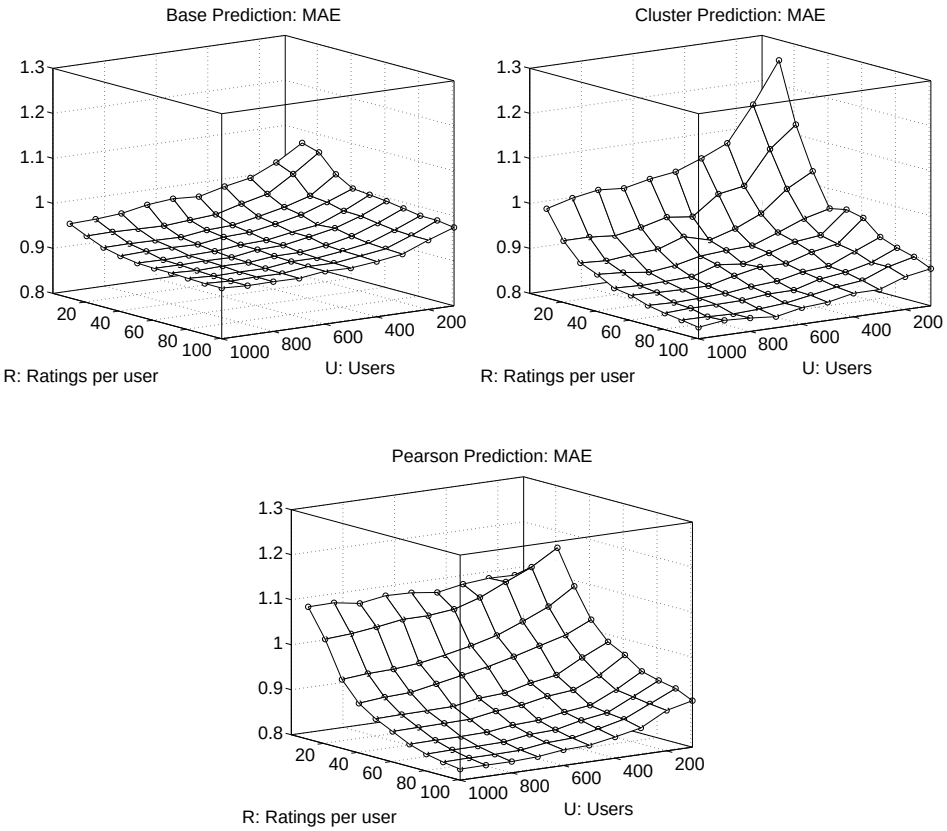



Figure 2.6: Bootstrap Experiment: Collaborative filtering algorithms within the context of the Bootstrap case: For various degrees of sparsity of the collaborative filtering database, modeled through the parameters $R$, ratings per user, and $U$, number of users, the prediction precision of various collaborative filtering algorithms is measured in mean absolute error (MAE).

the Clustering and Pearson algorithms generally perform poorly, even worse than the Base algorithm. This odd behavior can be explained by the following: when users in the database have rated very few ratings, it is very difficult to compare the rating vectors either for finding an appropriate cluster or for calculating the correlation between vectors for the Pearson algorithm. Even when for the test users many ratings are known, the rating vectors can not be easily compared because the vectors can only be compared respective to components which are defined for both. The unreliable comparability of rating vectors leads to large prediction errors for the Pearson and the Cluster algorithm for very low $R$ and $U$. On the other hand, the Base algorithm does not rely on vector comparisons. It is therefore more stable for rating vectors with very few defined components and leads to better prediction precision for very low $U$ and $R$. The odd peak for the Pearson algorithm for $R = 20$ and $U \leq 400$ is due to the fact that the Pearson prediction uses the Base algorithm as a backup strategy in cases when no relationships between the target user and the users in the database can be established using the vector correlation. Such cases are obviously more frequent when the database is very sparse. Concluding from this observation, for very sparse databases, i.e. very fresh Bootstraps, Base prediction should be used to stabilize the predictions. In a later phase more sophisticated algorithms such as Pearson and Clustering should yield better results.

These graphs still do not enable a good comparison of different algorithms for collaborative filtering. The next section performs a cost analysis based on the results of the Bootstrap simulation, which is of more practical interest to determine the best algorithm to use for providers of collaborative filtering services.

### 2.4.3 Cost Analysis of the Bootstrap Case

A provider of a collaborative filtering service needs to initialize(bootstrap) the collaborative filtering database, so that the service can be attractive to users. A low amount of data in the collaborative filtering database leads to poor prediction precision and therefore to an unattractive service. The service provider needs to know how much he has to invest to bootstrap the database in order to provide a desired prediction quality. As previously discussed, collaborative filtering databases can be grown in two dimensions: by increasing the number of users $U$ or by increasing the number of ratings per user $R$. Each dimension involves different costs, e.g. for paying people to rate objects. We identify the cost for one rating as $C_R$, since the effort to

provide ratings with no predictions in return has to be compensated. Also a rater has to be attracted to provide ratings, e.g. by advertisement. We identify the cost to attract a new rater as $C_U$. The total cost of a bootstrap can now be expressed as in the following formula:

$$\text{totalcost}(U, R, C_U, C_R) = U \cdot R \cdot C_R + U \cdot C_U \tag{2.8}$$

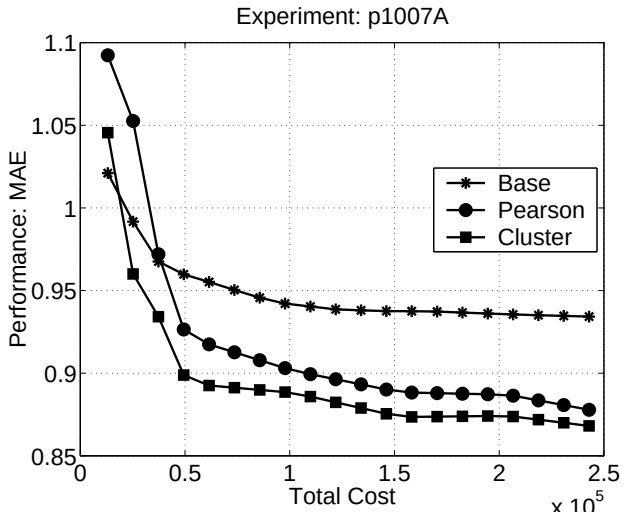
Depending on the settings of the parameters $C_U$ and $C_R$, the total cost of bootstrapping a collaborative filtering database to a sufficiently performing level can vary. As we have shown previously, different collaborative filtering algorithms reach differing levels of performance with the same collaborative filtering databases. Generally two characteristic configurations of the cost parameters can be assumed:

- $C_U = 0$ and $C_R > 0$: Raters can be attracted easily, for example they are users connecting from the Internet after having seen an advertisement, but they need to be compensated for each rating.
- $C_U \gg C_R > 0$: Raters are hard to obtain, for example they are hired by the service provider, but the cost of compensation for each rating is considerably smaller.

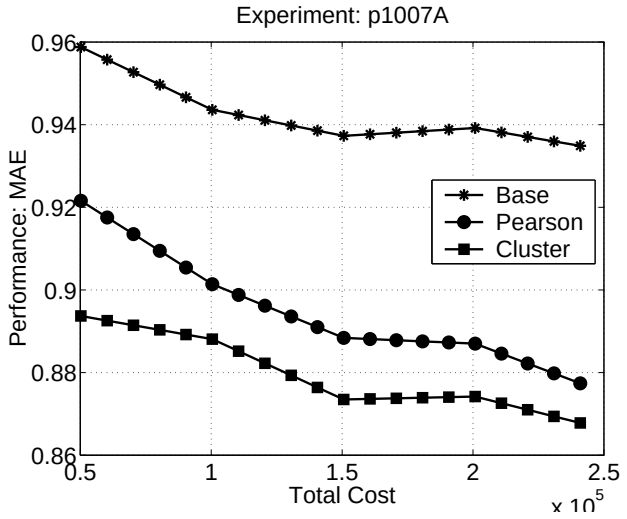
The graphs in Figure 2.7 plot the best performance (in terms of a low mean absolute prediction error) of the algorithms under study as functions of the minimal total cost for bootstrapping a collaborative filtering database using different cost models. In both cost scenarios the Cluster algorithm performs better than the Base and the Pearson algorithms. That means collaborative filtering databases can be bootstrapped more inexpensively if the Cluster algorithm is used. Also it is notable that the curve of the Cluster algorithm in the second pricing model has initially a smaller slope than the other algorithms. This fact indicates that the Cluster algorithm favors the increase of $R$ rather than $U$.

### 2.4.4 The New-User Case

New users in a collaborative filtering application are characterized by a small number of rated objects (parameter $R$). Even though the collaborative filtering database might be sufficiently filled, the prediction performance for new users is expected to be rather poor, since most algorithms rely on the fact that similar users can be detected in the database. If a new user increased




(a) $C_U = 0, C_R = 5$




(b) $C_U = 200, C_R = 1$

Figure 2.7: Minimum mean absolute predictions error (MAE) as a function of minimum Total-Cost of bootstrapping a collaborative filtering database.

his $R$ by rating more objects, then the prediction precision, computed based on his ratings, can be expected to increase.

For investigating the New-User case, we used a fully-grown collaborative filtering database. For some users (new users) ratings were removed from the database. Later the removed ratings were incrementally added to the collaborative filtering database (new users who rate more and more objects). In each step the prediction precision is measured for different algorithms for the new users.

The graphs of Figures 2.8 and 2.9 plot the MAE as an indicator for precision obtained for new users as functions of the number of ratings $R$ of new users for different collaborative filtering algorithms. These measurements indicate the magnitude of prediction errors to be expected for new users who have rated a certain number of objects.

Obviously for small number of initial ratings, the prediction precision is poor for all the algorithms under study. As the plots of Figure2.8 indicate, higher precision can be obtained by using the Cluster algorithm. Especially for lower $R$, the precision difference between the Pearson and Cluster algorithms is notably higher. The difference diminishes with the increase of $R$. This result indicates that for users with few ratings, the Cluster algorithm should be preferred to make predictions. It is also notable that the performance of the Cluster and Pearson algorithms is poorer than the Base algorithm for very low numbers of initial ratings. This can be explained by the failure of the Pearson algorithm to reliably determine correlation between rating vectors when very few ratings are known. Also when very few ratings are known for a user, it is difficult to locate the best suited cluster leaf for the user.

## 2.5 Conclusion

This chapter makes three main contributions:

First, it presents a new approach to collaborative filtering, the Cluster algorithm, which is designed to perform better in the case of sparse collaborative filtering databases. A heuristic capable of clustering large dimensional sparse rating vectors is presented. Further a prediction algorithm is presented. The structure of the cluster hierarchy and the design of the prediction algorithm adapts well to the requirements of new users (and new objects respectively): the clustering algorithm places the users into the hierarchy without assuming more than necessary about unknown ratings by the use of a technique which we call default value inheritance. Then the prediction algorithm smoothes the

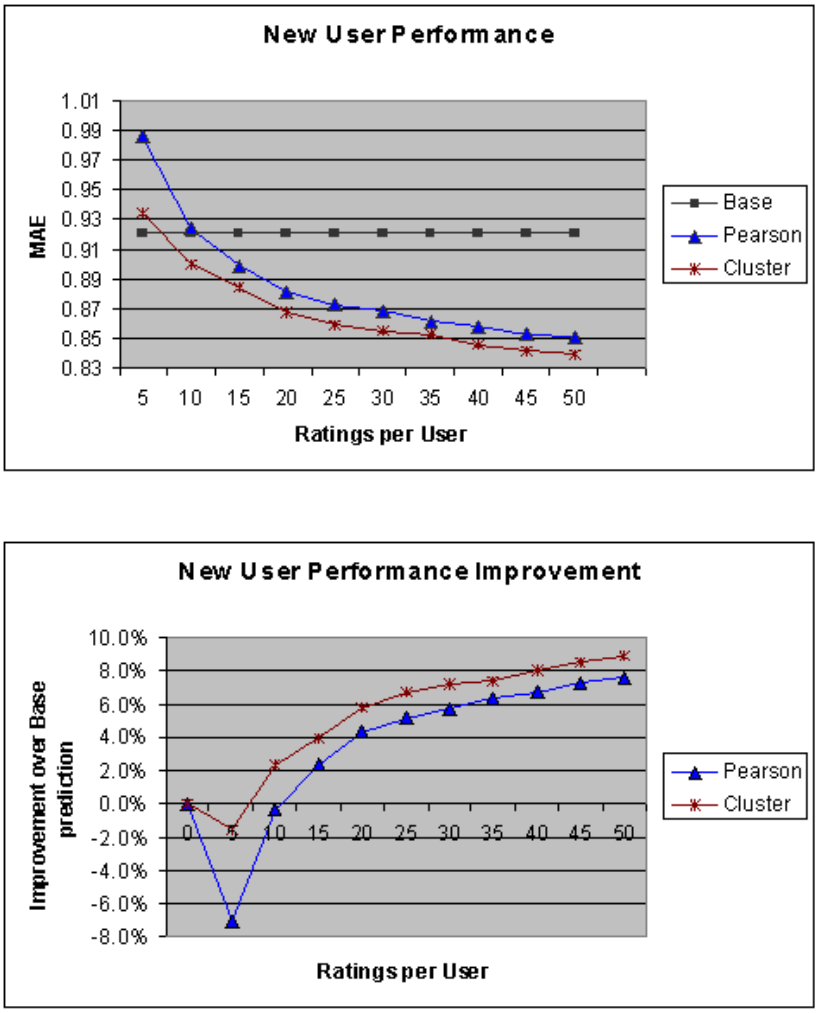
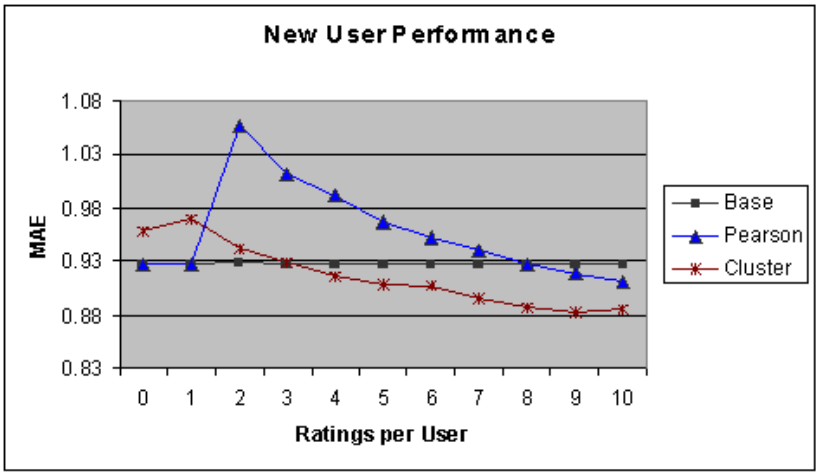



Figure 2.8: Prediction precision of collaborative filtering algorithms in the context of the New-User case. The first plot measures the $MAE$ of the predictions for new users. The second measures the improvement of the $MAE$ in comparison with the Base algorithm. Positive percentages indicate a better performance than the Base.



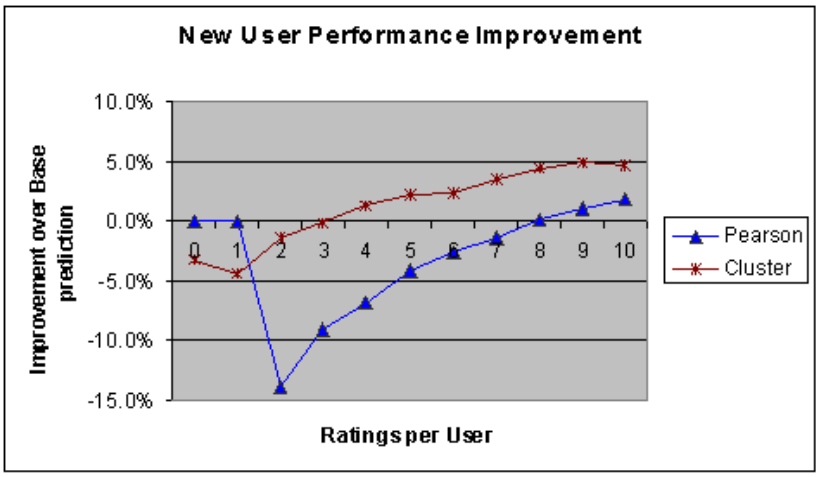




Figure 2.9: Prediction precision of collaborative filtering algorithms in the context of the New-User case studying very low values of initial ratings.

prediction result over a path of clusters. Therefore the proposed algorithm is especially useful for collaborative filtering systems during initialization (or bootstrap), when the objects and users with few ratings are the typical case.

Second, as a framework for validation, two cases of sparse collaborative filtering databases are described: the Bootstrap case and the New-User case are very important issues for providers of collaborative filtering services. Parameters which allow to model these problematic cases for simulations are identified. Further a cost analysis which allows to select the right algorithm for a given cost and performance requirement is presented.

Third and most importantly, the proposed Cluster algorithm is experimentally validated by comparing its performance with classically used algorithms for the described sparsity cases of collaborative filtering databases.

# Chapter 3

# A User-Adapted Web Site: The Active WebMuseum

This thesis relies for the most part on experiments which simulate collaborative filtering problems based on real collaborative filtering datasets. In the beginning of this thesis it was recognized that only very few datasets were publicly available for performing research. At the time only the EachMovie dataset was publicly available for research. This motivated us to implement our own recommender system based on collaborative filtering in order to collect our own dataset. Furthermore, we wanted to draw attention to collaborative filtering as a feasible technology for the personalization of Web sites.

We chose a paintings recommender system as an application for our prototype collaborative filtering system. The reasons for the choice are listed as follows.

- Paintings are attractive content for users in general.
- Paintings can be quickly evaluated (rated) by users without complications, e.g., renting a video or going to a movie.
- Digital images of paintings can be automatically indexed, so that content-based filtering can be applied.

In this section we first motivate the need to transform Web sites into user-adapted Web sites by collaborative filtering and at the same time give a broad definition. Then we present our prototype, the Active WebMuseum, a user-adapted Web site derived from the content of a static Web museum.

## 3.1 A User-Adapted Web Site

The design and creation of Web sites has become an industry by itself. A few years ago the creation of a hypertext Web for presenting products or information was the task of the site's owner. It has been recognized that with growing competitiveness and demand for originality, along with other pressing constraints, the creation of a Web site needs to be performed by professionals in order to present the content in the most efficient way. One way to increase the efficiency of Web sites is by adapting the presentation of the content closer to the audience. This increases efficiency as well as competitiveness and stickiness, as users are attracted by the capabilities of adaptation, and therefore determines the success of a Web site. Since Web pages come into existence when a user clicks on a URL, it is possible to adapt directly to the user specifically instead of to an aggregate audience. While a few years ago the Web was mostly used by the academic world, which as a group required less adaptation, the Web is nowadays open to a broad heterogeneous audience spread over the entire society with highly varying information needs. The differences in the audience often warrants different structures of the information presented in a Web site. Some Web sites anticipate several different groups of users and prepare predefined structures for the corresponding groups, e.g. most corporate Web sites offer structures to let users access as customers, as job seekers, as investors or as other specific groups or subgroups. This approach of pre-clustering follows the prevailing assumption that users can be categorized in a small number of categories according to their interests or information needs. However, it is impossible to anticipate the wishes of every individual user through a set of predefined, usually manually prepared, structures of Web pages. Other Web sites allow users to define their personal profile, e.g. most Web portals use this approach. The *Yahoo!*[1] site, for example offers a variety of free information services to its users. Advertisement banners are displayed in order to pay for the service. The *Yahoo!* Web site allows the users to define a list of services which they would like to use frequently and items in which they are particularly interested through their service *My Yahoo!*. For example users may access the weather report for their home town, and the quotes of their stock, news about their favorite sports through their personalized *My Yahoo!* profile. By storing the user preferences and presenting users with the selected bits of information, Yahoo! provides users with a more efficient way of accessing information, thus gaining in attractiveness and securing an audience for the advertising.

