Statistical Learning Approaches to Information Filtering

Dissertation im Fach Informatik
an der Fakultät für Mathematik, Informatik und Statistik
der Ludwig-Maximilians-Universität München

von

Kai Yu

Tag der Einreichung: 04 Mai, 2004
Tag der mündlichen Prüfung: 20 Juli, 2004

Berichterstatter:
Prof. Dr. Hans-Peter Kriegel, Ludwig-Maximilians-Universität München
Prof. Dr. Jiawei Han, University of Illinois at Urbana-Champaign, USA
Prof. Dr. Bernd Schürmann, Siemens AG, München

To my parents and my wife

Abstract

Enabling computer systems to understand human thinking or behaviors has ever been an exciting challenge to computer scientists. In recent years one such a topic, *information filtering*, emerges to help users find desired information items (e.g. movies, books, news) from large amount of available data, and has become crucial in many applications, like product recommendation, image retrieval, spam email filtering, news filtering, and web navigation etc..

An information filtering system must be able to understand users' *information needs*. Existing approaches either infer a user's profile by exploring his/her connections to other users, i.e. collaborative filtering (CF), or analyzing the content descriptions of liked or disliked examples annotated by the user, i.e. content-based filtering (CBF). Those methods work well to some extent, but are facing difficulties due to lack of insights into the problem.

This thesis intensively studies a wide scope of information filtering technologies. Novel and *principled* machine learning methods are proposed to model users' information needs. The work demonstrates that the uncertainty of user profiles and the connections between them can be effectively modelled by using *probability theory* and *Bayes rule*. As one major contribution of this thesis, the work clarifies the "structure" of information filtering and gives rise to principled solutions. In summary, the work of this thesis mainly covers the following three aspects:

- *Collaborative filtering*: We develop a probabilistic model for memory-based collaborative filtering (PMCF), which has clear links with classical memory-based CF. Various heuristics to improve memory-based CF have been proposed in the literature. In contrast, extensions based on PMCF can be made in a *principled* probabilistic way. With PMCF, we

describe a CF paradigm that involves interactions with users, instead of passively receiving data from users in conventional CF, and actively chooses the most informative patterns to learn, thereby greatly reduce user efforts and computational costs.

- *Content-based filtering*: One major problem for CBF is the deficiency and high dimensionality of content-descriptive features. Information items (e.g. images or articles) are typically described by high-dimensional features with mixed types of attributes, that seem to be developed independently but *intrinsically related*. We derive a generalized principle component analysis to merge high-dimensional and heterogenous content features into a low-dimensional *continuous* latent space. The derived features brings great conveniences to CBF, because most existing algorithms easily cope with low-dimensional and continuous data, and more importantly, the extracted data highlight the intrinsic semantics of original content features.

- *Hybrid filtering*: How to combine CF and CBF in an "smart" way remains one of the most challenging problems in information filtering. Little principled work exists so far. This thesis reveals that people's information needs can be naturally modelled with a *hierarchical Bayesian thinking*, where each individual's data are generated based on his/her own profile model, which itself is a sample from a *common distribution* of the population of user profiles. Users are thus connected to each other via this common distribution. Due to the complexity of such a distribution in real-world applications, usually applied parametric models are too restrictive, and we thus introduce a nonparametric hierarchical Bayesian model using *Dirichlet process*. We derive effective and efficient algorithms to learn the described model. In particular, the finally achieved hybrid filtering methods are surprisingly simple and intuitively understandable, offering clear insights to previous work on pure CF, pure CBF, and hybrid filtering.

Acknowledgements

This dissertation is based on my research work that I carried out as a Ph.D student in a joint Ph.D program between the KDD group at University of Munich (LMU) and the neural computing department of Siemens AG. During the past three and a half years, I have been extremely fortunate to have the guidance, support, and friendship of a number of people who helped me grow academically and personally.

First, I would like to thank Prof. Hans-Peter Kriegel, my supervisor, for his encouragements, constructive suggestions and constant support during this research. His door is always open to me whenever I need his help. I was impressed by his open-mindedness and academic guidance that make the KDD group at LMU so successful.

I am also greatly thankful to Prof. Jiawei Han, who kindly agreed to allocate his time on supervising my thesis, despite his extremely busy research and teaching work. I would also like to thank Prof. Martin Wirsing and Prof. Ralf Zimmer, for their very patient instructions on my oral examination.