[1] *Yahoo!* `http://www.yahoo.com`

However, this customization technique requires some effort on the part of users as they must define their , user profile before they can benefit by receiving a personalized selection of information. In view of eliminating the need for Web site creators to anticipate possible kinds of visitor-clusters, and for visitors to customize their view to the site by defining their user profile, in this chapter we evaluate an automated approach: a user-adapted Web site that adapts by the use of collaborative filtering.

In general, people explore a Web site differently according to their personal interests or information needs. In doing so they continuously make decisions, e.g. setting bookmarks for content they like and ignoring content they dislike. Mostly, these decisions are related to personal good and bad experiences in finding desired information based on personal preferences. These interactions are not further exploited to enhance the structure of the Web site in order to improve future visits by the same or other users. Collaborative filtering is a promising technology which predicts users' preferences based on other users experiences. Collaborative filtering seems to be an appropriate technology to automatically adapt Web sites by presenting a personalized structure to each user.

However, collaborative filtering technology cannot immediately be applied to Web sites in order to add automatic adaption features. Careful design and integration decisions are required. Collaborative filtering systems are generally used for recommender services, which provide personal recommendations based on ratings provided by the user. So two important tasks can be identified when collaborative filtering systems are used:

- Capturing feedback from the user to be stored in the collaborative filtering database and as a basis for recommendations.
- Adaptation of the presentation of information based on recommendations by the collaborative filtering system.

When meeting these requirements, other constraints might interfere. The requirement of providing explicit feedback might be perceived as inconvenient by the user. Also, if they are not designed carefully, the provisions for providing feedback could interfere with the design of the Web site. Therefore, it is important to allow user feedback in a convenient and non-disturbing way. On the other hand, the Web site needs to create an adapted presentation based on predicted preferences for a particular user. It is necessary to integrate the predictions in presentation elements without disturbing the editorial design of the Web site.

In the next section we describe our prototype user-adapted Web site, the Active WebMuseum. We explain how collaborative filtering is integrated in a Web museum site in view of providing the user with a personal experience, while addressing challenging issues presented above. The particular type of information that is contained in a Web-based museum, namely paintings, makes it simple to define an adaptive presentation of this information.

## 3.2 The Prototype: A Personalized Museum

In an ideal world, a visitor to a museum would enter the building and find in the very first corridor exactly those items which he would find most interesting. Given that real museums serve many people at the same time, it is not feasible to rearrange the collection for individual visitors. Often, real museums offer guided tours that cover a specific theme or address a particular audience,, but having personal guides who show exactly the items of high interest seems to be impractical. When a museum's art collection is presented through the Web, it is at least feasible to rearrange the collection for each individual visitor.

Numerous museum sites already exist on the Web. They present images of works contained in a hypertext structure, so that the navigation within a Web-based museum emulates strolling through the corridors of a real museum. Existing sites are static, which means that the hypertext structure linking the objects has been defined once and for all, and is the same for all users, in the same way that the topology of buildings does not change. In contrast, our Active WebMuseum has a dynamic topology that adapts to the museum visitor's tastes and choices.

Our site *The Active WebMuseum*[2] is based on the collection of art paintings of the *WebMuseum Paris,*[3] a static Web site without adaptation. The collection contains roughly 1,200 paintings by about 170 painters.

### 3.2.1 The Content Model

While filtering techniques allow automated recommendation of what users might prefer, using these techniques introduces new constraints for the design and presentation of the Web site. The content of the site should be

[2]The Active WebMuseum `http://www.eurecom.fr/~kohrs/museum.html`
[3]The WebMuseum Paris `http://metalab.unc.edu/wm/` created by Nicolas Pioch

rearrangeable according to filtering results while still offering attractiveness for users by providing typical features of the Web, interlinked content accessed through self-directed browsing. In our prototype we use art paintings as entities that can be rearranged for the user. At a higher level we use a *Multi-Corridor-Access-Paradigm*. Corridors are ordered containers of paintings with some common characteristics. The user may choose from several corridors. The Active WebMuseum currently supports four types of corridors, each containing paintings grouped according to the following criteria:

- By artist.
- From the same time period.
- By similarity in color.
- By similarity in textures.

The results of the filtering are then reflected in a personalized ordering of the paintings within the corridors.

The user is assumed to behave as follows:

- Choose a corridor.
- Enjoy the paintings in a corridor.
- At any time leave the corridor and choose another one.

Following this assumption, the user's visit to the museum is efficient, if the underlying recommender system's recommendations lead to the user's preferred order of paintings.

The corridors are interlinked. When possible the user may change from one corridor to another one related to the currently viewed painting, so that the user is not forced to follow a predefined structure of corridors. Corridors are created when needed. E.g., when a user visits a corridor with paintings by Van Gogh, he may at any time switch to a related corridor with paintings from the same time-period and from there to paintings by another painter who lived during that time.

By reordering and interconnecting existing corridors, which contain references to pages showing paintings, it is possible to dynamically restructure the museum site in a way adapted to the user. See Figures 3.1, 3.2 and 3.3 for examples.


Figure 3.1: Browsing dynamic corridors: When visiting users have entered a dynamic corridor (in this example a corridor containing paintings by Jackson Pollock), they are presented thumbnail images of the paintings ordered according to their preference. From here visitors may continue in the same corridor until they lose interest, may choose to see more details of the paintings, or may enter a new corridor.

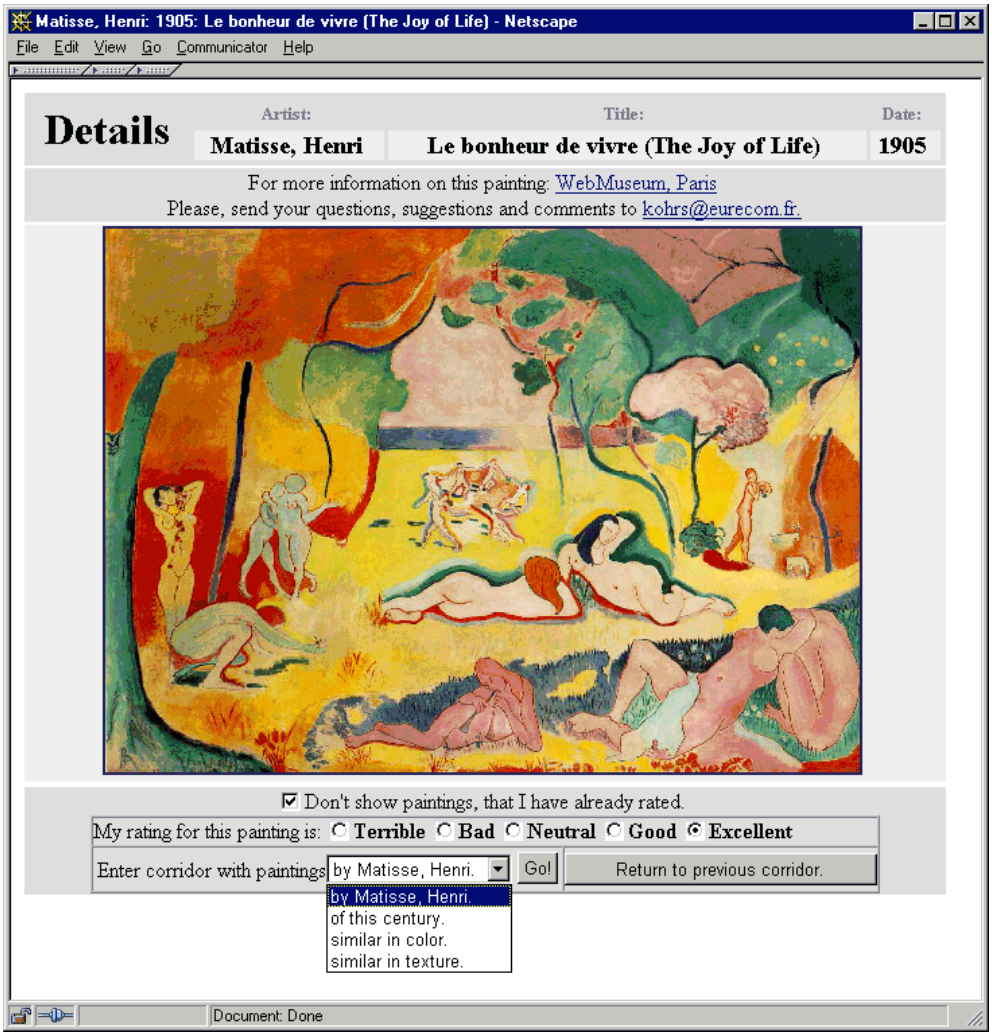



Figure 3.2: An detailed view of a painting: When the user clicks the thumbnail image of a painting from a corridor, the painting is presented in more detail (artist, title, creation date). From here the user may return to the current corridor or enter new corridors related to this painting.

### 3.2.2 User Profiling: Acquiring Preferences

In the previous section we explained how preferences are used in order to transform corridors into personalized corridors. This approach is based on the existence of the visitor's preferences. In the Active WebMuseum visitors can express preference by assigning numerical ratings to paintings (5 values: *excellent*, *good*, *neutral*, *bad*, *terrible*). For paintings which have not been rated by the visitor, the ratings are predicted using the ratings of other users and collaborative filtering technology.

Rating paintings should not be the main occupation of a user. Therefore, the ratings can be conveniently provided while enjoying the museum. In initial trials, we noticed that users are hesitant to give ratings, because giving a rating involves the inconvenience of having to make a decision. Therefore, we ensured in the user-interface that:

- the ratings can be provided with very little effort (one mouse click) without disrupting the user's chosen tour, and
- if a painting is viewed in detail, a rating is mandatory so that visitors must provide a rating before they can continue their tour.

Requesting the user to provide explicit ratings has been the design choice for the Active WebMuseum, since it is the most straight forward method. Later in Section 3.3 feedback collection is discussed more generally.

## 3.3 General Application

Let us step back from our prototype application of a user-adapted Web site and provide more general insight. In this section we summarize some general observations made while designing the Active WebMuseum. In order to create a more attractive Web site, we added adaptability to the user. This should lead to a win-win situation for:

- the user, because the site is more convenient to use (more interesting and less time-consuming);
- and for the Web site provider, because the site becomes more attractive. An attractive Web site leads to better leverage of underlying resources and therefore a more valuable Web site for the owner, e.g.,

the information can be accessed by a larger audience leading to more advertising revenues.

Here, we point out important issues when adding adaptability to a Web site by the use of collaborative filtering. The general scenario can be outlined by the following steps:

- The user visits a site containing Web pages of some sort of information.
- The server offers predefined structures to access the pages.
- The user navigates through the server while providing feedback.
- The server adapts subsequent pages reflecting a structure adapted to the user.

The most important issue is how to integrate preferences, predicted as well as known, about items into the structure of pages. In the case of the Active WebMuseum we use a multi-corridor-access-paradigm [54]: Interconnected corridors containing objects that are ordered according to the preferences generated by the filtering algorithm. Figure 3.3 depicts how the multi-corridor-access-paradigm is applied and implemented in the Active WebMuseum.

This paradigm can be applied in a similar way to other domains and is not exclusively suitable to paintings. For example in the general case, when the corridor is in fact a regular Web page, the links contained in a page could be highlighted according to importance, e.g. the color intensity would depend on importance (to reflect the ordering). Another possibility is to always accompany a page with a box of recommended links.

Another important issue is the acquisition of user feedback (e.g. explicit ratings) so that the underlying recommender system can learn about the user's tastes, upon which recommendations are based. In the case of the Active WebMuseum, we use a little bit of force to motivate the user to provide an explicit rating, by making it mandatory for users to provide a rating on a painting viewed in detail before they can proceed to another corridor.

Two general approaches are possible to attain user feedback:

- Implicit feedback: The user's actions are observed and the feedback is derived on the basis of assumptions on the user's behavior.

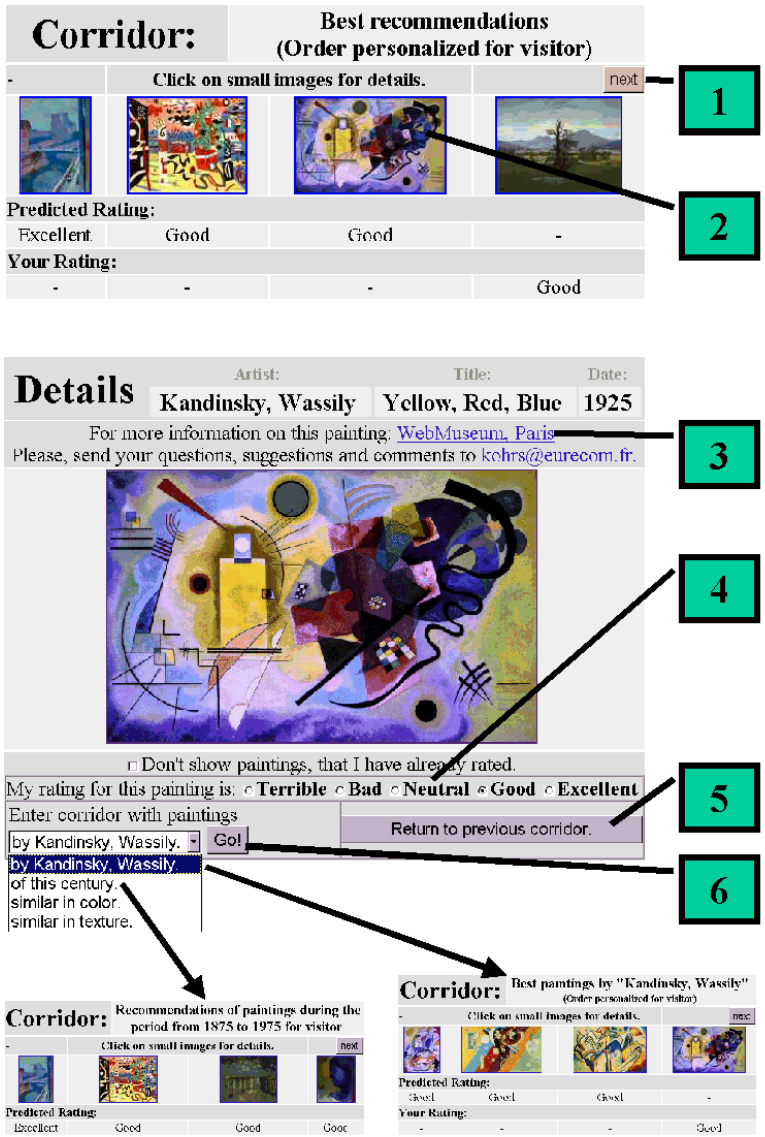



Figure 3.3: Multi-Corridor-Access: As a starting point visitors are presented the beginning of a corridor containing all paintings in the order best suited (predicted) for them. From here visitors may move within the corridor **1**. When they find an interesting painting, they may select it in order to see more details **2**. When viewing a painting in detail, visitors are provided with information about the painting and links to more information **3**. At this point, visitors express their opinion about the painting by selecting the corresponding score **4**. From here visitors may return to the previous corridor **5** or can take another direction by selecting a corridor related to the current painting **6**.

In the GroupLens [58] news article filtering project, implicit feedback was used in terms of measuring the time a user spends reading an article, assuming that users would spend more time with articles they enjoy. Balabanovic [5] proposed a system where the users interactions with an application are used to derive implicit information, e.g., the user bookmarks, files or deletes an object.

- Explicit Feedback: Explicit feedback in collaborative filtering systems is usually provided as numerical scores.

The advantage of implicit feedback lies in the ease for the user. The price for ease is paid in inaccuracy, since the interpretation of the user's actions might be wrong, e.g. when a user views a painting for several minutes, it is uncertain that he was not interrupted by a phone call, so that the conclusion from this observation is flawed with an estimation error.

While explicit feedback is more precise, it might appear unnatural in the context, e.g. some visitors of museums might find it annoying to decide on an appropriate rating for a painting. Furthermore, people might not see the direct benefit of providing feedback. It is the nature of collaborative filtering that user feedback is compensated by a better performance of the collaborative filtering system for other users, but also especially for the rating-providing users themselves. However, users might not always understand the technology and therefore incentives to provide feedback are more promising. Consequently, in the Active WebMuseum, users are reminded of the importance to provide ratings and are prevented from proceeding each time they neglect to provide a score for a painting. Avery [4] argues that voluntary provision of evaluations leads to a suboptimal supply. Therefore, models that compensate users directly for providing ratings are more promising.

A hybrid approach would contain the good elements of both approaches: reliable feedback and little or no disruption of the user's activity. If the user interface allowed more expressiveness, valuable information could be derived, for example by providing alternative *back* buttons in a browser:

- Back, because I don't like this page
- Back, because I like the previous page
- Back, but I like this page
- Back, for no particular reason

Even when the Web site is designed in a way that user feedback can be captured, typical situations may arise when the collaborative filtering performs poorly due to a lack of ratings, especially in the beginning when few users have used the system. In such cases, a combination of two independent technologies, collaborative-filtering and content-based filtering, leads to better prediction performance.

## 3.4 Conclusion

In this chapter, collaborative filtering has been identified as a key technology for the creation of user-adapted Web sites, sites which allow users to access information more efficiently by offering a personalized structure. User-adapted Web sites that apply collaborative filtering adapt better to the user than statically designed Web sites without the cost of having the user define every detail of the adaption, as with sites which require the user to explicitly set up a user profile.

A prototype user-adapted Web site, The Active WebMuseum, has been implemented and made publicly available. While the technical details about the implementation might be useful to undertake similar tasks, they are not described here in favor of focusing on the more difficult design issues for the integration of collaborative filtering. The most important issues of how to obtain user preferences and how predicted preferences can be used for an adapted structure of the Web site are discussed though the example of the Active WebMuseum.

The experience gained through the implementation of the Active WebMuseum as a user-adapted Web site and the unique collaborative filtering dataset obtained during the public trial form the foundation of the research described in the next two chapters.

# Chapter 4

# Improving Collaborative Filtering with Content-based Information

Earlier we identified typical problematic cases for collaborative filtering systems, cases where not enough ratings are available, due to an insufficient number of users and/or to few ratings per user. In these cases, basic collaborative filtering fails.

Conversely, content-based schemes are less sensitive to sparsity of ratings. While the New-User case (and consequently partially the Bootstrap case) remains a problem for content-based recommender systems, the Bootstrap and New-Object cases can be handled better since new objects can be compared to objects that one user has recently rated on the basis of a content-based criteria.

However, pure content-based schemes are only appropriate when objects can be reliably automatically compared, i.e. the content can automatically be indexed or other meta data is available. While it is possible to obtain meta information about movies (actors, director, genre etc), it is hard to automatically index the video stream of a movie with current technology in order to derive the content. Once the content of an object is indexed, it is crucial for content-based filtering to decide as to how much two objects are similar, e.g., in order to recommend the second object to a user if he/she likes the first and the first is similar to the second object.

Comparative studies [2, 1] indicate that content-based filtering performs almost as well as collaborative filtering in the domain of movie recommen-

dations, where the content of the objects (movies) is described in terms of textual attributes obtained from movie databases, e.g. director, genre etc. In other studies content-based filtering and collaborative filtering have been combined in the context of document recommendation in order to overcome inherent disadvantages of both technologies [7, 28]: Advanced techniques for indexing and comparing textual documents, as the vector space model, were borrowed from the "Information Retrieval"community. In the newsgroup article recommendation track of the GroupLens project, weak content-based criteria, such as spelling and article length, is used to fabricate ratings. Our research has a similar focus; in order to improve collaborative filtering, we address key problematic cases which result from lack of ratings. We chose a domain, art paintings, where content-based filtering and indexing techniques are more promising. Combination techniques for content-based and collaborative filtering for art paintings should therefore have similar properties as for other domains where indexing techniques are less advanced.

In the following we motivate the use of content-based filtering in combination with collaborative filtering by first describing the criteria studied and observations made from the users' ratings. Then, we present our approaches of integrating content-based criteria into collaborative filtering in order to improve the prediction performance. The proposed techniques are evaluated in off-line experiments.

## 4.1 Content-Based Filtering

It is reasonable to expect that images with similar content will be almost equally interesting to users. However, defining image content and image similarity is still an open problem. Ongoing research in Multimedia indexing is focusing on two directions:

- either each image is described by a textual caption, and captions are compared using techniques derived from document retrieval;
- or analysis and recognition techniques are applied to the image pixels to automatically extract features which are compared using some distance measure in the feature space.

We focus on the second approach because it can be entirely automated. In our prototype, we have currently implemented two feature-extraction components, color histograms and texture coefficients, derived from the work described by Smith and Chang. [100, 101]

## 4.2 Color Histograms

The original images of the paintings were converted to RGB format, where each pixel is defined by the values (0-255) of the three components red, blue and green. We project these values in the HSV (Hue, Saturation, Value) space, which more accurately models the human perception of colors. The HSV coefficients are quantized to yield 166 different colors. For each image, the histogram of these 166 colors is computed (proportion of pixels with a given quantized color).

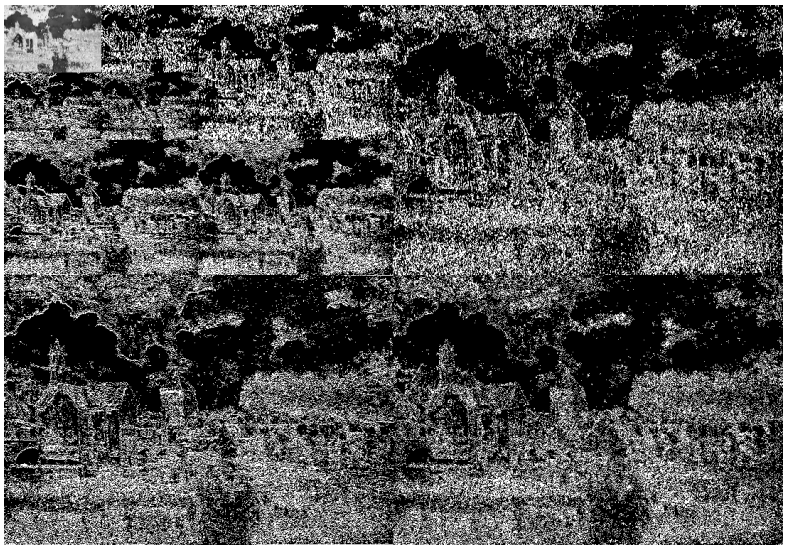
To compare two images, we compute the $L_1$ distance (equation 4.1) between their color histograms:

$$\begin{aligned} h_i(j) &: \text{percentage of number of pixels of painting } i \text{ with the color } j. \\ L_1(h_k, h_l) &= \sum_j |h_k(j) - h_l(j)| \\ d^{color}(p, p') &= L_1(h_p, h_{p'}) \\ d^{color} &\in [0, 2] \end{aligned} \tag{4.1}$$

## 4.3 Texture Coefficients

While color histograms do not take into account the arrangement of pixels, texture coefficients can be computed to characterize local properties of the image. We are using a wavelet decomposition using the Haar transform, by which a number of sub-images corresponding to a frequency decomposition are generated. These sub-images (see Figure 4.1 as an example) are quantized to binary values, so that each pixel of the original image is associated with a binary vector of length 9. The histogram of these vectors (512 dimensions) is the feature vector associated to the texture analysis of the image. As previously for color histograms, the $L_1$(see equation 4.1) distance is used to measure the distance between images.

(a) Original image



(b) Wavelet decomposition

Figure 4.1: In order to determine the similarity between paintings according to texture, all the images of paintings in the database are decomposed into sub images using wavelet decomposition. From the decomposition a feature histogram is derived which can then be compared by the use a vector metric.

## 4.4 Relationship between Ratings and Color and Texture

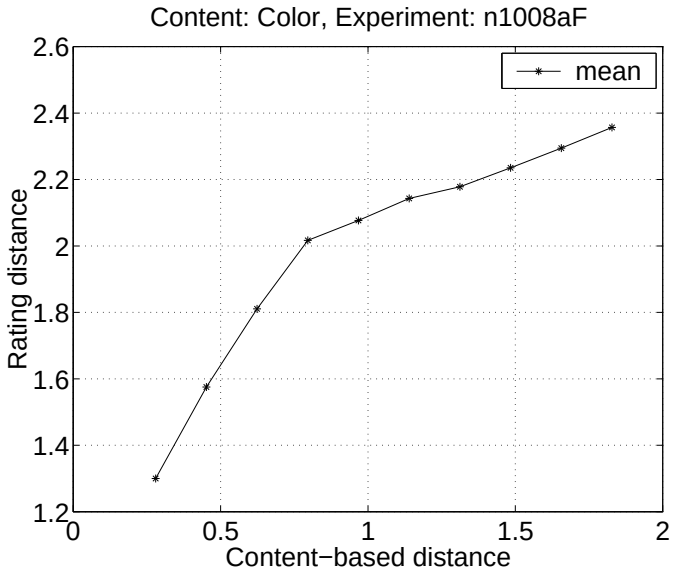
After advertising in several mailing lists related to recommender systems, we attracted initial users to the *Active WebMuseum*. The (ongoing) trial led to a database of approximately 11,500 ratings of 468 users. We use this dataset now to analyze the relationship between the users' rating behavior and the chosen content-based criteria color and texture.

From the rating database we sampled instances when a user rates two different paintings. For each instance we measured the absolute difference between the rating (a score between 0 and 10) and the absolute content-based distance (according to the distance measures, described in sections 4.2 and 4.3). In a second step these instances were clustered into ten equal-length intervals of content-based distances. For each interval the mean absolute rating distance was measured, so that a statistic is approximated as to what absolute rating distance to expect for one user for two paintings with a given content-based distance.

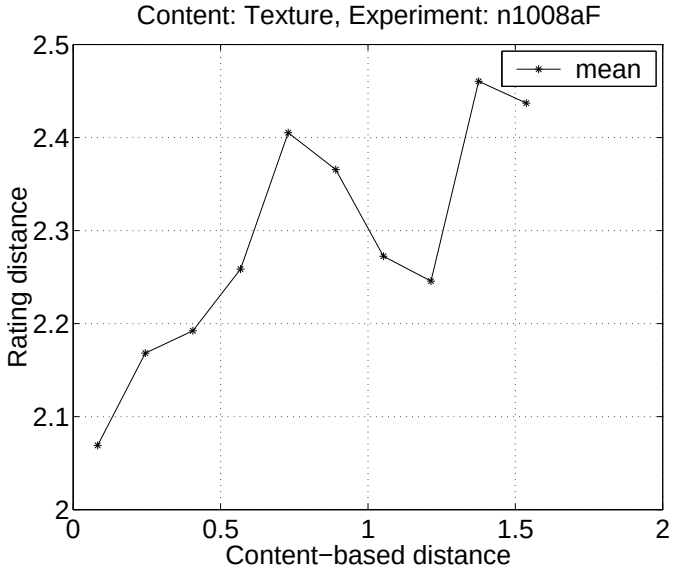
The results of these measurements for the color histogram distance and the texture coefficient distance measure are plotted in figure 4.2. It can be noted that there is a clear positive correlation between color histogram distance and the rating difference, suggesting that paintings which are close in color are more likely to get similar ratings by one user than paintings which are dissimilar in color. The plot for texture is less indicative: while there seems to be a trend towards a positive correlation, there is some fluctuation. Furthermore, it can be noted that the correlation of color is stronger than the correlation of texture in terms of magnitude of rating difference.

It should be noted that the observations above average all the users from the dataset, while it is probable that users vary strongly respective to the correlation between color, texture and ratings, e.g., for some users the correlation might apply less while for others it applies more.

In the next sections we suggest techniques which combine collaborative filtering with content-based measures in order to take advantage of the correlation between color, texture and ratings.




(a) Color-histogram distances




(b) Texture-coefficients distances

Figure 4.2: Correlation of distances between color-histograms(texture-coefficients) of paintings and differences of ratings assigned by users.

## 4.5 Linear Combination with Content-Based Prediction

Based on the findings of the previous measurements concerning the relationship between content-based painting distance and rating difference, we derived a content-based prediction model. We use a linear estimator for the content-based prediction which we model as follows:

$$\begin{aligned} r_{u,i} &: \text{user } u\text{'s rating for painting } i \\ I_u &: \text{Paintings, rated by user } u. \end{aligned}$$

Distance intervals:

$$\begin{aligned} j &= 1..n_\lambda \\ interval_1 &= [0,1); interval_2 = [1,1.5) \ldots \end{aligned}$$

Distance classes:

$$C_j(i) = \{i' \in I_u : d^{color}(i,i') \in interval_j\}$$

Prediction for painting $i$ for user $u$:

$$p^{color}(u,i) = \sum_{j \in 1..n_\lambda} \lambda_j \cdot \frac{\sum_{i' \in C_j(i)} r_{u,i'}}{|C_j(i)|}$$

Expressed in words, the prediction works as follows: if a prediction is to be made for a user $u$ and a target painting $i$, all the paintings previously rated by user $u$ are grouped into distance classes ($C_j(i)$) according to color-based distance to target painting $i$ (see figure 4.3 for an illustration).

Each class is associated with a weight $\lambda_j$, the degree of confidence. The prediction is then the weighted sum of the mean ratings of each class. The weights $\lambda_j$ are estimated through linear regression by using a previously separated subset of the ratings. For each content-based criteria, color histogram and texture coefficients, a predictor was created: $p^{color}$ and $p^{texture}$.

In a second step these content-based predictors are combined with the collaborative filtering predictor $p^{collab}$, as described in Section 5.3.2, linearly using the following formula:

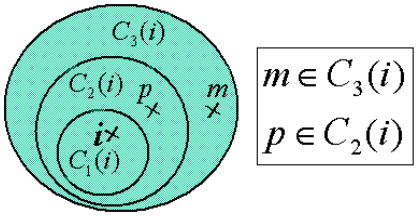



Figure 4.3: All paintings are classified in distance classes according to painting $i$, so that paintings that have a similar distance to painting $i$ fall in the same class.

$$\begin{aligned} p^{comb}(u,i) &= \mu^{collab} \cdot p^{collab}(u,i) + \\ &\quad \mu^{color} \cdot p^{color}(u,i) + \\ &\quad \mu^{texture} \cdot p^{texture}(u,i) \\ \text{with } \sum_k \mu^k &= 1 \end{aligned}$$

The weights $\mu^{\{collab,color,texture\}}$ are estimated by the use of linear regression with a set-aside subset of the ratings, so that the weights are adapted to the relevance of a predictor, e.g., higher for color-based than for texture-based prediction. The estimation of the weights should be repeated as the rating database grows, in order to take into account a precision-gaining collaborative predictor.

As research leads to additional content-analysis tools for paintings, this combination approach allows unlimited inclusion of new content-based predictors. Later in this chapter, we present experimental results for this approach, after presenting a second technique in the following section.

## 4.6 Deriving Artificial Users

While the previous approach combined the collaborative filtering and content-based linearly, assuming that these are independent, we now present another approach which does not change the collaborative prediction algorithm (see section 1.6.1.2) but instead alters the rating database according to the content-based criteria. The rating database is extended with artificial users whose ratings are based on content-based criteria and ratings of real users.

This approach was inspired by Sarwar's *rating-bots* approach [97] for the GroupLens news filtering project: in GroupLens, software agents that which use simple content-based criteria (spelling and article length) to rate news articles automatically are used to increase the number of ratings in the database.

For each described distance metric of Section 4.1 and for each real user $u$, a corresponding artificial user $u^{color}$ and $u^{texture}$ is derived. The artificial users inherit the ratings from the original user $u$, so that if $r_{u,i}$ is defined, then $r_{u^{color},i} = r_{u^{texture},i} = r_{u,i}$. Additionally, artificial ratings are derived for some images, which the original user $u$ had not rated. The artificial ratings are content-based predictions for that particular user. This means that some unrated items are assigned a predicted rating based on similarity between the rated items and the item for which the rating is missing. In order to make a content-based prediction, we define a restricted neighborhood $N_{u,i}$ around a painting $i$ containing paintings $j$ whose rating is defined by the user $u$ and whose content-based distance to painting $i$ is below a threshold $\theta$:

$$N_{u,i}^{color} = \{j \in I_u | d^{color}(i,j) \leq \theta^{color}\}$$

These neighboring paintings are then used to predict a score for the artificial ratings. The prediction formula for color is described below:

$$p^{color}(u,i) = \sum_{j \in N_{u,i}^{color}} \frac{r_{u,j}}{|N_{u,i}^{color}|}$$

In summary the database (or rating matrix) is extended for color as follows:

$$r_{u^{color},i} = \begin{cases} r_{u,i} & \text{if } r_{u,i} \text{ is defined.} \\ p^{color}(u,i) & \text{if } N_{u,i}^{color} \text{ is not empty.} \\ \text{undefined} & \text{else.} \end{cases}$$

A similar process is conducted with the texture distance. Figure 4.4 illustrates the extension technique.

The extended rating database is then used with the collaborative filtering algorithm, which has been described earlier (see Section 1.6.1.2). By extending existing users, the possibility of correlation with the artificial users is

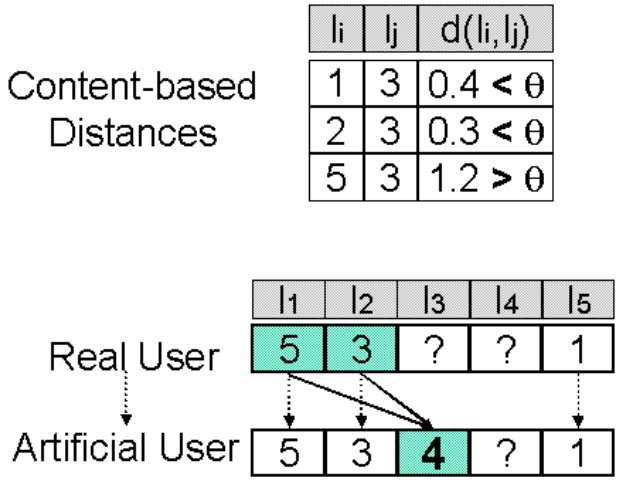



Figure 4.4: Illustration of the extension technique: The artificial user inherits ratings from the real user for the paintings $I_1$, $I_2$ and $I_5$. The rating for $I_3$ is estimated on the basis of ratings for $I_1$ and $I_2$ which are related to $I_3$ according to a content measure (not shown in the illustration).

increased, because the number of commonly rated objects is maximal. In fact, a user $u$ correlates perfectly with its counterparts $u^{color}$ and $u^{texture}$, which causes the content-based prediction to be a strong part of the collaborative predictions for user $u$ and transitively also of all other users similar to user $u$.

## 4.7 Evaluation

We evaluate our approaches for integrating content-based information in the collaborative filtering task in off-line experiments. The experiments are based on rating data which we collected from users who participated in our ongoing public online trial of the Active WebMuseum.

### 4.7.1 The Dataset

While the dataset is still growing, we used a snapshot of the dataset, which contains 11,500 ratings by 468 users for 1,082 paintings.

Figure 4.5 depicts a histogram of the collected user ratings.

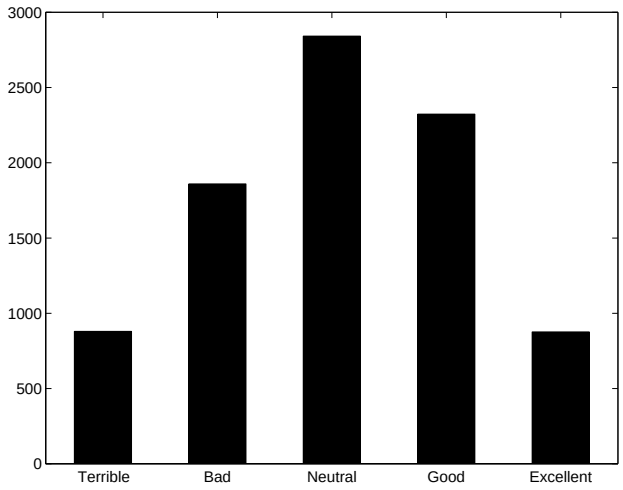



Figure 4.5: Distribution of the user ratings for the paintings in the Active WebMuseum.

### 4.7.2 Measurements

In order to evaluate various approaches of collaborative filtering, we divided the rating dataset in a test set ($r^{test}$) and a training set ($r^{training}$). The training set is used to predict ratings in the test set. The predictions are then compared to the ratings that users provided in the test set using mean absolute prediction error (MAE, see section 1.6.2.2).

The test set contains 50 users for which more than 40 ratings are known. From each user in the test set, 10 ratings are sampled and put aside into the test set; the remaining ratings are used in the training set.

### 4.7.3 Parameter Estimation

Both combination approaches presented use parameters to control the mix of collaborative filtering and content-based filtering. For the linear combination approach the parameters ($\mu^{collab,color,texture}$) determine the weight in the sum of each predictor. The algorithm is designed so that the parameters are adjusted automatically. Through our experiments, we found the following values for these parameters:

$$\mu^{collab} = 0.53, \qquad \mu^{color} = 0.41, \qquad \mu^{texture} = 0.06$$

This shows that the most important component in the linear combination is collaborative filtering closely followed by color-based prediction. Texture-based prediction has only negligible importance.

For the approach with artificial users, threshold values ($\theta^{color}$ and $\theta^{texture}$) are needed in order to determine the neighborhood paintings of a target painting which should determine the artificial rating. The threshold values need to be assigned for the extension algorithm to produce reasonable values. In order to find reasonable values, we searched in the possible range: the previously described dataset was divided twenty times into test set and training set by sampling each time a different subset as the test set. The extension algorithm was then used to predict each test set.

In the graphic: color = Color texture = Texture

The graphs of figure 4.6 plot the MAE as a function of $\theta^{color}$ and $\theta^{texture}$. The experiments suggest that good settings for these parameters are as follows:

$$\theta^{color} = 0.8, \qquad \theta^{texture} = 0.1$$

### 4.7.4 Comparing Extension and Linear Combination

Figure 4.7 shows the histogram of the absolute prediction errors created by using pure collaborative filtering (collab),the linear combination method (combined) and the extension with the artificial users technique (extended). It can be noted that while the collaborative predictor shows more frequent smaller errors, the combined predictor avoids large errors. However, it is hard to judge which one should be better.