I feel grateful to Prof. Bernd Schürmann, the leader of the neural computation department at Siemens, for his review of this thesis and constant support to my research. I appreciate his emphasis on both scientific research and real-world applications, which greatly influences my commitment to my career plan.

My co-supervisor at Siemens, Dr. Volker Tresp, is the person who had the greatest role in my intellectual development. He introduced me into the field of statistical machine learning. His enthusiasm, sharp thoughts, open-mindedness, and humor made my research a really memorable and joyful journey.

I am indebted to the friendship and fellowship with Dr. Anton Schwaig-

hofer. We had a effective and memorable cooperation during our PhD work. I was amazed by not only his solid knowledge in machine learning, but also his way of treating research, just like his way of making a cup of coffee, proceeding with serious but joyful steps.

Finishing a thesis is not a one-person thing. Here I wish to thank the following people: Prof. Xiaowei Xu, Prof. Martin Ester, Dr. Wei-Ying Ma, Zhao Xu, Shipeng Yu, Mrs. Christine Herzog, Mrs. Susanne Grienberger, Dr. Stefan Schönauer, Stefan Weber, Dr. Kai Heesche, Franz Krojer, ...

Of course, I am grateful to my parents and my wife, for their patience and *love*. Without them this work would never have come into existence.

Kai Yu
Munich, Germany
April, 2004

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Information Access Technologies: Retrieval and Filtering

Recent years have witnessed the explosive growth of the volume of digital information. A study conducted by the University of California, Berkeley (2000)[1] revealed that

> *The world's total yearly production of print, film, optical, and magnetic content would require roughly 1.5 billion gigabytes of storage. This is the equivalent of 250 megabytes per person for each man, woman, and child on earth.*

To keep being informed and entertained, people have to spend considerable time online everyday to search *information items*, like web pages, books, music, images, movies, news, and advertisements, etc. We are suffering from the problem of "information overload" [Mae94] — the gap between the overwhelming amount of information and human limitations is so large.

People need effective means to efficiently find the information that they really need, and avoid the irrelevant information that does not fit in their interests. Thus *information access* technologies emerge to meet the challenge.

[1] http://www.sims.berkeley.edu/research/projects/how-much-info/index.html

Information retrieval and *information filtering* are two major information access techniques:

- **Information Retrieval**. The research started from 1960's and traditionally focused on *textual document retrieval*. Now it has become a very wide research field that studies the representation, storage, organization, and access to digital information items (e.g. , documents, web pages, images, and multimedia, etc.). The primary goal is to retrieval relevant information items in response to user queries, while returning as few irrelevant ones as possible. One notable example is web search engine systems like Google.com [2]. In this paradigm, a user is required to specify his/her *information need* in the form of *query words*, e.g. "Monet and Painting", and then the search engine returns possibly millions of relevant web pages ordered by their relevance to the query words. In some other non-textual retrieval applications where item contents can not be indexed by key words, the retrieval can be triggered by query examples. For the instance of content-based image retrieval (CBIR) [CMOY96, FSN+95, MM99, RHOM98, CLL00], a user is often required to provide some image examples to feed the system, which then returns similar images by comparing visual similarities between images in database and given examples. Information retrieval usually aims at the scenario where information need is very dynamic and temporary. That is to say, a user normally raises a query which reflects his or her *immediate need*.

- **Information Filtering** The topic has become attractive since 1990's, motivated by the demand of personalized on-line information service, like news filtering (e.g. [RIS+94]) and movie recommender systems (e.g. [SM95]). Information filtering also aims to help people find desired information items and filter out undesired items. However, unlike information retrieval, it generally focuses on users' *long-term* and *stable* information need, often being preferences, and operates on dynamically changing information streams (e.g. email and news). Based on a user's profile, which is learned from the user's previously expressed opinions

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

> on example items, a filtering system processes a new item and takes appropriate actions that either ignore it or bring it to the user's notice. Typical applications of information filtering include recommender systems (e.g. movie, music, book and so on), news filtering systems, spam email filters, and so on. The central problem of information filtering is how to learn a user's profile and how to make decisions based on the profile.

This thesis will mainly focus on the side of information filtering, namely, modelling people's long-term information need. However, it is worthy noticing that, though "information retrieval" and "information filtering" conventionally stand for different concepts in research literature, distinguishes between them are not very fundamental. If we observe a user's way of querying a retrieval system like Google, we may find some long-term patterns. For example, a computer science scientist may always click the search results which link to Citeseer[3]. On the other hand, content-based information filtering actually has its root in information retrieval (see Sec. 1.2.3). Thus generally speaking, the work presented in this thesis is also applicable to information retrieval.

The rest of this chapter is organized as follows. In Sec. 1.2, We provide an overview to the *state-of-the-art* of information filtering, mainly covering *content-based filtering*, *collaborative filtering*, and *hybrid filtering*. By explaining the essential ideas of each approach, we will present the major research challenges, which serve as the motivations of my thesis work. Then in Sec. 1.3 we will summarize the research described in this thesis and the contributions as well. Finally the outline of the whole thesis is given in Sec. 1.4.

[3]http://citeseer.ist.psu.edu/

1.2 Information Filtering

1.2.1 Characterizing Information Items

Information filtering systems process large volumes of upcoming information items, e.g. movies, and find those which are likely to satisfy a user's information need. In order to pursue this goal, we first need to characterize information items. Nowadays people are facing quite diverse types of information, varying from news, emails, music, to video or television, whose descriptions are normally heterogenous and unstructured. We need to do *feature extraction* to convert them into forms that can be easily processed, namely, descriptive feature vectors. However, depending on some nature of items, feature vectors are often deficient in conveying the characteristics of items that account for user interests. In particular, let's consider the following situations.

- *Information items that are easily characterized.* This category typically includes textual information items, e.g. emails, news articles, and web pages. It is well known that *tf-idf* scheme [SM83] has been proven successful in representing each document in a corpus as a high-dimensional numerical vector, i.e. , term frequencies within documents penalized by term frequencies across documents. The scheme is based on the "bag-of-words" assumption—that the order of words in a document can be ignored. Since semantic contents of textual documents are well indicated by key words, the so-called "vector space model" with *tf-idf* scheme is current the most popular retrieval model in information retrieval/filtering applications [BYRN99].

- *Information items that are not easily characterized.* A typical family of information items is multimedia, including speech, music, image and video, for which many efforts have been done to extract meaningful visual [SC96, MM96, CLL00] or auditorial features [UZ98, DG02]. Alternative feature extraction is to annotate multimedia items with texts. Since manual annotation costs too much human efforts, one has to develop intelligent algorithms for automatic annotation, such as

speech-to-text via speech recognition. However, few technologies have been developed for general types of media like image, music and video. One recent progress was made in automatic indexing images with keywords [LW03]. Although nowadays new multimedia products (e.g. , movies or music CDs) are often presented with textual information, it only tells us the content of multimedia items, but does not necessarily indicate how good they are. Hence one should seek for features that indicate not only content but also quality. In general, since different features characterize information items from different perspectives, we should combine comprehensive information together to achieve the best performance.

- *Information items that are impossibly or not yet characterized.* We often encounter cases where it is almost impossible to extract meaningful features that are relevant to people's interests. One example is joke recommender systems (e.g. Jester [GRGP01])—current vector space model based on 'bag-of-words' assumption indicates nothing about the sense of humor. We need very high-level language processing technologies, that are unfortunately far beyond our current limit. In some other situations, the descriptive features of information items are not available at all. Information filtering systems should be able to meet these challenges.

Accordingly, information filtering should be able to adapt themselves to various situations of information items. We will go to this issue in Sec. 1.2.3.

1.2.2 Learning User Profiles

Understanding people's information needs is generally a fundamental problem in information access. Users usually express their information needs through *query* in information retrieval applications. While for information filtering we are always interested in learning a user's long-term information needs, referred as *user profile*. One way to build a information filter is to specify a set of filtering rules based on domain knowledge. For example, an administrator can specify a set of rules to build a spam email filtering

system. But this way may lack the flexibility to match each individual's specific requirements. Another choice is to give the handle to ordinary users (e.g. [FD92]), which, however, could be tedious for them. Moreover, human-specified filtering systems are not suitable in situations where rule-based filtering are not applicable at all.