Table 4.1 lists the measured precision for the previously discussed predictors. When the combined or the extended technique is used, fewer errors are obtained than with the pure collaborative predictor. The observed improvements in prediction precision through the combination of collaborative filtering and content-based filtering, based on color and texture, suggest that content-based information should be exploited if available. However, the combination models presented do not distinguish between individual users, i.e. for each user the same mix of predictors is used. For some users the content-based measures might be less appropriate than for others. In further developments of the combination models, the differences in sensitivity of different users to content features should be considered.

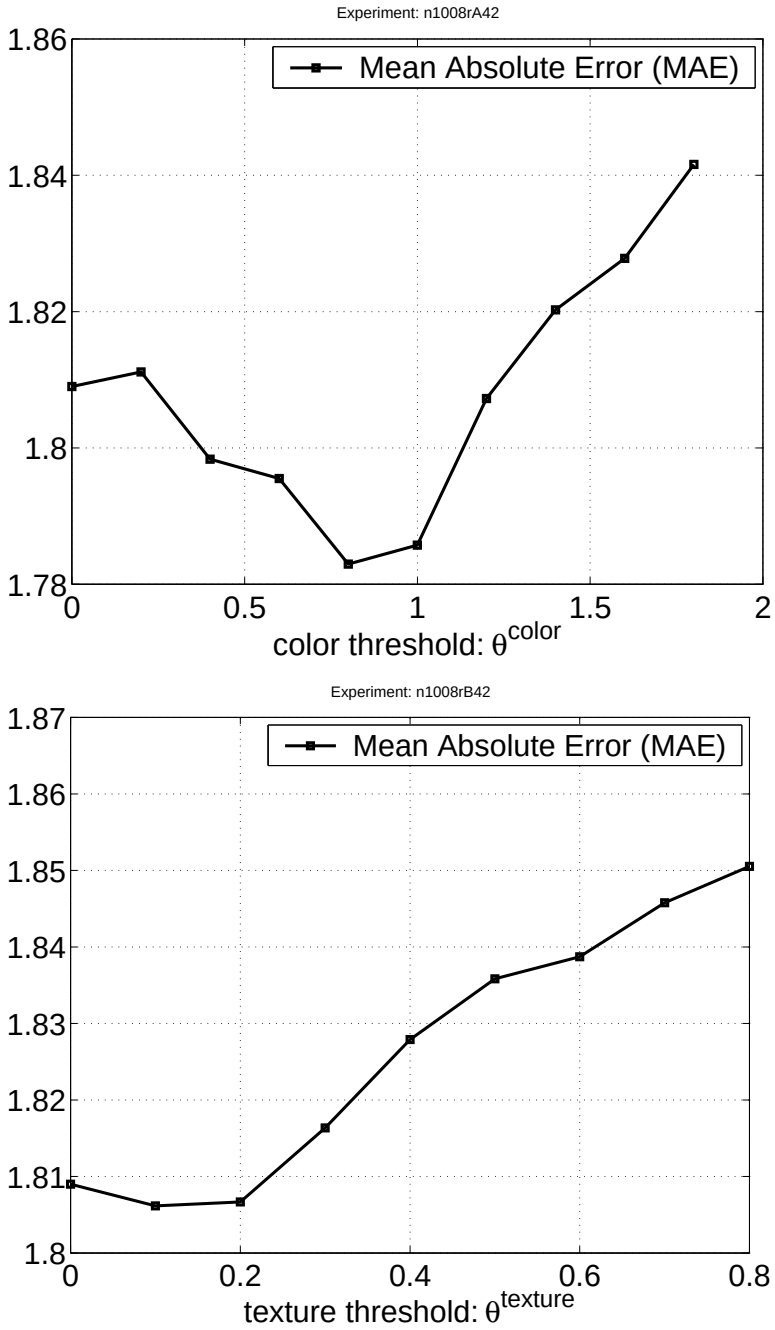



Figure 4.6: Variation of the parameters $\theta^{color}$ and $\theta^{texture}$

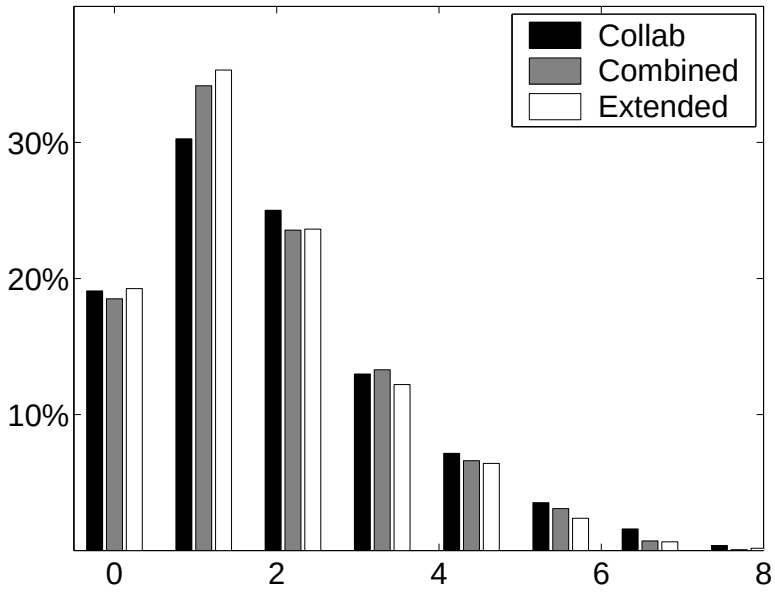



Figure 4.7: Histogram of the absolute prediction errors.

| **Prediction Method** | **MAE** |
|---|---|
| Collab | 1.787 |
| Combined | 1.772 |
| Extended | 1.771 |

Table 4.1: Prediction precision of collaborative, combined and extended predictors: The mean absolute prediction error(MAE) for ratings of the test set was measured. The measured prediction error is lower for the combination approaches (*Combined* and *Extended*) than the error obtained by pure collaborative filtering based on Pearson prediction (*Collab*) which does not use content-based information. This observation, although based on a very limited dataset, suggests that the combination algorithms should be considered in order to improve collaborative filtering.

## 4.8 Conclusion

Collaborative filtering is a key technology for the creation of user-adapted Web sites, sites which allow users to access information more efficiently by exposing a personalized structure of the site. Our key interest lies in the enabling technology. Filtering is a complex problem that can not be addressed by one filtering technology alone. Due to limitations of both collaborative and content-based filtering, it is useful to combine these independent approaches to achieve better filtering results and therefore better user-adapted Web sites. We described how content-based techniques (in this case based on image indexing) can be used to add content-based capabilities to collaborative filtering, and then evaluated our approach through off-line experiments.

This work opens a number of directions for further research:

- Improve the combination with content-based filtering with respect to individual sensitivity to content features.
- Propose evaluation measures that are more related to user satisfaction in particular based on a multi-corridor-access-paradigm.

# Chapter 5

# Multi-Corridor-Access: Using Category-Based Collaborative Filtering

In general the efforts of improving filtering algorithms and using the predictions for the presentation of filtered objects are decoupled. Therefore, common measures (or metrics) for evaluating collaborative filtering (recommender) systems focus mainly on the prediction algorithm. It is hard to relate the classic measurements to actual user satisfaction because the way that the user interacts with the recommendations, determined by their representation, influences the benefits to the user. We propose an abstract access paradigm, which can be applied to the design of filtering systems, and at the same time formalizes the access to filtering results via multi-corridors (based on content-based categories). This leads to new measures which better relate to the user satisfaction. We use these measures to evaluate the use of various kinds of multi-corridors for our prototype user-adapted Web site, the Active WebMuseum.

## 5.1 Introduction

Usually, the performance of filtering algorithms (or prediction algorithms) is evaluated using measures which assess the prediction error. Other possible measures consider the order of all predicted ratings and compare it to the order of ratings that the user would assign.

Both error-assessing and order-assessing performance measures can have rel-

evance to the user but it depends heavily on the application of the prediction results. For example if a system predicts a score of 6 for an object while the user would value the object with a score of 10 then the error is $|10 - 6| = 4$. This high prediction error implies that the recommender system performs badly for this user, because an object which the user values highly is not recommended. On the other hand, if this same prediction was the first of a sequence of prediction for three objects $6 - 5 - 4$ for objects which the user would value $10-7-5$ and an order-assessing measure is used, then the recommender system would be considered perfect because it predicted the correct value ordering for the user. So two different performance measurements lead to two different interpretations about the quality of a recommender system or algorithm. This does not have to be controversial because in the appropriate contexts the interpretations can be correct. It depends on the application and on the task that the user wants to accomplish.

While standard measures are commonly used to evaluate and tune the performance of filtering algorithms, they are hard to relate to the utility of the recommender system and therefore to the user satisfaction. The utility of predictions depends heavily on how the user accesses the recommendations, on available choices during the exploration, and on which choices the user picks. Current measures do not consider arrangements and user choices. Recommender systems which are tuned to the wrong measures might not be optimal for the users.

In this section, we examine how to evaluate the Active WebMuseum from a user perspective. We define a measure based on the multi-corridor-access-paradigm. Multi-corridors are constructed as follows:

- The objects are divided into categories, possibly by using features obtained through automatic indexing.
- For each category the objects are sorted according to the filtering algorithm and arranged in a single *corridor*.

Objects organized in multi-corridors differ from clusters of objects in so far as objects may appear in more than one corridor and that corridors are ordered in contrast to clusters.

The user is assumed to behave as follows:

- Choose a corridor.
- Enjoy the objects of a corridor.

- Exit the corridor when disappointed

This paradigm gives an abstract definition of arrangement and usage of filtering results. This paradigm is obvious and simple. It allows a reliable formalization so that performance measures can be deduced that are based on the interaction between the user and the system and therefore relate more to the user satisfaction. When this paradigm is applied to user-adapted Web sites or other types of recommender systems, more choice for the users can be provided through various multi-corridors and therefore increase performance from the perspective of the user.

## 5.2 Categories

Filtering techniques are used to create predictions of a user's ratings for objects. The simplest way of using these predictions is to provide the user with a ranked list of objects which the user has to follow in best-first order to benefit from the filtering algorithm. While this represents the way that filtering systems are usually evaluated, this access-paradigm is rather awkward for the user and might not be perceived as beneficial. We therefore propose the multi-corridor-access-paradigm, which formalizes the arrangement of filtered objects into corridors according to a categorization scheme. The user may choose a corridor while still benefiting from the predictions of the filtering algorithm.

Here we provide a formal model for multi-corridors. This model leads to two measures, *count* and *score*, which more closely capture the benefit that users get from filtering algorithms within the context of multi-corridor. Then we explore the multi-corridor paradigm by applying the proposed measures to various combinations of categories and prediction algorithms.

### 5.2.1 Multi-Corridor Model and Metrics

When presented as multi-corridors, the objects are grouped according to a categorization scheme. Each category contains objects, which when ordered by a filtering algorithm are presented in a corridor-like fashion. Users choose a corridor and sequentially see as many objects as they like. When they are done with one category, users switch to the next. A performance measure for the whole system should relate to the satisfaction the user experiences. The users' satisfaction is maximized when the system shows them many objects

which they like and few objects which they do not like enough. For our evaluations we use metrics, which capture the user's satisfaction resulting from the combination of a categorization and a prediction algorithm. In our metric it is assumed that users stop using a corridor as soon as they are presented an object that they do not like, i.e. an object which they would rate with a rating below a threshold $t$. The value gained by visiting the corridor is determined in terms of the sum of the ratings which the user would have assigned to the seen objects or simply the number of objects that the user sees.

In the following we provide a framework for assessing the value of a multi-corridors scheme. When assigned to corridors (ordered categories) the objects can be referenced by $o_{c,i}$, which refers to the $i$th objects in the $c$th corridor. The value $\hat{r}_{c,j}$ refers to the predicted rating of $o_{c,j}$. The value $r_{c,j}$ refers to the rating that the user would assign to the object $o_{c,j}$. The absolute prediction error $|\hat{r}_{c,j} - r_{c,j}|$ which is commonly used for assessing the performance of recommender systems is not important for the following considerations. Within a corridor $c$ the objects are ordered according to the predicted rating, so that

$$\hat{r}_{c,j} \geq \hat{r}_{c,j+1}$$

holds for all objects. In our model we assume that users stop using a corridor as soon as they are disappointed when they see an object with a rating ($r_{c,j}$) below a threshold $t$, which is the *neutral* rating. So that

$$stop_c = min\{j : r_{c,j} < t\}$$

is the index of the object in corridor $c$ which causes the user to exit corridor $c$. Until the users see $o_{c,stop_c}$, they see the objects $\{o_{c,1}, \ldots, o_{c,stop-1}\}$. The ratings that the user would assign to those seen objects or the number of objects seen can be assumed to relate to the user's satisfaction gained by visiting this corridor. If the prediction was perfect, users would see all objects which are important to them in this corridor. This leads us to the measures which estimate the percentage of experienced ratings (*score*) and the number of objects seen in one corridor (*count*):

$$score_c = \frac{\sum_{j<stop_c} r_{c,j}}{\sum_{j\in\{l:r_{c,l}\geq t\}} r_{c,j}}$$
$$count_c = \frac{stop_c - 1}{|\{l : r_{c,l} \geq t\}|}$$

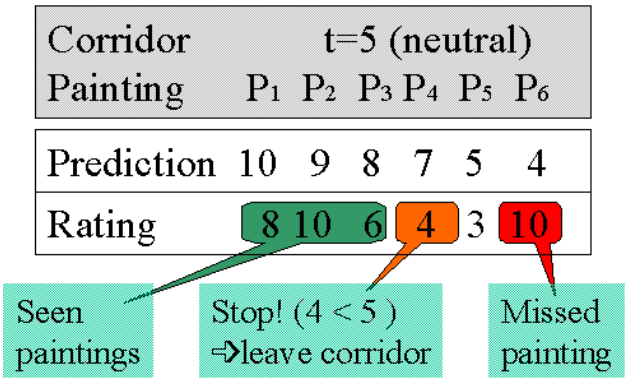



Figure 5.1: When users enter a corridor, they keeps on seeing paintings as long as they like the paintings, i.e. they would rate them higher than $t$. As soon as they are disappointed they leave the corridor and might miss paintings which they would also like.

In a typical visit it can be assumed that the user visits one or more corridors. In order to assess the utility of a multi-corridor for the user, we model the access in a simplified way: the user is expected to choose only one corridor. Each corridor is chosen with a probability $w_c = P(\text{corridor } c \text{ is chosen})$, so that the expected experienced *score* and *count* can be estimated as follows.

$$score = \sum_c w_c \cdot score_c \tag{5.3}$$
$$count = \sum_c w_c \cdot count_c \tag{5.4}$$

The distribution of $w_c$ is assumed to be uniform if not otherwise indicated for the following experiments.

## 5.3 Experiments

To further study the new corridor-based measures, we conducted experiments using a dataset derived from the Active WebMuseum. For the user prediction experiments, we use 8,780 ratings assigned to paintings by 320 users during the ongoing trial of the prototype Active WebMuseum.

### 5.3.1 Experimental Procedure

Various sets of off-line experiments were conducted in order to resolve questions concerning the application of multi-corridors in the Active WebMuseum. For the experiments, the user ratings are split into a test set and a training set.

For 16 users with more than 100 ratings, a subset of 80 ratings is sampled in the test set (1,280 ratings). The remaining ratings are used as a training set (7,500 ratings). In order to eliminate the effects of biased splits, the splitting is repeated 50 times using random samples for the test set. For each split the performance is measured (using the measure proposed in section 5.2.1) and accumulated to a mean performance over all splits and all users.

### 5.3.2 Prediction Algorithms

In order to compare the performance of collaborative filtering, we also use the following filtering algorithms in the experimental context:

**Collab:** This is the standard collaborative filtering algorithm using Pearson correlation.

**Base:** The base algorithm uses the mean rating of all users for an object as a prediction. This is also a collaborative filtering algorithm but it does not produce personalized results and is solely used for comparison and as a backup algorithm in rare cases when collaborative filtering fails.

**Random:** The random algorithm uses uniformly distributed random numbers within the rating range as prediction. This algorithm is only useful for experimental purpose to estimate baseline results.

### 5.3.3 Categorization Datasets

In this study we examine three basically different categorizations:

**Automatic:** In previous work we discovered a correlation between colors of paintings and the users' ratings for these paintings (see Section 4.4). This leads us to assuming that users might benefit from a color-categorized presentation of the paintings. This categorization is based on automatically generated color histograms, which have been clustered in a fixed number of categories using a K-Mean algorithm respective to the color histogram distance defined in Section 4.2. This categorization scheme is referred to as an *automatic* categorization, since no manual work is needed to create the categorization.

**Manual:** The paintings of the Active WebMuseum were extracted from Web pages. These pages usually contained descriptions of the painter and the style. In these descriptions the mentions of typical keywords were counted, which allowed the assumption that the paintings belong to a certain style. By this technique all the paintings ($\geq 1,200$) were categorized into ten different styles, i.e *Baroque, Cubism, Expressionism etc.* This categorization is based on the *manual* work of the authors of the descriptions and their expert knowledge, and could not be automatically generated without the descriptions.

**Random:** As a third categorization we created random categories. The paintings are randomly distributed over all categories. This categorization scheme serves solely as a basis for comparison.

In order to allow comparisons between different categorization schemes, each categorization has the same number of categories ($C_n = 10$) and each painting is assigned to only one category within each categorization.

### 5.3.4 Results

We will now approach certain aspects of the use of multi-corridors in recommender systems. For the following results we use the *score* (see Equation 5.3) measure which is strongly correlated to the *count* measure.

#### 5.3.4.1 Comparison of Categorization Schemes

Table 5.1 compares the performance using the *score* measure for various prediction algorithms in combination with the three kinds of categorization schemes.

| Categorization \ Algorithm | Random | Base | Collab |
|---|---|---|---|
| Random | 0.324 | 0.495 | 0.500 |
| Automatic | 0.379 | 0.525 | 0.528 |
| Manual | **0.406** | **0.555** | **0.558** |

Table 5.1: Performance comparison of categorization schemes and prediction algorithms.

In general the *manual* categorization performs best, as would be expected. However, the *automatic* categorization performs notably better than the random categorization.

Surprisingly in the first column, the random predictor leads to different results for different categorizations. This can be explained by the fact that the categories are correlated with the ratings for the contained paintings (even though that is not intended), e.g. some categories contain more generally good objects so that even a random predictor can succeed. This result implies that the presentation in multi-corridors based on meaningful categorizations relating to style and even the automatic categorization improves the performance of the recommender system in terms of more perceived *score*, especially if the prediction algorithm is weak.

These results indicate that using *base* is better than *random*, and *collaborative* is better than *base* prediction. This is not surprising since *base* takes into account general popularity and *collaborative* prediction personalizes the predictions. However, while generally remarkably outperforming *base* prediction, here *collaborative* only adds little performance over *base* prediction. This can be explained with the limited dataset which is available, i.e. a critical mass of users to have groups of users with differing and common tastes.