In order to avoid the difficulties of human-specified profiling, one can go to the side of machinery—building automatic profiling strategies, which generally apply *machine learning* algorithms to learn a user's profile. A learning process generalizes the observations about a user's interests, which can be acquired by either explicitly asking the user for annotations on sample items [FD92, RIS+94, BP99], or implicitly observing the user's behaviors [MK93, JFM97, Lie95, BP99] (e.g. purchasing a music CD, or clicking a hyperlink).

By using *model-based* (or *parametric*) learning algorithms, we can explicitly construct a compact *profile model* (e.g. a classifier) to describe a user's profile and directly predict the user's interest for new items. While for *memory-based* (or *nonparametric*) algorithms, we never build any descriptive model but just retain all the observations, which implicitly convey the user's profile, and delay the learning procedure in prediction phase.

1.2.3 Information Filtering Approaches: Content Effect vs. Social Effect

We will briefly review major information filtering approaches, which predict how likely an item is to satisfy a user's information needs based on the user's profile. In this dissertation, we adopt the terminology that an item is said to be *relevant* if the item fits in a user's interest, otherwise the item is *irrelevant*. In order to make filtering algorithms suitable for specific situations, one should consider two major aspects, i.e. *content effect* and *social effect*. That is to say, we can evaluate whether an item is interesting to a user, either by examining the item's content or taking advices from others who have similar profiles with this user. As illustrated in Fig. 1.1, depending on the nature of application scenario, we should accordingly adopt different strategies, e.g.

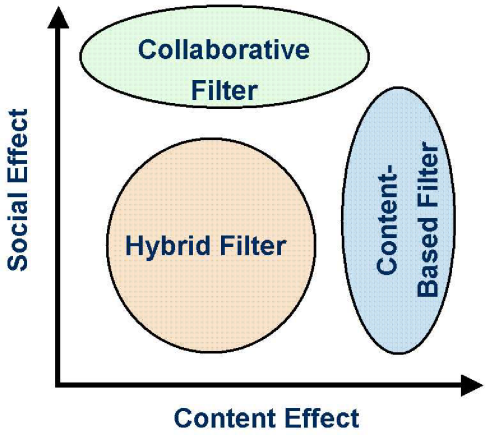


Figure 1.1: Information filtering approaches: content effect vs. social effect

content-based filtering, *collaborative filtering* and *hybrid filtering*.

Content-Based Information Filtering

As probably the most commonly applied technique, *content-based filtering* that analyzes an item's content and predicts its relevance based on the user's profile (e.g. , [BS97, MR00, PMB96, YT04]). This technique has its roots in the information retrieval community.

In this paradigm, each user is assumed to operate independently and thus the social effect is completely ignored. The approach, in a large sense, is analog to the so-called *relevance feedback* in information retrieval literature. One earliest relevance feedback technique is the well-known Rocchio's algorithm [Roc71], which adapts the query vector (i.e. a vector of term weights as the case of vector space model) by iteratively absorbing a user's relevance judgements (e.g. relevant or irrelevant) on newly returned documents. In information filtering paradigm, the tuned query vector is actually a profile model, specifying the key words as well as their relative importance. Based on the constructed user profile, a new item's relevance is measured by computing the inner product of the query vector and the item's feature vector—larger value indicating higher relevance. Operating with solely linear inner product, Rocchio's algorithm does not work well when the relevance assessment based

on content features is nonlinear, which is, unfortunately, the common case for most non-textual media.

A user profile is constructed from training data, explicitly or implicitly given by the user, and then is applied to judge other information items relevant or not. The paradigm can be formulated as a typical classification/regression problem in *machine learning*—Given a set of labelled examples, a predictive model is trained and then used to predict other previously unseen data. Compared with *ad-hoc* solutions, machine learning emphasizes on the generalization ability in the sense that the learned model not only gives a compact way to summarize the training data but also promises a good performance on future unseen data. The generalization performance is even more crucial in information filtering/retrieval applications, because we can not require a user to give too many feedbacks before we return something. It has been reported that support vector machine [Vap95], as a state-of-the-art learning algorithm, is superior to many other algorithms in text and image filtering/retrieval tasks [Joa98, DSG01, HTH00].