#### 5.3.4.2 Category Weighting

Until now, for the *score* measure, we assumed equal probabilities that a user chooses a particular corridor (Equation 5.3), i.e. the distribution of the weights $w_i$ is uniform. In practice that would mean that the user would

blindly choose one of the available corridors. Sometimes it might be reasonable to assume that the user favors one corridor over the others, i.e. because the user usually likes objects in a particular corridor. We model this by adapting the weights $w_i$ to the known ratings that a user had assigned to objects of one corridor. Low weights are assigned to corridors with low average ratings, and high ratings are assigned to corridors with high average ratings. This weighting scheme is referred to as the *popular* weighting scheme, as opposed to the *uniform* weighting scheme.

| Categorization \ Algorithm | Uniform | Popular |
|---|---|---|
| Random | 0.500 | 0.502 |
| Automatic | 0.528 | **0.562** |
| Manual | 0.558 | **0.568** |

Table 5.2: Comparison of uniform and popular weighting of categories.

Table 5.2 compares the results of uniform and popular weighting. Significant improvements can be observed for the *automatic* categorization with the *popular* weighting. For the *manual* categorization the improvement is not as significant. One could argue that the results are improved by changing the measurement. However, since the measurement is in the same quantity related to the user satisfaction, it can be concluded that the user is better served with a weighted categorization, i.e. particular categories are recommended.

## 5.4 Conclusion

The contributions of this chapter are threefold:

First, the multi-corridor-access-paradigm is identified and defined. Second, we provide metrics based on the multi-corridor-access-paradigm which are focused on the performance of information filtering prediction algorithm while at the same time considering the access patterns of the users. Third we apply the metrics found within the context of the Active WebMuseum using prediction algorithms and categorization schemes.

The work presented is only a first step in an important direction of more conscious recommender systems. The multi-corridor-access-paradigm and measures can be further refined. Certainly, a measure is needed for online experiments which can then lead to highly useful tools for observing

the quality of a recommender system at run-time. For example when an on-line measure detects that a current user experiences bad performance, a system operator (maybe in the form of an intelligent agent) could intervene and provide some incentive for the user to stay. Further, the integration of categorization and prediction algorithms need to be explored with the hope that prediction results in general improve and become more reliable.

# Chapter 6

# Smart Object Selection for New Users

An important issue of collaborative filtering lies in the dependence of a filtering system's performance on the number of ratings available from users. Lack of data leads to a family of problems commonly known as sparsity issues. One particular issue is the *New-User-Case*: When a new user, one that has not used the system before, uses a collaborative filtering system, the system is unable to produce personalized recommendations since the relationship between other users of the collaborative filtering system cannot be determined. For such cases the collaborative filtering system has to either rely on an alternative fall-back algorithm, e.g. using average ratings of the whole population of users, or require the new user to first rate some objects in order to use the capabilities of collaborative filtering. In such situations there is a trade-off in how much effort users spend in rating objects before they receive any benefits and the gain in precision of predictions of collaborative filtering due to more information about the user.

In this chapter we study this trade-off, and further provide some solutions for making the pre-rating procedure more efficient in terms of either having to rate less objects or getting better precision out of the collaborative filtering system with the same amount of effort. We propose methods for smarter selection of objects which should be rated by new users.

The proposed methods are validated in the context of our Active WebMuseum project. Data collected in the ongoing trial is used in this research to validate new algorithms in off-line experiments. To further support our results, we also conducted experiments using the movie rating dataset of the EachMovie collaborative filtering project.

In this chapter we first describe the selection of objects as a general problem for collaborative filtering. We then describe our three approaches, variance, entropy and structure selection, in more detail, followed by results from experiments which validate our ideas. Finally, we discuss the general applicability of smart object selection in collaborative filtering systems and conclude this chapter.

## 6.1 Selecting Objects to be Rated by New Users

When a user uses a collaborative filtering system for the first time and very few or no objects have been rated by the user, the collaborative filtering system can only perform poorly as compared to the case when many objects have been rated by the user. In [51] the lack of performance of collaborative filtering for users with few ratings is identified as the *New-User-Case*. New users of a collaborative filtering system need to provide some ratings first to obtain personalized results from the collaborative filtering system. Pennock [83] discusses value of information (VOI) analysis in the context of collaborative filtering, in order to more cost-effectively inquire about objects. However, a proposal as to how to proceed is not yet suggested. A related research field is active learning. The scope of active learning is to minimize the amount of labeled samples in the training data for classification problems. In general, active learning is applied in cases when labeling data is expensive [47, 22, 14].

We added the option of rating a random sequence of paintings in a batch to the core functionality of the *Active WebMuseum* (see Figure 3.2 for an illustration of the rating process). New users are asked to use this functionality to *train* the system, and therefore obtain personalized results in the subsequent visit to the museum. While this is a necessity, it is however very annoying for visitors of the Active WebMuseum who want to enjoy a personalized dynamic tour focusing on preferred paintings.

Asking for the user to rate random objects is probably not the best way to train a collaborative filtering system. In the following we focus on identifying the most promising objects to be rated in the training phase of new users in contrast to just choosing random sequences of objects. The goals are to require fewer training ratings from new users and to provide better prediction performance for the same training effort.

In order to improve upon random selection of objects to rate, we try to

prioritize objects according to the amount of precision improvement a user would get by rating the object. This is of course impossible to know before the user actually rates the object.

We use statistical analysis to find promising candidates of objects to be rated by new users.

The task of object selection is to identify the size-limited set $S$ of objects from a given set of potential objects $P$ which when rated will maximize the future prediction performance of the collaborative filtering system for a target set of objects $T$. The problem can be generalized as follows:

$$\hat{S} := \arg\max_{S \subset P}(\text{performance}\,(p_{/S}(T)))$$

In the following sections, we describe three approaches for prioritizing objects which lead to selecting subsets of objects to be rated by new users: *variance*, *entropy* and *structure*.

### 6.1.1 Variance and Entropy Selection

The first two approaches are based on statistics of the ratings given by other users for the objects in the dataset. The idea is to order all potential objects respective to a statistic taken from their rating distribution.

#### 6.1.1.1 Variance

The first approach measures the variance of all ratings that each object received.

$$\text{variance}(o) = \frac{\sum_{u \in U_o} (r_{u,o} - \overline{r_o})^2}{|U_o|}$$

The term $U_o$ refers to the set of indexes of all users who have rated the object $o$ so far:

$$U_o = \{u : r_{u,o} \quad \text{is defined}\}$$

The term $\overline{r_o}$ indicates the mean rating assigned to this object:

$$\overline{r_o} = \sum_{u \in U_o} \frac{r_{u,o}}{|U_o|}$$

The objects are selected for the set $\hat{S}$ from the potential set $P$ satisfying the following condition:

$$\forall s \in \hat{S}, r \in P \backslash \hat{S} : \text{variance}(s) \leq \text{variance}(r)$$

High variance is considered as noise, so we assume that objects with a small variance are more characteristic.

#### 6.1.1.2 Entropy

The second approach is aimed at answering the question: Which object, when a user's rating for it is known, best reveals the user's identity? In order to answer this question we consider the random variable $U$ as the identity of a user and the random variable $R(o)$ for the observed rating of a user. Now the

$$\text{entropy}(o) = H(U|R(o))$$

of the random variable $U$ under the constraint that $R(o)$ is known can be calculated as follows:

$$H(U|R(o)) = -\sum_{u,r} p(U = u, R(o) = r) \cdot \log(p(U = u|R(o) = r))$$

Here, $p(U = u, R(o) = r)$ is the probability of the event that user $u$ rates object $o$ with $r$ and $p(U = u|R(o) = r)$ is the probability that the rating user is the user $u$ under the condition that the score $r$ is observed as rating for object $o$. The objects are selected in increasing order.

$$\forall s \in \hat{S}, r \in P \backslash \hat{S} : \text{entropy}(s) \leq \text{entropy}(r)$$

High entropy implies that little information is known about a user, therefore objects are preferred which yield the most information about the users, i.e. the objects with a low entropy.

#### 6.1.1.3 Removing Correlation

When several objects are selected at the same time, it might occur that similar objects are selected with strongly correlated rating vectors and therefore might not provide as good a selection as if less correlated objects were chosen. We propose an optimization step which removes the most correlated object of a selected set $\hat{S}$ and replaces it with the least correlated object of potential objects $P$:

$$\begin{aligned} c &= \arg\max_{r \in \hat{S}} (\sum_{s \in \hat{S}} \mathrm{corr}(r, s)) \\ n &= \arg\min_{p \in P} (\sum_{s \in \hat{S}} \mathrm{corr}(p, s)) \\ \hat{S} &:= (\hat{S} \backslash \{c\}) \cup \{n\} \\ P &:= (P \backslash \{n\}) \cup \{c\} \end{aligned}$$

The optimization step may be repeated a variable number of times.

### 6.1.2 Structure Selection

Our next approach is inspired by research conducted in the direction of Latent Semantic Indexing (LSI) for Information Retrieval and, further, the application of Singular Value Decomposition (SVD) for collaborative filtering. When SVD is applied to collaborative filtering, the rating matrix is in general transformed in a lower dimensional space, i.e. reducing the matrix to the most important directions [94, 11, 87]. In our approach for finding best starting sequences we try to identify the objects which are a good approximation of the most important directions. Since our approach is aimed at identifying those objects for which the rating vectors depict most closely the structure of the rating matrix, it is referred to as the *structure* approach.

Ratings for collaborative filtering can be represented as a matrix $r$ which is spawned by the objects and users dimensions. The vector

$$\overrightarrow{o} = (r_{1,o}, r_{2,o}, \ldots, r_{n,o})$$

stands for the object rating vector of the $o$-th object. In the following the structure selection algorithm is described in detail:

1. Initialization:
   $S_0 = \{\},\ P_0 \subset O,$
   $$\forall (u,o) \in U \times O : o_u := \begin{cases} r_{u,o} & \text{if } r_{u,o} \text{ defined} \\ \overline{r_u} & \text{else} \end{cases}$$

2. Centering of vectors:
   $\forall (u,o) \in U \times O : o_u := o_u - r_{u,o}$

3. Selection:
   $$\begin{aligned} s &:= \arg\max_{p \in P_i} \sum_{o \in O} (\overrightarrow{p} \cdot \overrightarrow{o})^2 \\ S_{i+1} &:= S_i \cup \{s\} \\ P_{i+1} &:= P_i \setminus \{s\} \end{aligned}$$

4. Projection:
   $$\forall o \in O : \overrightarrow{o} := \overrightarrow{o} - \frac{\overrightarrow{o} \cdot \overrightarrow{s}}{||\overrightarrow{s}||^2} \cdot \overrightarrow{s}$$

5. Continue with step 3 until the required number of objects is selected.

In the first step the set of selected objects is initialized, a subset of objects is determined to be used as potential objects and the object vectors are initialized with known user ratings. In the second step, the object vectors are centered using the mean user ratings. (We also experimented with the mean object rating and the middle of the rating scale for centering but mean user rating produced the best results.) By using the mean user ratings to center, all vector components that refer to undefined ratings are set to zero. In the third step an object is selected from the set of potential objects by choosing the object for which the rating vector is most aligned with all other objects. The fourth step projects all vectors on the hyperplane orthogonal to the selected vector so that the direction determined by the chosen vector $s$ becomes irrelevant for further selections. The goal is to avoid that in subsequent iterations objects are chosen for which the rating vectors are highly correlated with rating vectors of previously chosen objects. Steps 3 and 4 are repeated until a desired number of objects is selected.

### 6.1.3 Base and Upper-Bound Estimation

We use *random selection* of objects as the base-line approach, i.e. as a lower boundary for performance. It is desirable to also estimate a boundary for the best performance of a selection scheme, i.e. a boundary of performance which is beyond the achievable maximum of a perfect selection algorithm. In order to estimate the upper boundary of a selection algorithm, we designed the following heuristic:

1. A random subset of objects is selected from the set of potential objects.
2. The object which, when removed, least decreases the performance of the selection is replaced by the object of the potential objects which most increases the performance of the selected set. The performance is measured by predicting some known held-out ratings.
3. Step 2 is repeated until no more performance improvements on a second held-out set can be observed.
4. The resulting set is assumed to have a close to optimal performance.

The estimation of the optimal performance boundary is based on finding a suboptimal-performing set by using a greedy algorithm. By repeating the steps with a different randomization, the estimation of the performance can be refined. As this procedure is allowed to look at the test data on which performance is computed, it surely can provide good results. However, it is not certain that there exists another procedure which would lead to similar performance without looking at the test data. Therefore, this performance is an upper-bound of the admissible training method. In the following we refer to the performance of the upper-bound estimation algorithm as the *upper boundary*.

In the next section we present results from experiments to study the performance of the above previously presented algorithms.

## 6.2 Experiments

In order to validate the effectiveness of the object sequence selection algorithms of Section 6, we conducted a series of off-line experiments. The relative improvements of the selection methods presented is shown in comparison with the base algorithm, which uses random selection to choose objects.

### 6.2.1 Datasets and Methodology

One of our target applications is the *Active WebMuseum* and we therefore use the limited dataset of ratings collected during the ongoing public trial. We further support our results by using the larger dataset of the collaborative filtering project *EachMovie*. Table 6.1 lists the dimensions for the datasets in use for the following experiments. The datasets have been reduced from their original size in order to remove users and objects with only few ratings.

| **Dataset** | **users** | **objects** | **ratings** |
|---|---|---|---|
| Active WebMuseum | 468 | 1116 | 11500 |
| EachMovie | 1315 | 408 | 70047 |

Table 6.1: Dimensions of the datasets

Using the same procedure for each dataset, for 30 randomly selected target users (selected from those users which had at least 60 ratings), the ratings were divided into a target set of 10 ratings, to be predicted by collaborative filtering, and a potential training set. From the potential training set, a subset of objects is selected as a training sequence using one of the algorithms described in the previous section. Then the target set is predicted using the remaining ratings database in conjunction with the selected training sequence. The performance of a selected training sequence is measured by the prediction precision, the mean absolute prediction error (MAE, see section 1.6.2.2).

The sampling of target and potential object sets is repeated several times for each user, using each time a different initialization for the random process. The MAE is averaged over all repetitions. The repetition takes place in order to avoid random effects by the choice of a target set. We saw convergent results for two series of experiments based on the same datasets and only differing in the random number generator initializations.

It must be noted that this experimental setup does not fully match the real world application. In the real world application the objects can be chosen from all available objects in the dataset, i.e. 1,200 paintings, while in the experimental setup the choice of objects is limited to a potential training set (as low as 50 objects in these experiments), so that improvements in the real world application are expected to be better than the improvements proven by the results of the experiments. Experiments which match real online conditions are difficult to conduct since it would require a dataset which has complete user profiles, i.e. the test users' ratings for all objects need to be

known.

### 6.2.2 Comparing the Selection Algorithms

The measurement comparisons for the *Active WebMuseum* dataset and for the EachMovie dataset are shown in Figure 6.1.

Here the prediction precision in terms of MAE of our three strategies are compared with the precision of random selection and upper bound. Obviously, the MAE decreases (improves) for increasing number of training ratings. As compared to the upper bound selection our algorithms are relatively close to random selection. This could be a consequence of an overly optimistic upper bound. However, the very distant upper bound further supports the validity of this chapter's intention, by suggesting that significant performance improvements can be obtained by carefully selecting objects to be rated by new users.

Our goal in finding an algorithm for systematic object selection was to improve upon the random selection. Therefore we will normalize our representations of the results relatively to the base of random selection. This makes it possible to better recognize improvements or degradations.

The graphs in Figure 6.2 compare variance, entropy and structure selection with random selection. For the EachMovie dataset it can be observed that entropy and structure selection consistently outperform random selection, while variance selection consistently under-performs the random selection. For the Active WebMuseum dataset the results are mixed. Here only structure selection consistently outperforms random selection, and entropy starts outperforming for users with eight or more training ratings. The variance selection initially outperforms but then later, for users with an increasing number of training ratings, degrades in advantage over random selection. Judging from these results it can be concluded that structure selection is a good means of systematically outperforming random selection for both the EachMovie and the Active WebMuseum applications, and probably also for other collaborative filtering applications. Entropy selection is the second choice candidate. However, variance selection seems unreliable for systematically outperforming random selection.

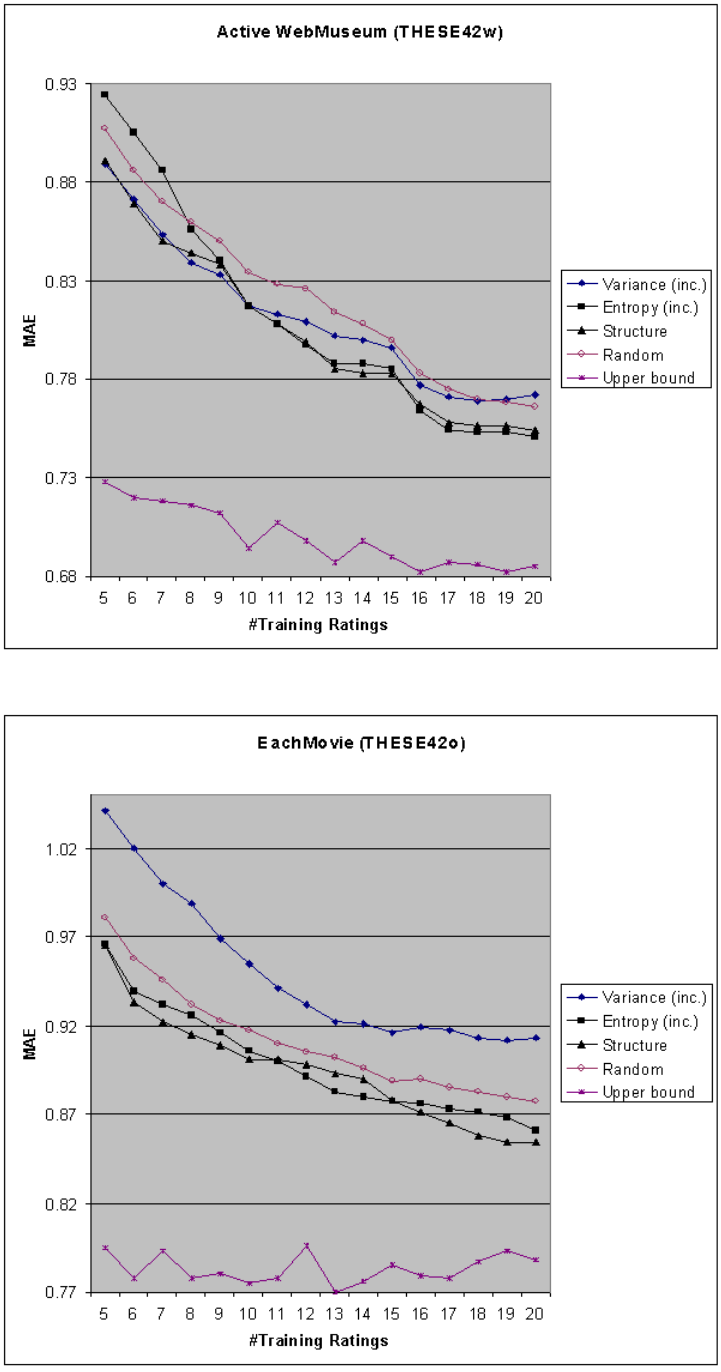



Figure 6.1: Object selection algorithms in comparison with random selection and upper bound. A lower mean absolute error (MAE) indicates a higher prediction precision for the target users. The number of training ratings indicates how many ratings are known for the target users.

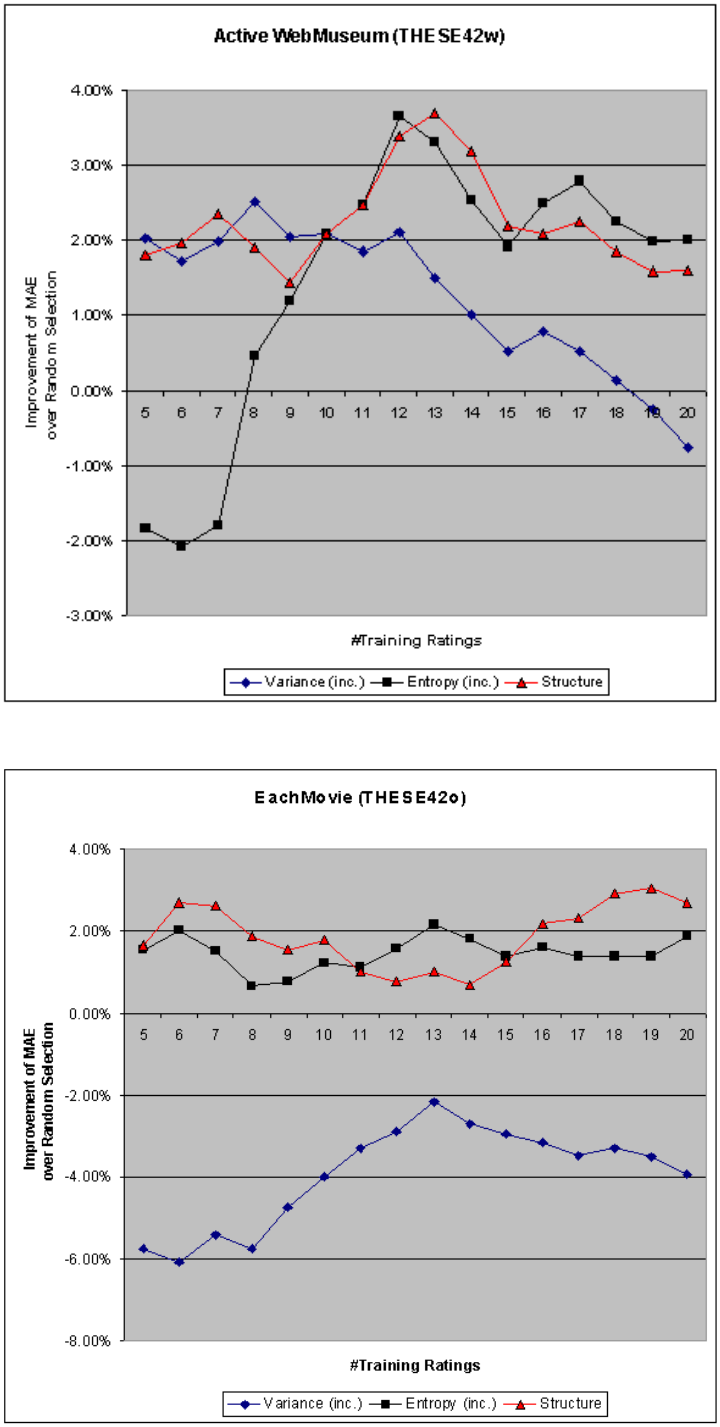



Figure 6.2: Relative improvement of the precision of selection algorithms in comparison with random selection. A positive percentage implies a higher prediction precision, i.e. a lower MAE, than the random selection.

### 6.2.3 Application of Optimization

Earlier, an optimization step for the variance and entropy selection was introduced (see Section 6.1.1). The optimization steps replace objects of the selected set which are highly correlated with other objects in the selected set with objects from the potential set which are least correlated with the selected sets. The graphs of Figure 6.3 and Figure 6.4 compare the selection algorithms in their pure form with the selection algorithms with optimization (using 5 optimization steps).

The application of optimization (5 optimization steps) improves the variance selection drastically for the EachMovie set. For the Active WebMuseum dataset, neither improvement nor degradation of the precision can be observed. This may be due to the fact that the variance selection leads to rather correlated selected sets for the EachMovie dataset but not for the Active WebMuseum dataset. This is plausible because the potential sets of the EachMovie dataset are large and therefore the choice of correlated sets might be more probable than for the Active WebMuseum dataset, where the potential sets are relatively small. With the option of optimization the variance selection becomes a feasible alternative for consistently outperforming the random selection.

The use of optimization in combination with entropy selection degrades the precision of the entropy selection, and eventually leads to worse than random selection performance for both datasets (see Figure 6.4). Apparently, entropy selection does not suffer from overly correlated objects in the selected sets and therefore the application of optimization can be considered harmful for the performance.

### 6.2.4 Application of Projection

Finally, the projection step of structure selection is investigated. The graphs of Figure 6.5 compare structure selection to structure selection without the projection step.

Surprisingly, the results indicate that the omission of projection leads to better performance of the structure selection for both algorithms. Therefore, structure selection without projection becomes a favorable choice for systematically outperforming random selection for both the EachMovie and the Active WebMuseum applications.

In our theoretical approach, the potential object vectors are projected on the

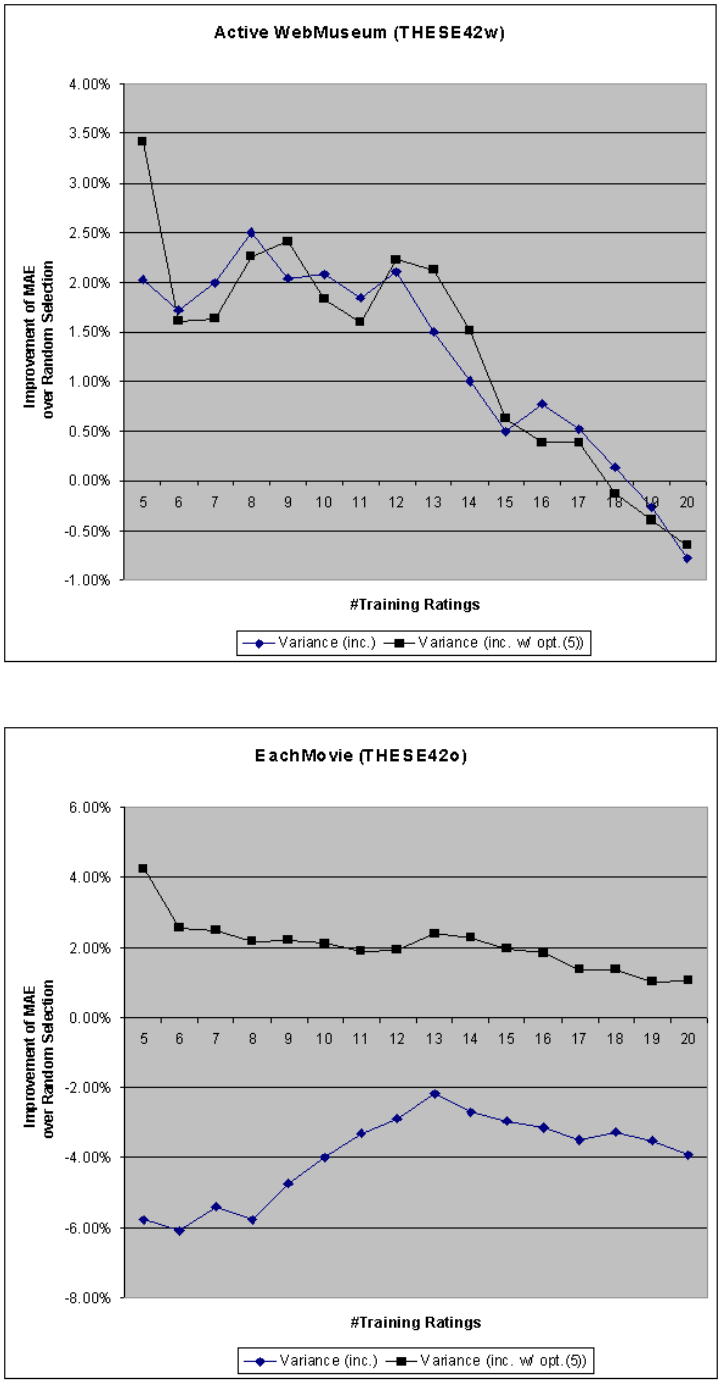



Figure 6.3: Relative improvement of the precision of variance selection and variance selection with optimization in comparison with random selection.

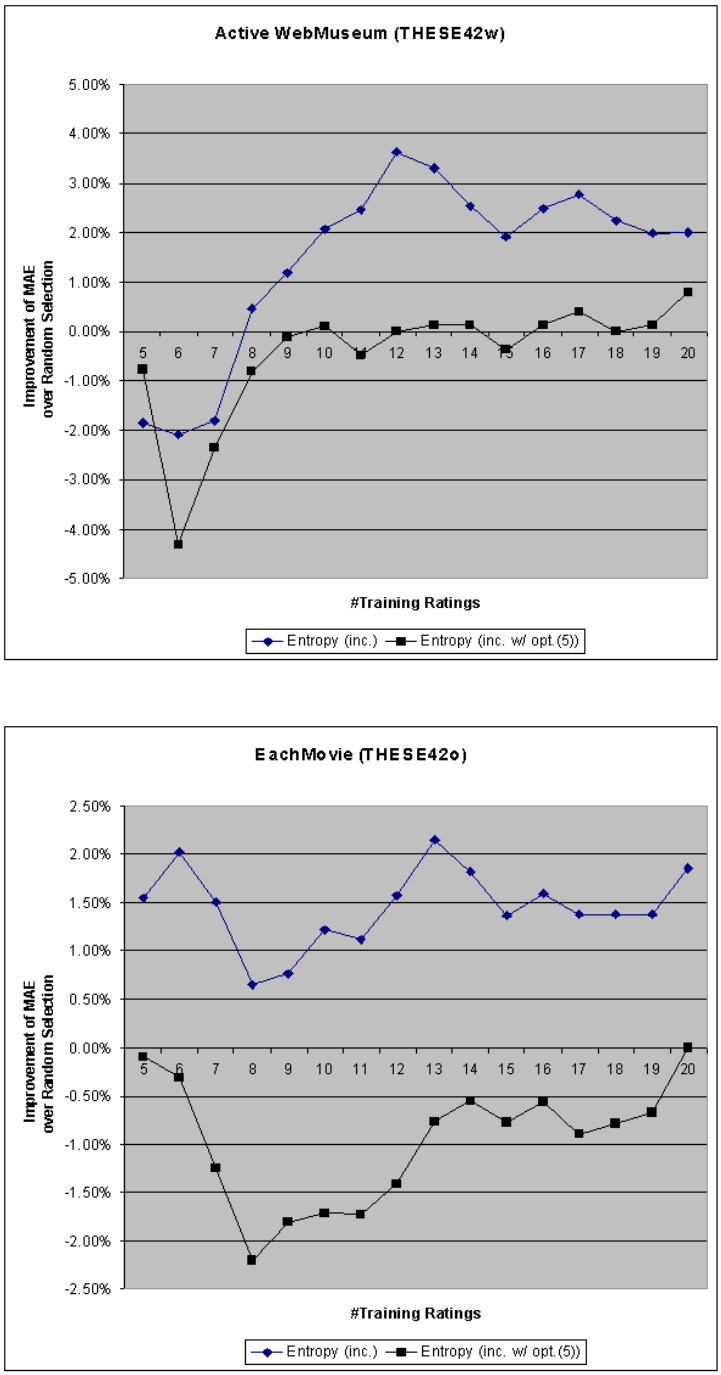



Figure 6.4: Relative improvement of the precision of entropy selection and entropy selection with optimization in comparison with random selection.

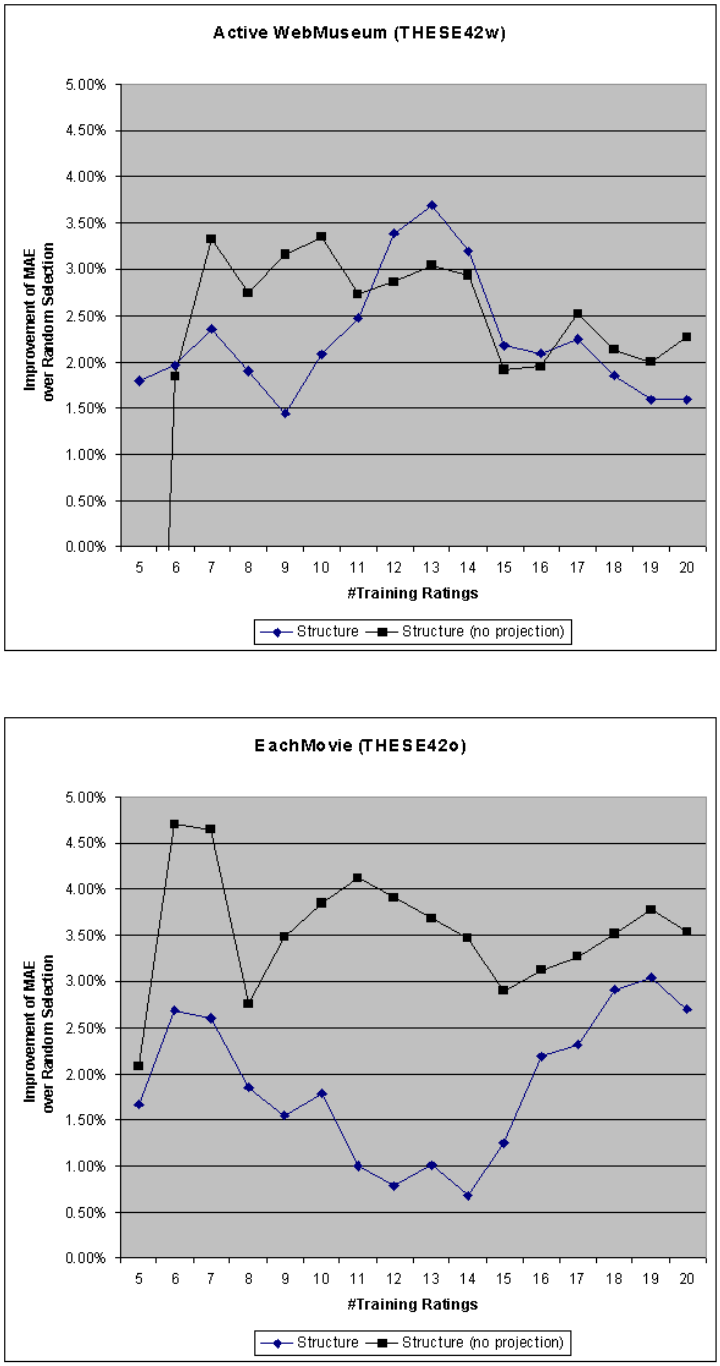



Figure 6.5: Relative improvement of the precision of structure selection and structure selection without the projection.

hyperplane defined by the previously selected object vector. We considered this step necessary in order to avoid selecting correlated objects within the same sequence. On the contrary, in our experiments, the projection step does not improve the performance. Better results can be achieved when the projection is left out. We are not sure what the reasons are for these unexpected results. For one, it is possible that object vectors are in general not very correlated, so that the effects of selecting two or more correlated vectors in the same sequence are not as negative as expected. This, however, does not explain a better performance when the projection is omitted. Another source of distress could be the application of the projection operator on the object rating vectors. The centered object rating vectors are very sparse vectors, i.e. they contain many zeroes when the rating by a particular user is unknown. As soon as some projections are performed, the zeros in the vectors disappear and are replaced by values other than zero, which are interpreted by subsequent computations as significant, while they originate from undefined positions. A special vector algebra that deals with undefined components is probably needed in order to implement the vector operations of our algorithm.

## 6.3 Applicability of Intelligent Object Selection

We were primarily motivated to research the selection of objects to be rated, by our Active WebMuseum application. For new users to obtain personalized recommendations by the underlying collaborative filtering system, it was necessary that they rate a batch of paintings before entering the network of virtual corridors. The need to *train the system* for each new user is a common one for all collaborative filtering systems. Without this step, the system can only make general assumptions about the user and therefore the recommendation service provided by the underlying collaborative filtering system is not optimal, i.e. not personalized. However, the Active WebMuseum stands out as a system for capturing user ratings: new users are generally capable of rating virtually all of the paintings immediately since it requires only a brief look to form an opinion on how much one likes the painting. Therefore, the set of potential objects to select from for new users to be rated is exceptionally large.

Controversial domains of objects which do not comply to these constraints come to mind:

- Books take a long time to read until a reader may rate the content.
- Movies have to be seen in full for the viewer to be able to assess the quality.
- Restaurants cannot be evaluated in a mouse click, etc.

In general the set of objects to choose from is limited to objects that the user may evaluate immediately, which implies the following constraints to the applicability of our proposed method:

1. The system knows which objects the user may rate immediately without performing a complicated evaluation, i.e. because the user knows the objects before.
2. The class of objects treated by the recommender system is generally easy for new users to evaluate, e.g. paintings.

In order to broaden the wide applicability of our proposed methods, these constraints have to be met by the recommender system. Constraint (1) is difficult to meet because, in general, very little is known about new users, especially about which objects they already know. However, it would be possible to use traces which the user left elsewhere for other purposes, e.g. purchase records, to determine a potential set of objects. For example, the online retailer Amazon.com could select, from a list of a client's known purchased objects, the ones which when rated would best improve the recommendation capabilities of the underlying recommender system.

However, for most applications knowledge about known objects is not available, then constraint (2) can be approximated. In order to make objects quickly assessable the system may provide short summaries about the objects. TV listings summarize shows in one sentence and make it possible to form a superficial judgment about it. Other domains can be approached similarly. Unfortunately, the ratings obtained by users who have only superficial knowledge about the objects are less reliable then ratings for objects that the user has thoroughly evaluated.

## 6.4 Conclusion

In this chapter, we have proposed and evaluated methods to efficiently select objects to be rated by a new user in a collaborative filtering system with

the goal that new users get faster, better recommendations by the collaborative filtering system. In contrast to the alternative of randomly choosing objects to be rated, the three proposed selection algorithms select objects intelligently by analyzing the rating database.

Experiments indicate that some of our selection methods make it possible to improve the precision of the prediction for a given number of ratings over random selection. In particular we found that structure selection without projection, entropy selection, and variance selection with optimization are feasible candidates as alternative selection algorithms to random selection. These selection algorithms seem to consistently outperform the random selection. The results have been validated on two different collaborative filtering datasets: the EachMovie dataset and the Active WebMuseum dataset.

Further investigations are needed to better understand the relationships between the object ratings and the performance improvement, in particular when the selection tries to select uncorrelated objects.

Our methods are directly applicable in many situations of collaborative filtering systems, and adaptations can be considered in others. It is probable that smart selection of training objects will improve the common applicability of collaborative filtering systems.

# Chapter 7

# Conclusion and Outlook

In this final chapter we give a short summary of the contributions of this dissertation. Furthermore, we allude to important issues which need to be addressed in future research related to collaborative filtering.

## 7.1 Results Obtained

Collaborative filtering depends on ratings by users. The lack of ratings (referred to as sparsity) may lead to failure of collaborative filtering algorithms. In Chapter 2 a new algorithm for collaborative filtering is presented which is designed to better address situations where rating data is lacking. This algorithm is based on hierarchical clustering. It aims at balancing prediction stability and accuracy. The algorithm was implemented and compared with other standard collaborative filtering algorithms in off-line experiments. Furthermore, cases of sparsity of ratings were formalized in order to study and experiment with collaborative filtering systems in the context of sparsity.