However, content-based filtering only works well if content effect is sufficient, i.e. when we can extract descriptive features that are highly relevant to people's interests. As pointed in Sec. 1.2.1 this is often not the case. Furthermore, due to the complexity of information sources and the heterogenous interests of a user, content-based filtering has the *small-sample problem* that profile models are often learned from insufficient training data. If a new item significantly differs from training data, the learned profile model will produce a high predictive variance on it. This thesis presents two recipes for these drawbacks. First is to derive highly indicative features to represent information items. The other solution is to take into account the social effect, i.e. users are no longer assumed to be independent. We will briefly introduce the ideas in Sec. 1.3 and then present the details in Sec. 3 and Sec. 4.

Content-based filtering has been studied in various research projects, including Web browsing (Letizia [Lie95], and Syskill&Webert [PMB96]), news filtering (NewsWeeder [Lan95], and Webmate [CS98]), and email filtering [MDH99].

	item1	item2	item3	
user 1	4	3		
user 2	1	5	4	...
user 3		4	5	
		⋮		
active user	2	?	3	

Figure 1.2: An illustration of collaborative filtering based on accumulated user rating data

Collaborative Information Filtering

Collaborative filtering methods [RIS+94, SM95, BP98, YST+04] typically accumulate a database of item annotations (or ratings), as illustrated in Fig. 1.2, cast by a large set of users, and then use those ratings to predict a query user's preferences for unseen items. Collaborative filtering does not rely on the content descriptions of items, but purely depends on preferences expressed by a set of users. These preferences can either be expressed explicitly by numeric ratings (as shown in Fig. 1.2), or can be indicated implicitly by user behaviors, such as clicking on a hyperlink, purchasing a book or reading a particular news article.

A variety of collaborative filtering algorithms have been proposed in the last decade. One can identify two major classes of collaborative filtering algorithms [BHK98], *memory-based* approaches and *model-based* approaches. Memory-based collaborative filtering can be motivated from the observation that people usually trust the recommendations from like-minded friends. These methods apply a nearest-neighbor-like scheme to predict a user's ratings based on the ratings given by like-minded users. Earliest collaborative filtering systems, e.g., Grouplens [RIS+94] and Ringo [SM95], fall into this category. In the literature, the term collaborative filtering is sometimes used to refer only to the memory-based methods.

In contrast, model-based collaborative filtering first learns a descriptive model of user preferences and then uses it for predicting ratings. Many of these methods are inspired from machine learning algorithms. Examples include neural network classifiers [BP98], induction rule learning [BHC98],

linear classifiers [ZI02], Bayesian networks [BHK98], dependency networks [HCM+00], latent class models or mixture models [HP99, Lee01], item-based collaborative filtering[SKKR01], principle component analysis based collaborative filtering [GRGP01], and hybrids of model- and memory-based approaches [PHLG00].

Collaborative filtering has been widely used in various areas ranging from recommender systems (for example, Amazon and CDnow[4]), web browsing (e.g. WebWatcher [JFM97]), to computer-supported collaborative work [SKKR00]. Related research projects include Grouplens (the first automatic collaborative filtering algorithm, [RIS+94]), Ringo [SM95], Video Recommender [HSRF95], Movielens [DKHR98], and Jester [GRGP01].

Hybrid Filtering

Pure Collaborative filtering only relies on user preferences, without incorporating the actual content of items. It often suffers from the extreme sparsity of available data, in the sense that users typically rate only very few items, thus making it difficult to compare the interests between users. Furthermore, pure collaborative filtering can not handle items for which no user has previously given annotations (or ratings). Such cases are easily handled in content-based filtering systems, which can make predictions based on the content of the new item. On the other hand, why one user likes or dislikes a joke, or prefers one CD over another is virtually difficult to formalize by content-based analysis. Similarly it is hard to derive features which represent the difference between an average news article and one of high quality. Collaborative filtering provides a powerful way to overcome these difficulties, since personal preferences, tastes, and item qualities are all carried in user annotations. It is hence necessary to build hybrid filtering systems that combine collaborative filtering and content-based filtering together, to compensate the drawbacks of each single aspect.

Recently many efforts were made in this direction. The key challenge is how to combine the two types of filters. A family of approaches, e.g. [Paz99, CGM+99], treat content-based filtering and collaborative filtering separately

[4] `www.amazon.com`, `www.cdnow.com`

and present the weighted average of both predictions as the final outcome. As another example, Fab [BS97] is a distributed implementation of a hybrid system. It maintains user profiles based on content analysis, and directly compare these profiles to determine similar users for collaborative filtering. An item is recommended to a user both when it scores highly against the user's own profile, and when it is also rated highly by users with similar profile. Basu et al [BHC98] proposed a classification approach that extends content-based filtering based on not only normal descriptive content features but also collaborative features (i.e. , other users' ratings on items). In another approach [MMN02], a different combination strategy was taken in *content-boosted collaborative filtering*, where content-based filters are built for each user and then applied to reduce the sparsity by generating pseudo ratings for non-rated items. The augmented user rating data is used to feed a collaborative filtering algorithm.