Collaborative filtering is a promising technology for multi-user applications on the World Wide Web. The user experiences during interaction with the Web sites can be used to personalize Web sites according to the specific interests, tastes or needs of the user. During the course of the work leading to this dissertation, a User-Adapted Web site had been implemented: the Active WebMuseum, an on-line Web museum for art paintings. The implementation of the Active WebMuseum led to general observations concerning the use of collaborative filtering for the personalization of Web sites. Furthermore, the prototype user-adapted Web site allowed the validation of algorithms developed for this dissertation. Most importantly the Active WebMuseum

made it possible to collect a unique collaborative filtering rating dataset.

In an attempt to improve collaborative filtering prediction results, combination algorithms for combining collaborative with content-based filtering are studied in Chapter 4. The main focus in this attempt was the use of *weak* content-based information which can easily be indexed, such as color or texture, to augment the prediction performance of collaborative filtering algorithms, especially for cases of sparsity. Two combination approaches are motivated, presented and validated in off-line experiments using the dataset collected in the Active WebMuseum.

When collaborative filtering is applied in an on-line application, such as the Active WebMuseum, two important issues arise:

- How are the predictions used to personalize the application?
- How is the personalization performance, as perceived by the user, measured?

In Chapter 5 the multi-corridor-access-paradigm is described. This paradigm provides a model for how the prediction results can be used for the personalization of a Web site and at the same time provides a model for deriving metrics for measuring the performance of collaborative filtering prediction in the context of this paradigm. This approach is validated by off-line experiments.

When new users start using a collaborative filtering system, they need to rate objects so that the system can adapt to their preferences. In order to make the initial learning phase more efficient, several approaches for selecting the first objects to be rated by new users are presented in Chapter 6. The proposed methods are validated in off-line experiments and lead to improvements over random selection.

## 7.2 Outlook for Future Work

One of the main scopes of this work was the accuracy of collaborative filtering algorithms. The improvement of accuracy is still an important issue. Future work should be directed at applying well-known technologies of other fields, such as multi-media indexing or machine-learning, to improve prediction accuracy of collaborative filtering.

In order to evaluate the performance of collaborative filtering, better-adapted metrics are necessary to assess the value that collaborative filtering adds to

an application. A possible direction is the pursuit of strategies similar to the multi-corridor-access-paradigm. On-line experiments would be a good way to assess the real value that users perceive by personalization efforts based on collaborative filtering.

In order to accommodate future improvements of collaborative filtering algorithms, the data model should be refined. For example with the current rating matrix it is not possible to model complexities in the user's interests such as

- Drifting interest with time.
- Many distinct interests for the same person.
- Context specific interests etc.

The open access to many services and the logging of interactions between the users and the Internet leads to the risk of improper use of private information. This risk becomes especially apparent when the main purpose of an application, such as a collaborative filtering application, is to collect information in order to most accurately model the user profile. The privacy of users within collaborative filtering systems needs to be protected. Algorithms are needed which guarantee users privacy during the use of collaborative filtering systems, in the same way as encryption technology allows users to access their bank accounts without fear of eavesdroppers.

# Annexe A

# Résumé

Le filtrage collaboratif est une technologie récente qui permet d'obtenir une recommandation personnalisée. Il est employé pour des systèmes de recommandation et en particulier pour la personnalisation de sites Web. Les utilisateurs indiquent leurs préférences (c.-à-d. évaluent des objets), et le système de filtrage collaboratif associe ce nouveau profil à d'autres profils d'utilisateurs existants et similaires, afin de pouvoir prédire des préférences. Cette thèse se concentre sur l'amélioration des algorithmes de filtrage collaboratif et leur application dans des sites Web.

Le cas où les préférences de l'utilisateur sont peu nombreuses (ou "sparsity") pose un problème au filtrage collaboratif. Nous proposons un algorithme de filtrage collaboratif basé sur un groupement hiérarchique des profils d'utilisateurs. Il est conçu afin d'améliorer ces situations avec peu de données disponible.

Pour étudier l'utilisation du filtrage collaboratif afin de personnaliser des applications sur le Web (user-adapted Web sites), nous avons conçu un prototype d'un site Web qui s'adapte à l'utilisateur : L'Active WebMuseum, un musée en ligne pour des peintures d'art.

Afin d'améliorer la personnalisation, le filtrage collaboratif est combiné au filtrage par le contenu (couleur, texture d'une peinture). Nous proposons et comparons deux approches pour cette combinaison.

Dans une application en ligne, telle que l'Active WebMuseum, il est important de définir comment les predictions sont utilisées pour personnaliser l'application et comment la personnalisation est évaluée par l'utilisateur. Nous proposons le "Multi-Corridor-Access-Paradigm"comme modèle pour formaliser l'interaction entre l'utilisateur et le site Web personnalisé. Il permet aussi

de déduire des métriques afin de mesurer la performance liée à la personnalisation.

Afin de rendre la phase d'initialisation (nouveaux utilisateurs) plus efficace, nous proposons plusieurs approches pour choisir les premiers objets susceptibles d'être évalués par les nouveaux utilisateurs.

# Annexe B

# Résumé Étendu

Ce chapitre reprend les principaux points abordés dans ma thèse. Il est écrit en français, contrairement à ma thèse qui est rédigée en anglais. Dans un premier temps, je décris les domaines d'application, les motivations et les buts de ma recherche. Puis, dans un second temps, je présente les diverses contributions apportées par mon travail de thèse.

## B.1 Introduction

La société moderne est aujourd'hui basée sur l'information. Elle évolue très rapidement grâce à Internet. Des millions d'utilisateurs "surfent" chaque jour sur Internet, dans le but de trouver différents types d'informations. Bien souvent, ils se retrouvent alors submergés par l'abondance d'information disponible de la multitude des sites Web existants.

Il est aujourd'hui primordial de canaliser les moyens d'accéder à l'information. Pour cela, il est nécessaire de se pencher sur un certain nombre de problèmes :

- Il est important de fournir aux utilisateurs des outils de recommandation et de filtrage leur permettant de maîtriser cette quantité énorme d'information.
- Compte tenu de la variété des intérêts des différents utilisateurs, il est crucial de personnaliser le contenu d'un site Web en fonction de la personne qui le consulte.

Ce problème de personnalisation devient particulièrement important pour les sites Web commerciaux, qui ont l'obligation de rester attrayants et compétitifs

afin de mieux satisfaire leurs utilisateurs. La personnalisation permettrait donc une interaction beaucoup plus efficace entre l'utilisateur et le site Web.

L'approche classique de recommandation et de personnalisation consiste à filtrer l'information par son contenu. On compare les caractéristiques de l'information au profil de l'utilisateur. Cette approche soulève de nombreuses difficultés, par exemple comment obtenir ces caractéristiques, comment construire le profil, comment faire évoluer un profil lorsque les intérêts changent, etc...

C'est dans ce contexte, que le filtrage collaboratif devient une technologie particulièrement adéquate. Cette approche récente permet de prédire l'intérêt d'une information pour un utilisateur, en se basant sur les avis d'autres utilisateurs. Elle présente de nombreux avantages :

- Elle offre une grande diversité du type d'informations pouvant être traité. Des systèmes ont déjà été expérimentés pour recommander des articles, des musiques, des films, des histoires drôles, etc ...
- Elle s'adapte bien à la philosophie d'Internet. Chaque utilisateur donne un peu d'information (des avis) et en retour reçoit beaucoup (des prévisions fondées sur les avis des autres).
- Elle tire le meilleur de l'aspect collectif. Lorsqu'un utilisateur rajoute des avis, le système de prédiction s'améliore pour tous les utilisateurs.

Cependant, le filtrage collaboratif est une technologie relativement nouvelle et il reste encore beaucoup de problèmes à résoudre :

- Quelle est la sensibilité d'un système de filtrage collaboratif au nombre d'utilisateurs utilisant le système ? Il est clair qu'un faible nombre d'utilisateurs mène à une petite base de données d'avis, ce qui limite la qualité de la personnalisation. C'est ce que l'on appelle communément le problème de "sparsity".
- Comment un système de filtrage collaboratif peut-il être intégré de façon cohérente dans un site Web, sans imposer de contraintes irréalistes sur la charte graphique et l'apparence de l'interface utilisateur ?

Ma thèse présente des résultats significatifs liés à cinq défis dans le filtrage collaboratif. La figure B.1 offre une vue d'ensemble de ces défis.

### B.1.1 Le Manque d'Informations sur les Préférences de l'utilisateur

Le filtrage collaboratif prédit les préférences qu'un utilisateur cible pourrait avoir sur un objet (une vidéo, une musique, une peinture, une information). Ces prédictions reposent sur les bases de données de préférences déjà existantes pour d'autres utilisateurs ayant un profil similaire à l'utilisateur cible

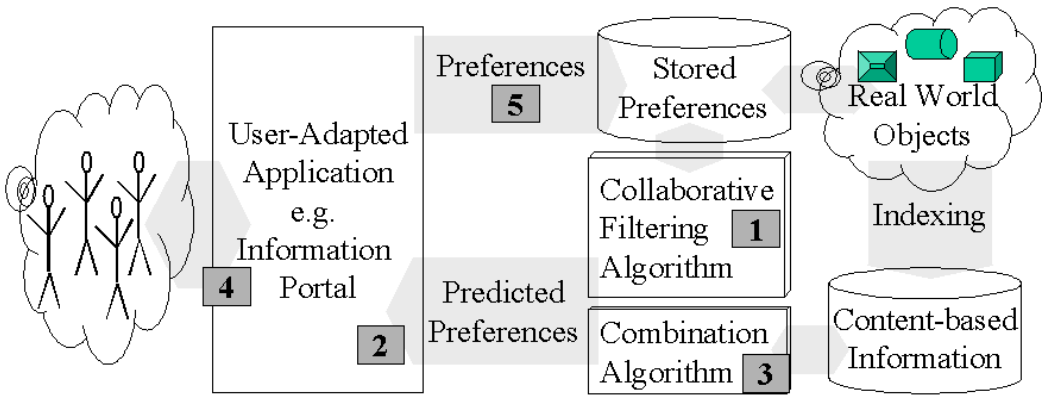



FIG. B.1 – Vue d'ensemble des point-clés de filtrage collaboratif : Les point-clés abordés dans ma thèse sont représentés dans ce diagramme par différentes cellules associées à un numéro :

**1** Comment les algorithmes de filtrage collaboratif peuvent-ils faire face au problème de "sparsity" ?

**2** Comment peut-on employer des prédictions de préférences pour personnaliser une application ?

**3** Comment le filtrage collaboratif peut-il être combiné avec le filtrage par le contenu pour des médias complexes (vidéo, musique, peinture) ?

**4** Comment mesurer l'utilité de la personnalisation dans des applications adaptées à l'utilisateur ?

**5** Comment peut-on être efficace pour accéder aux préférences de l'utilisateur, sans le noyer dans des multitudes de questions ?

pour d'autres objets.

Le manque d'informations sur les préférences de l'utilisateur mène à l'échec du filtrage collaboratif. Cet échec est en premier lieu dû au fait que la similitude entre les utilisateurs ne peut pas être déterminée, et en second lieu au fait que d'autres préférences ne sont pas disponibles en nombre suffisant pour en déduire une prédiction personnalisée.

Le manque de données de préférences est très typique dans les systèmes de filtrage collaboratif. Des cas typiques de ce type de situation sont énumérés ci-dessous :

**Le cas "nouvel utilisateur" :** Il est difficile de comparer le profil d'un nouvel utilisateur pour qui aucune ou très peu de préférences sont connues, avec d'autres utilisateurs ayant déjà une base de données bien établie. Ceci implique qu'il est difficile de faire une prédiction pour un nouvel utilisateur.

**Le cas "nouvel objet" :** Il est difficile de faire une prédiction sur un nouvel objet pour lequel très peu de préférences sont connues.

**Le cas "Bootstrap" :** Une combinaison des deux cas surgit quand un nouveau système de filtrage collaboratif est créé. En général les utilisateurs fournissent leurs préférences à un système de filtrage collaboratif dans le but de recevoir des prédictions de préférences en retour. Mais, les nouveaux systèmes de filtrage collaboratif ne possèdent pas suffisamment de préférences pour faire une personnalisation efficace. Ils manquent donc d'intérêt pour les utilisateurs potentiels et ne peuvent ainsi pas augmenter leur banque de données de préférences. Ce mécanisme mène à un dilemme appelé communément "Bootstrap".

Le manque de banque de données sur les préférences des utilisateurs est traité dans ma thèse de la manière suivante :

- Un algorithme de filtrage collaboratif a été conçu en se basant sur un regroupement hiérarchique des profils d'utilisateurs (hierarchical clustering). Il exploite l'information fournie sur les préférences plus efficacement que les algorithmes généralement utilisés.
- Le filtrage collaboratif associé au filtrage par le contenu est étudié afin de combiner les avantages des deux approches et ainsi d'améliorer l'efficacité de la prévision.
- Pour traiter le cas particulier de l'utilisateur nouveau, des techniques sont explorées afin de rassembler de manière plus efficace les préférences des utilisateurs en choisissant intelligemment les objets à évaluer.

### B.1.2 Création des Sites Web Adaptés à l'Utilisateur

Les applications se trouvant sur un site Web ont l'avantage inhérent de pouvoir supporter des utilisateurs multiples ce qui positionne le filtrage collaboratif comme une technologie de support potentielle. Une utilisation prometteuse du filtrage collaboratif est la personnalisation d'un site Web, conçu en s'adaptant aux prédictions des préférences de l'utilisateur. Pour cela, les points suivants doivent être abordés :

- Des règles générales sont indispensables pour intégrer le filtrage collaboratif dans une application établie sur le Web.
- Des mesures d'évaluation sont nécessaires, pour déterminer l'utilité pour les utilisateurs d'une personnalisation par filtrage collaboratif.

Ma thèse aborde ces questions par les approches suivantes :

- La conception et le développement d'un prototype d'application adapté à l'utilisateur : "Active WebMuseum". Cette application utilise le filtrage collaboratif afin d'obtenir un haut niveau de personnalisation.
- La présentation des objets dans l'application est basée sur des prédictions de préférences. L'interaction de l'utilisateur avec ses prédictions a été formalisée avec le concept de "multi-corridor-access-paradigm". Ce concept est d'une grande aide pour évaluer l'utilité des efforts de personnalisation.

### B.1.3 Méthodologie de Recherche

Une majeure partie des résultats produits dans ma thèse a été réalisée à partir d'expériences. Ces expériences ont été conçues dans le but d'évaluer l'efficacité des améliorations et des modifications d'algorithmes de filtrage collaboratif déjà existant. Pour cela, j'ai utilisé des banques de données de préférences déjà existantes établies dans le passé pour d'autres projets de filtrage collaboratif, ainsi que sur un système prototype de filtrage collaboratif en ligne, qui emploie des algorithmes et des méthodes décrits dans ma thèse.

Pendant cette thèse, une bibliothèque de logiciel a été créée pour le filtrage collaboratif. Cette bibliothèque (employant des classes C++) a permis la mise en oeuvre facile d'expériences, ainsi qu'une application du prototype en ligne.

## B.2 Résultats et Conclusion

Ma thèse a permis d'apporter diverses contributions au filtrage collaboratif :

Le filtrage collaboratif dépend des notes ou des préférences des utilisateurs. Le manque de notes ("sparsity") peut mener à l'échec des algorithmes de filtrage collaboratif. Dans le chapitre 2, je présente un nouvel algorithme dans le cadre du filtrage collaboratif. Cet algorithme a pour but d'améliorer les situations où l'on manque de données sur les préférences des utilisateurs. Cet algorithme est basé sur un regroupement hiérarchique des profils des utilisateurs ("Hierarchical Clustering"). Il vise à équilibrer la stabilité et l'exactitude de la prédiction. L'algorithme a été mis en application et comparé à d'autres algorithmes standards de filtrage collaboratif dans des expériences. De plus, le manque de données sur les préférences des utilisateurs ("sparsity") a été formalisé afin d'étudier les systèmes de filtrage collaboratif dans ce contexte.

Le filtrage collaboratif est une technologie prometteuse pour des applications multi-utilisateurs sur le World Wide Web. Les différentes expériences de chaque utilisateur lors de l'interaction avec le site Web peuvent être utilisées pour affiner la personnalisation du site Web en fonction des intérêts, du goût ou des besoins spécifiques de chaque utilisateur. Lors de cette thèse, un site Web adapté à l'utilisateur a été mis en application : L'Active WebMuseum, un musée en ligne pour des peintures d'art. La mise en place de ce musée virtuel mène aux observations générales concernant l'utilisation du filtrage collaboratif pour la personnalisation de sites Web. De plus, le site Web personnalisé à l'utilisateur a permis la validation des algorithmes développés dans cette thèse. D'une manière primordiale l'Active WebMuseum a permis de rassembler un ensemble de données unique de filtrage collaboratif. L'Active WebMuseum est décrit dans le chapitre 3.

Afin d'améliorer les algorithmes de filtrage collaboratif, le filtrage par le contenu combiné au filtrage collaboratif est étudié dans le chapitre 4. Le but de cette recherche est l'utilisation d'information sur le contenu des objets, telles que la couleur ou la texture d'une peinture. Pour affiner la capacité de prédiction des algorithmes de filtrage collaboratif, particulièrement pour des cas de "sparsity", deux approches de combinaison sont présentées et validées dans les expériences. Ces expériences utilisent l'ensemble des données rassemblées dans l'Active WebMuseum.

Quand le filtrage collaboratif est appliqué dans un programme en ligne, tel que l'Active WebMuseum, deux questions importantes surgissent :

– Comment les prévisions sont-elles employées pour personnaliser l'application ?
– Comment peut-on mesurer l´efficacité de la personnalisation, perçu par l'utilisateur ?

Dans le chapitre 5 le "multi-corridor-access-paradigm"est décrit. Il fournit un modèle de la façon dont les prédictions peuvent être utilisées pour la personnalisation d'un site Web et fournit simultanément un modèle pour dériver la métrique afin de mesurer la performance de la prévision du filtrage collaboratif dans le contexte de ce paradigme. Cette approche est validée par des expériences.

Quand de nouveaux utilisateurs commencent à utiliser un système de filtrage collaboratif, ils doivent évaluer des objets pour que le système puisse s'adapter à leurs préférences. Afin de rendre la phase initiale plus efficace, plusieurs approches pour choisir les premiers objets susceptibles d'être évalués par les nouveaux utilisateurs sont présentées dans le chapitre 6. Les méthodes proposées sont validées par des expériences et amènent à des améliorations sur le choix des d'objets.

# Bibliography

[1] Joshua Alspector, Alexander Kolcz, and Nachimuthu Karunanithi. Feature-based and clique-based user models for movie selection: A comparative study. *User Modeling and User Adapted Interaction*, pages 279–304, 1997.

[2] Joshua Alspector, Alexander Kolcz, and Nachimuthu Karunanithi. Comparing feature-based and clique-based user models for movie selection. In *DL '98. Proceedings of the third ACM Conference on Digital libraries*, pages 11–18, 1998.

[3] Brian Amento, William C. Hill, Loren G. Terveen, Deborah Hix, and Peter Ju. An empirical evaluation of user interfaces for topic management of web sites. In *CHI'99*, pages 552–559, 1999.

[4] Christopher Avery, Paul Resnick, and Richard Zeckhauser. The market for evaluations. *American Economic Review*, 89(3):564–584, 1999.

[5] Marko Balabanovic. An interface for learning multi-topic user profiles from impicit feedback. In *AAAI Workshop for Recommender Systems*, 1998.

[6] Marko Balabanovic. *Learning to Surf: Multiagent Systems for Adaptive Web Page Recommendation*. Thesi, Stanford University, March 1998.

[7] Marko Balabanovic and Yoav Shoham. Fab: Content-based, collaborative recommendation. *Communications of the ACM*, 40(3):66–72, March 1997.

[8] Chumki Basu, Haym Hirsh, and William Cohen. Recommendation as classification: Using social and content-based information in recommendation. In *AAAI Workshop for Recommender Systems*, 1998.

[9] Patrick Baudisch. Recommending tv programs: How far can we get at zero user effort? In *AAAI Workshop for Recommender Systems*, 1998.

[10] Nicholas J. Belkin and W. Bruce Croft. Information filtering and information retrieval: Two sides of the same coin. *Communications of the ACM*, 35(12):29–38, December 1992.

[11] M.W. Berry, S.T. Dumais, and G.W. O'Brien. Using linear algebra for intelligent information retrieval. *SIAM Review*, 37(4), 1995.

[12] D. Billsus and M. Pazzani. Revising user profiles: The search for interesting web sites. In *Proceedings of the Third AAAI International Workshop on Multistrategy Learning*, 1996.

[13] Daniel Billsus and Michael J. Pazzani. Learning collaborative information filters. In *AAAI Workshop for Recommender Systems*, 1998.

[14] Arvin L. Blum and Pat Langley. Selection of relevant features and examples in machine learning. *Artificial Intelligence*, 97(1–2):245–271, 1997.

[15] Jack Breese, David Heckerman, and Carl Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence*, Madison, WI, July 1998. Morgan Kaufmann Publisher.

[16] Jack Breese, David Heckerman, and Carl Kadie. Empirical analysis of predictive algorithms for collaborative filtering. Technical Report MSR-TR-98-12, Microsoft Research, 1998.

[17] Marie-France Bruandet. Outline of a knowledge base model for an intelligent information retrieval system. In *Proceedings of the tenth annual international ACM SIGIR conference on Research and development in information retrieval*, pages 33–43, 1987.

[18] Robin Burke. Semantic ratings and heuristic similarity for collaborative filtering. In *AAAI Workshop on Knowledge-based Electronic Markets. Austin, TX. July, 2000*, 2000.

[19] Miguel A. Carreia-Perpinan. A review of dimension reduction techniques. Technical report, University of Sheffield, 1997. Technical Report CS-96-09.

[20] Liren Chen and Katia Sycara. Webmate: A personal agent for browsing and searching. Technical report, The Robotics Institut, Carnegie Mellon Institute, September 1997.

[21] Mark Claypool, Anuja Glokhale, Tim Miranda, Pavel Murnikov, Dmitry Netes, and Matthew Sartin. Combining content-based and collaborative filters in an online newspaper. In *ACM SIGIR'99 Recommender System Workshop*, 1999.

[22] David A. Cohn, Zoubin Ghahramani, and Michael I. Jordan. Active learning with statistical models. In G. Tesauro, D. Touretzky, and T. Leen, editors, *Advances in Neural Information Processing Systems*, volume 7, pages 705–712. The MIT Press, 1995.

[23] C. Colombo, A. Del Bimbo, and I. Genovesi. Interactive image retrieval by color distribution. In *Proceedings IEEE International Conference on Multimedia Computing and Systems*, pages 255–258. IEEE Computer Society, 1998.

[24] Ph.D. Dan R. Greening. Building consumer trust with accurate product recommendations. Technical report, LikeMinds, 1997.

[25] S. Deerwester, S. Dumais, G. Furnas, T. Landauer, and R. Hashman. Indexing by latent semantic indexing. *Journal of the American Society for Information Science*, 41(6), 1990.

[26] Joaquin Delgado and Naohiro Ishii. Memory-based weighted-majority prediction for recommender systems. In *ACM SIGIR'99 Recommender System Workshop*, 1999.

[27] Joaquin Delgado and Naohiro Ishii. Online learning of user preferences in recommender systems. *International Journal of Knowledge-Based Intelligent Engineering Systems (KES)*, 3(3):194–199, July 1999.

[28] Joaquin Delgado, Naohiro Ishii, and Tomoki Ura. Content-based collaborative information filtering. In *Cooperative Information Agents II*, Lecture Notes in Artificial Intelligence 1435, pages 206–215. Springer Verlag Berlin Heidelberg New York, 1998.

[29] Joaquin A. Delgado. *Agent-based information Filtering and Recommender Systems*. PhD thesis, Nagoya Institute of Technology, 2000.

[30] Anne Eisenberg. Find me a file, cache me a catch. New York Times, February 2000. `http://www.nytimes.com/library/tech/00/02/circuits/articles/10matc.html\%`.

[31] P. Faudemay, L. Chen, C. Montacie, M.J. Caraty, and X. Tu. Segmentation multi-canaux de videos en séquences. In *Coresa 97*, 1997.

[32] Robert Fung and Brendan Del Favero. Applying baysian networls to information retrieval. *Communications of the ACM*, 38(3):42–48, March 1995.

[33] David Goldberg, David Nichols, Brian M. Oki, and Douglas Terry. Collaborative filtering to weave an information tapestry. *Communications of the ACM*, 35(12):61–70, December 1992.

[34] Nathaniel Good, J. Ben Schafer, Joseph A. Konstan, Al Borchers, Badrul Sarwar, Jon Herlocker, and John Riedl. Combining collaborative filtering with personal agents for better recommendations. In *Proceedings of the AAAI'99*. AAAI Press, 1999.

[35] Daniel R. Greening. Collaborative filtering for web marketing efforts. In *AAAI Workshop for Recommender Systems*, 1998.

[36] Dhruv Gupta, Mark Digiovanni, Hiro Narita, and Ken Goldberg. Jester 2.0: A new lineartime collaborative filtering algorithm applied to jokes. In *Workshop on Recommender Systems: Algorithms and Evaluation*, August 1999.

[37] Vu Ha and Peter Haddawy. Towards case-based preference elicitation: Similarity measures on preference structures. In *Proceedings of Fourteenth Conference on Uncertaincy in Artificial Intelligence*, 1998.

[38] David Heckerman and Michael P Wellman. Bayesian networks. *Communications of the ACM*, 38(3):27–30, March 1995.

[39] Jonathan L. Herlocker. Position statement - explanations in recommender systems. In *ACM SIGCHI'99 Workshop: Interacting with Recommender Systems*, 1999.

[40] Jonathan L. Herlocker, Joseph A. Konstan, Al Borchers, and John Riedl. An algorithmic framework for performing collaborative filtering. In *Conference on Research and Development in Information Retrieval*, 1999.

[41] Jonathan Lee Herlocker. *Understanding and Improving Collaborative Filtering Systems*. PhD thesis, University of Minnesota, September 2000.

[42] J.B. Hey. System and method for recommending items. U.S. Patent 4,996,642, 1989.

[43] J.B. Hey. System and method of predicting subjective reactions. U.S. Patent 4,870,579, 1989.

[44] Will Hill, Larry Stead, Mark Rosenstein, and George Furnas. Recommending and evaluating choices in a virtual community of use. In *CHI'95 Conference Proceedings*, pages 194–201. ACM, 1995.

[45] William C. Hill and Loren G. Terveen. Involving remote users in continuous design of web content. In *Symposium on Designing Interactive Systems*, pages 137–145, 1997.

[46] Sukumal Imudum and B. Clifford Neuman. A framework supporting collaborative filtering for internet information. In *AAAI Workshop for Recommender Systems*, 1998.

[47] Vijay S. Iyengar, Chidanand Apte, and Tong Zhang. Active learning using adaptive resampling. In *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, August 2000.

[48] M. Jaczynski and B. Trousse. Broadway : A world wide web browsing advisor reusing past navigations from a group of users. In *Proceedings of the Third UK Case-Based Reasoning Workshop (UKCBR3), Manchester, UK*, September 1997.

[49] Kenichi Kamiya, Martin Röscheisen, and Terry Winograd. Grassroots: A system providing a uniform framework for communicating, structuring, sharing information, and organizing people. In *Proceedings of the Fifth World-Wide Web Conference*, 1996. Also published in part as a short paper for CHI'96 (conference companion).

[50] Henry Kautz, Bart Selman, and Mehul Shah. Combining social networks and collaborative filtering. *Communications of the ACM*, 40(3):63–65, March 1997.

[51] Arnd Kohrs and Bernard Merialdo. Clustering for collaborative filtering applications. In *Computational Intelligence for Modelling, Control & Automation (CIMCA'99), Vienna*, pages 199–204. IOS Press, February 1999.

[52] Arnd Kohrs and Bernard Merialdo. Improving collaborative filtering with multimedia indexing techniques to create user-adapting web sites. In *Proceedings of the 7th ACM Multimedia Conference, Orlando*. ACM, 1999.

[53] Arnd Kohrs and Bernard Merialdo. Using color and texture indexing to improve collaborative filtering of art paintings. In *Proceedings of the European Workshop on Content-Based Multimedia Indexing (CBMI'99)*, 1999.

[54] Arnd Kohrs and Bernard Merialdo. Using category-based collaborative filtering in the active webmuseum. In *IEEE International Conference on Multimedia and Expo, New York*, 2000.

[55] Arnd Kohrs and Bernard Merialdo. Creating user-adapted web sites by the use of collaborative filtering. *Interfaces for the Active Web. Special Issue of Interacting with Computers, The Interdisciplinary Journal of Human-Computer Interaction*, 2001. Accepted for publication.

[56] Arnd Kohrs and Bernard Merialdo. Improving collaborative filtering for new users by smart object selection. In *International Conference on Media Futures, Florence, Italy*, 2001.

[57] Joseph A. Konstan. Adding value in the digital age. `http://www.netperception.com/`, 1997.

[58] Joseph A. Konstan, Bradley N. Miller, David Maltz, Jonathan L. Herlocker, Lee R. Gordan, and John Riedl. Grouplens: Applying collaborative filtering to usenet news. *Communications of the ACM*, 40(3):77–87, March 1997.

[59] Bernard Lang and Francois Rouaix. The v6 engine. WWW5 Workshop, Paris, May 1996.

[60] Likeminds. Corporate web site. `http://www.macromedia.com/software/likeminds/`, 2001. (Likeminds has been aquired by Macromedia Corporation recently).

[61] Linkeminds. Moviecritic prototype. `http://www.moviecritic.com`, 1998.

[62] Shoshana Loeb. Architecting personalized delivery of multimedia information. *Communications of the ACM*, 35(12), December 1992.

[63] Christopher Lueg. Issues un understanding collaborative filtering. In *ACM SIGCHI'99 Workshop: Interacting with Recommender Systems*, 1999.

[64] Patty Maes. Agents that reduce work and information overload. *Communications of the ACM*, 7:31–40, 1994.

[65] T. W. Malone, K. R. Grant, F.A. Turbak, S.A. Brobst, , and Cohen D. Intelligent information sharing systems. *Communcations of the ACM*, 30(5):390–402, 1987.

[66] David Maltz and Kate Ehrlich. Pointing the way: Active collaborative filtering. In *Proceedings CHI'95*, 1995.

[67] David A. Maltz. Distributing information for collaborative filtering on usenet new news. Master's thesis, MIT, 1994.

[68] Bernard Merialdo. A corpus-based approach to video indexing. Technical report, Multimedia Communications Department,Institut Eurecom, 1997.

[69] Bradley N. Miller, John T. Riedl, and Joseph A. Konstan. Experience with grouplens: Making usenet useful again. In USENIX, editor, *1997 Annual Technical Conference*, pages 219–233, Berkeley, CA, USA, 1997. USENIX.

[70] J. Mostafa, S. Mukhopadhyay, W. Lam, and M. Palakal. A multilevel to intelligent information filtering: Model, system, and evaluation. *ACM Transactions on Information Systems*, 15(4):368–399, October 1997.

[71] Henning Mueller, Wolfgang Mueller, Stéphane Marchand-Maillet, and David McG Squire. Strategies for positive and negative relevance feedback in image retrieval. In *In Proceedings of the International Conference on Pattern Recognition (ICPR'2000)*, 2000.

[72] Atsuyoshi Nakamura, Naoki Abe, Hiroshi Matoba, and Katsuhiro Ochiai. Automatic recording agent for digital video server. In *ACM Multimedia Conference*, pages 57–66. ACM, Nov 2000.

[73] Chahab Nastar. Indexation d'images par le contenu: un État de l'art. In *Proceedings of Coresa'97*, 1997.

[74] Hien Ngyen and Peter Haddawy. The decision theoretic video advisor. In *AAAI Workshop for Recommender Systems*, 1998.

[75] Douglas W. Oard and Jinmook Kim. Implicit feedback for recommender systems. In *AAAI Workshop for Recommender Systems*, 1998.

[76] Douglas W. Oard and Gary Marchionini. A conceptual framework for text filtering. Technical report, Medical Informatics and Computational Intelligence Laboratory, Electrical Engineering Department, May 1996.

[77] Mark O'Connor and Jon Herlocker. Clustering items for collaborative filtering. In *ACM SIGIR'99 Recommender System Workshop*, 1999.

[78] Clark F. Olson. Parallel algorithms for hierarchical clustering. *Parallel Computing*, 21:1313–1325, 1995.

[79] O'Reilly and Associates and Web Consortium (W3C), editors. *World Wide Web Journal: The Fourth International WWW Conference Proceedings*. O'Reilly & Associates, Inc., 1996.

[80] M. Pazzani and D. Billsus. Adaptive web site agents. In *Proceedings of the Third International Conference on Autonomous Agents (Agents'99)*, May 1999.

[81] Michael J. Pazzani, Jack Muramatsu, and Daniel Billsus. Syskill & webert: Identifying interesting web sites. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence, Portland, OR*, pages 54–61, 1996.

[82] P.Bellot and M.El-Beze. Clustering by means of decision trees without learning or hierarchical and k-means like algorithms. *RIAO'2000*, 2000.

[83] D. Pennock and E. Horvitz. Collaborative filtering by personality diagnosis: A hybrid memory- and model-based approach. In *IJCAI Workshop on Machine Learning for Information Filtering, Stockholm*, 1999.

[84] David M. Pennock, Eric Horvitz, and C. Lee Giles. Social choice theory and recommender systems: Analysis of the axiomatic foundations of collaborative filtering. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence (AAAI-2000)*, pages 729–734, July 2000.

[85] M. Perkowitz and O. Etzioni. Adaptive web sites: an AI challenge. In *Fifteenth International Joint Conference on Intelligence*, 1997.

[86] Gregory Piatetsky-Shapiro. The date-mining industry coming of age. *Intelligent Systems*, pages 32–34, November/December 1999.

[87] Michael Pryor. The effects of singular value decomposition on collaborative filtering. Technical report, Dartmouth College, 1998. CS Technical Report, PCS-TR98-338.

[88] Michael Reiter and Aviel D. Rubin. Crowds: Anonymity for web transactions. Technical report, DIMACS, August 1997. `http://www.research.att.com/projects/crowds/`.

[89] Paul Resnick, Neophytos Iacovou, Mitesh Suchak, Peter Bergstrom, and John Riedl. Grouplens: An open architecture for collaborative filtering of netnews. In *Proceedings of ACM CSCW'94 Conference on Computer-Supported Cooperative Work*, Sharing Information and Creating Meaning, pages 175–186, 1994.

[90] Paul Resnik and Val R. Vian. Recommender systems. *Communications of the ACM*, 40(3):56–58, March 1997.

[91] James Rucker and Marcos J. Polanco. Personalized web navigation for the web. *Communications of the ACM*, 40(3):73–75, March 1997.

[92] Matthias Runte. *Personalisierung im Internet: Individualisierte Angebote mit Collaborative Filtering*. PhD thesis, University of Kiel, 2000.

[93] Gerard Salton and Chris Buckley. Term weighting approaches in automatic text retrieval. *Information Processing & Management*, 24(5):513–523, 1988.

[94] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Application of dimensionality reduction in recommender systems–a case study. In *ACM WebKDD Workshop*, 2000.

[95] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Analysis of recommendation algorithms for e-commerce. In *Proceedings of ACM E-Commerce*, 2000.

[96] Badrul M. Sarwar, George Karypis, Joseph A. Konstan, and John Riedl. Analysis of recommendation algorithms for e-commerce. In *ACM Conference on Electronic Commerce*, pages 158–167, 2000.

[97] Badrul M. Sarwar, Joseph A. Konstan, Al Bochers, Jon Herlocker, Brad Miller, and John Riedl. Using filtering agents to improve prediction quality in the grouplens research collaborative filtering system. In *Proceedings of ACM CSCW'98 Conference on Computer-Supported Cooperative Work*, 1998.

[98] Uprendra Shardanand. Social information filtering for music recommendation. Master's thesis, MIT, 1994.

[99] Uprendra Shardanand and Pattie Maes. Social information filtering: Algorithms for automating "word of mouth". In *Proceedings of Human Factors in Computing Systems, CHI '95*, 1995.

[100] J. R. Smith and S.-F. Chang. Local color and texture extraction and spatial query. In *IEEE Proceedings International Conference on Image Processing*, September 1996. Lausane, Switzerland.

[101] J. R. Smith and S.-F. Chang. Tools and techniques for color image retrieval. In *Symposium on Electronic Imaging: Science and Technology - Storage and Retrieval for Image and Video Databases IV*, volume volume 2670, San Jose, CA, February 1996.

[102] David MgC Squire, Wolfgang Mueller, Henning Mueller, and Thierry Pun. Content-based query of image databases: inspirations from text retrieval. *Pattern Recognition Letters (Selected Papers from The 11th Scandinavian Conference on Image Analysis SCIA '99)*, 2000.

[103] T. Mitchell T. Joachims, D. Freitag. Webwatcher: A tour guide for the world wide web. In *Proceedings of IJCAI97*, August 1997.

[104] Loren Torveen, Will Hill, Brian Amento, David McDonald, and Josh Creter. Phoaks: A system for sharing recommendations. *Communications of the ACM*, 40(3):59–62, March 1997.

[105] B. Trousse. Viewpoint management for cooperative design. In *Computational Engineering in Systems Applications (CESA'98), Nabeul-Hammamet, In Proceedings of the the IEEE, Tunisia.*, 1998.

[106] Don Turnbull. Augmenting information seeking on the world wide web using collaborative filtering techniques. Research Plan, 1998.

[107] Lyle H. Ungar and Dean P. Foster. A formal statistical approach to collaborative filtering. In *CONALD'98*, 1998.

[108] Peter Valkenburg and CHIC-Pilot Group. Standarts in a distributed indexing architecture. Draft, CHIC-Pilot Group, 1998.

[109] Kent Wittenberg, Duco Das, Will Hill, and Larry Stead. Group asynchronous browsing. In O'Reilly and Associates and Web Consortium (W3C) [79], page 748.

[110] J. Wolf, C. Aggarwal, K-L. Wu, and P. Yu. Horting hatches an egg: A new graph-theoretic approach to collaborative filtering. In *In Proceedings of ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 1999.

[111] Tak W. Yan and Hector Garcia-Molina. Index structures for information filtering under the vector space model. Technical report, Department of Computer Science, Stanford University, 1993.

[112] Tak W. Yan and Hector Garcia-Molina. Index structures for information filtering under the vector space model. submitted to ICDE'94, 1994.