So far the problem is mainly solved in an *ad-hoc* way. There are only few examples of a unifying framework for these two basic information filtering ideas, one being the three-way aspect model [PUPL01], which builds a probabilistic generative model assuming that both terms (i.e. textual content features) and user logs are generated from some hidden variables. This approach, however, is only applicable to text data and suffers from the sparsity of data.

1.3 Research Work of this Dissertation

This dissertation will focus on building various statistical machine learning algorithms, in order to solve major problems of information filtering, including collaborative filtering, content-based filtering and hybrid filtering. Our work demonstrates that statistical learning methods can always provide principled approach to information filtering. The section briefly introduce the research work of this thesis, while details will be given in the following chapters.

1.3.1 Collaborative Filtering: A Probabilistic Memory-Based Framework

Memory-based collaborative filtering is probably the most popular algorithm applied in recommender systems [RIS+94, SM95, SM95, BHK98, YST+04]. It is intuitively understandable and also presents results as accurate as those state-of-the-art model-based methods, e.g. Bayesian networks [BHK98]. However, the main drawback comes from it very slow prediction response.

We will introduce probabilistic memory-based collaborative filtering (PMCF), a probabilistic framework for collaborative filtering systems that is similar in spirit to the classical memory-based collaborative filtering approach. As the primary ingredient, we present a probabilistic model to describe the density distribution of user preferences. We use a mixture model built on the basis of a set of stored *prototypes* of user profiles. Thus the model clearly links with memory-based collaborative filtering methods, but provides a principled view to understand the memory-based approach, and outcomes better results.

Various heuristics to improve memory-based collaborative filtering have been proposed in the literature (e.g. [RAC+02]). In contrast, extensions to PMCF can be based on a principled probabilistic way. We argue that this is one of the major advantages of PMCF. We use PMCF to derive solutions for two particularly important problems in collaborative filtering.

- *Actively acquiring user profiles*: Currently most collaborative filtering methods only *passively* receive information from a user and learn the user's profile. It should be desired if the learning system is aware of what to learn and thus can *actively* query the most useful information from users. Intuitively, this active learning strategy may help a collaborative filtering system quickly grasp a user's profile by requiring minimum user efforts. This property is particularly useful to solve the "new user problem", meaning that a collaborative filtering system can not serve for a new user whose profile is completely unknown. Within the proposed PMCF framework, an active learning component is able to sequentially choose *informative* unrated items and ask the new user for feedbacks. This active information acquiring procedure is guided by

minimizing the uncertainty of user profile, leading to very quick profile learning.

- *Reducing computational costs*: The second major extension aims at reducing the computational burden in the prediction phase typically associated with memory-based collaborative filtering. PMCF allows us to select a small subset, called the *profile space*, from a (possibly huge) database of user ratings. The selection procedure, based on PMCF framework, ensures that a small profile space can lead to predictions that are as accurate as predictions made by using the whole database of user ratings. The algorithms preserves the probabilistic density of user ratings and finally derives a very simple and intuitive algorithm—we only retain the preference patterns which are *novel*, based on previously selected data, but actually being *typical* in the whole database.

1.3.2 Content-Based Filtering: A Generalized Principal Component Analysis Model

Content-based information filtering works well only when descriptive features are relevant enough to users' information need. However, feature extraction is the major problem for most of filtering applications, due to the following reasons:

- *Deficiency of features*: People always extract low-level features from information items. For example, visual features describing color, texture or shape information can not sufficiently describe the semantics of images.

- *Heterogeneous features*: Since features are always weak, it might be helpful to combine multi-source features together. Again, for the example of images, one might consider the combination of visual features, associated short texts, and user annotations, and finally obtain feature vectors with mixed types of attributes (e.g. , continuous, categorical and binary attributes). However, it is still unclear how to directly work with such heterogeneous feature vectors with mixed types of attributes.

- *High dimensionality*: The performance of information filtering is also hampered by the high dimensionality of feature vectors. The problem is even severer after the combination of multi-source features. Therefore dimensionality reduction, a classical problem in pattern recognition, is also crucial for information filtering. Traditional methods only handles continuous data. It is a challenge that how to reduce the dimensionality of mixed types of high dimensional features.

We present a novel probabilistic latent-variable model, which can be viewed as a generalization of probabilistic principal component analysis. The new model is capable of characterizing mixed data types (continuous, binary and categorical data) by a small number of hidden continuous variables. We adopt a variational approximation to the likelihood of observations and describe an expectation-maximization (EM) algorithm to fit the model. The model allows a unified treatment to mixed types of attributes and thus brings great benefits for multivariate data analysis, dimensionality reduction, and information filtering. We demonstrate the advantages of the proposed model in a painting image recommender system.

1.3.3 Hybrid Filtering: A Hierarchical Bayesian Framework

In this part we present a theoretical framework to combine content-based filtering and collaborative filtering, based on hierarchical Bayesian model. The introduced model provides a deeper understanding towards information filtering and smoothly leads to a principled hybrid filtering algorithm. More interestingly, we demonstrate that pure collaborative filtering (e.g. the proposed PMCF in Ch. 2) and pure content-based filtering are subcases of the framework. This point indicates that the hierarchical Model is not only a way to derive principled hybrid filtering algorithm, but also a general framework for information filtering.

In the hierarchical Bayesian framework, we assume each user's profile model is generated from a *prior distribution*. Then by repeatedly sampling the prior distribution, a population of users' profile models are also generated

from the same prior distribution. In this situation the prior distribution becomes a *common prior* shared by all the users and thus can be learned from gathered data from all the users. Finally the learned prior distribution can be applied to constraint the inference for individual users by using the Bayes rule. In this framework, the common prior distribution actually serves as an *informative* Bayesian prior in the content-based profiling process for individuals (i.e. content-based filtering), and meanwhile statistically connects all the individuals in the population (i.e. the idea of collaborative filtering).

Since the common prior distribution can be in arbitrary form, it is not suitable to assume a known parametric distribution (e.g. , Gaussian) for it. We adopt a flexible nonparametric Bayesian approach to learn the common prior distribution, based on the *Dirichlet process*. In the phase of inferring individual profiles, we adopt various approximations to simplify the computation. The finally derived algorithm is quite simple and intuitive—we first train profile models for users via content-based filtering and then at the prediction phase combine many profile models to form a *committee machine*. The weights of committee members are based on like-mindedness between users.

As we know, this is probably the earliest work to theoretically combine collaborative and content-based filters in a single framework. The derived algorithm is simple and intuitive, with big potentials in information filtering applications. On the other hand, this work could also be a strong contribution to nonparametric Bayesian learning in a sense that a flexible algorithm is suggested to learn many different but related models.

1.4 Outline

The remaining chapters of this thesis are organized as follows. Ch. 2 introduces the work on probabilistic memory-based information filtering framework. Ch. 3 describes the generalized probabilistic principal component analysis for content-based information filtering. Then the hierarchical Baysian framework for hybrid filtering will be presented in Ch. 4. Finally we draw conclusions and point out future work in Ch. 5.

Chapter 2

Collaborative Filter: A Probabilistic Memory-Based Framework

2.1 Introduction

One major difficulty in designing content-based filtering systems lies in the problem of formalizing human perception and preferences. Why one user likes or dislikes a joke, or prefers one CD over another is virtually impossible to formalize. Similarly it is difficult to derive features which represent the difference between an average news article and one of high quality. Collaborative filtering provides a powerful way to overcome these difficulties. The information on personal preferences, tastes, and quality are all carried in (explicit or implicit) user ratings.

A variety of collaborative filtering algorithms have been proposed in the last decade. One can identify two major classes of collaborative filtering algorithms [BHK98], memory-based approaches and model-based approaches.

Memory-based collaborative filtering can be motivated from the observation that people usually trust the recommendations from like-minded friends. These methods apply a nearest-neighbor-like scheme to predict a user's ratings based on the ratings given by like-minded users. The first collaborative filtering systems Grouplens [RIS+94] and Ringo [SM95] fall into this category. In the literature, the term collaborative filtering is sometimes used to

refer only to the memory-based methods.

In contrast, model-based collaborative filtering first learns a descriptive model of user preferences and then uses it for predicting ratings. Many of these methods are inspired from machine learning algorithms.

2.1.1 Motivation

Up to now, research on collaborative filtering primarily focused on exploring various learning methods, hoping to improve the prediction accuracy of recommender systems. Other important aspects, like scalability, accommodating to new data, and comprehensibility have received little attention. In the following we will review five general issues which are important for collaborative filtering and greatly motivated the work presented in this paper.

Accuracy

As a central issue in collaborative filtering research, prediction accuracy has received a high degree of attention, and various methods were proposed for improvement. Still, conventional memory-based methods using Pearson correlation coefficient remain among the most successful methods in terms of accuracy. The experiments presented in Sec. 2.5.4 show that our proposed probabilistic interpretation of memory-based collaborative filtering can outperform a set of other memory- and model-based collaborative filtering approaches.

Interactive Learning of User Profiles

A recommender system cannot provide accurate service to a new user, whose preferences are initially unknown. This has been referred to as the "new user problem" [BS97, GSK+99, RAC+02] Before being able to make predictions, a collaborative filtering system typically requires the new user to rate a list of query items in an initial information gathering stage. Efficient heuristics [RAC+02] are essential to select informative query items and thus keep the

information gathering stage as short as possible, since users may easily lose patience when faced with a long list of query items.

Within our proposed probabilistic framework for collaborative filtering, we show in Sec. 2.3 how informative query items can be selected in a principled way. At each information gathering step, those query items are presented to the user which are expected to maximally sharpen the user's profile. Our experiments (see Sec. 2.5.5) confirm that this interactive approach outperforms other ways of selecting query items [RAC+02] both in terms of necessary user effort and achieved accuracy of predictions.

Efficiency

Memory-based collaborative filtering often suffers from slow response time, because each single prediction requires the scanning of a whole database of user ratings. This is a clear disadvantage when compared to the typically very fast responses of model-based collaborative filtering. In the proposed probabilistic memory-based collaborative filtering approach, predictions are generated from a carefully selected small subset of the overall database of user ratings, which we call *profile space*. As a consequence, predictions can be made much faster than in a classical memory-based collaborative filtering system. Still, the accuracy of a system using the full data set can be maintained. We will describe this process of data selection in Sec. 2.4. The results presented in Sec. 2.5.6 confirm that the constructed profile space does indeed allows a both accurate and fast prediction of user ratings.

Incrementally accommodating to new data

Recommender systems must be capable of handling new data, be it new users or new items. For example, in a music recommender system, the recommender system must be able to adapt itself to newly arising styles of music and thus new preference patterns. This suggests that the training process of any underlying collaborative filtering algorithm should be incremental. However, model-based collaborative filtering approaches are typically trained using batch algorithms. To our knowledge, little work has addressed the use

of on-line learning in collaborative filtering. Thus, re-training a model with new data can become quite expensive, in particular if it needs to be performed regularly [BHK98]. In contrast, memory-based collaborative filtering can easily accommodate to new data by simply storing them. In the proposed probabilistic memory-based collaborative filtering framework, this goal can be achieved by a straight-forward extension of the data selection procedure introduced in Sec. 2.4.

Comprehensibility

The results in [HKR00] indicate that allowing users to know more about the result-generating process can help them understand the strengths and weaknesses of collaborative filtering systems. With this knowledge, users can make low-risk decisions. For example, consider the following two cases: (1) Among Julia's like-minded users there are 50% percent of users who rated 'like' to Titanic, while 50% of them rated 'dislike'. (2) In the other case, most of her neighbors give neutral ratings to that movie. A traditional collaborative filtering system may only give a neutral rating in both of the cases. A more sophisticated system may remind Julia of the underlying reasons in the first case and, for example, output an estimated distribution of a user's rating for some item, either in graphical or textual form ("I guess you will like that movie, and I am pretty sure (or very unsure) about that"). This suggests that a probabilistic collaborative filtering approach, as presented in this paper, can improve the comprehensibility and thus the acceptance of a collaborative filtering system. Furthermore, memory-based collaborative filtering has a clear interpretation that can be easily conveyed to users, such as "You seem to be sharing opinions with user A, who liked the following items...".

2.1.2 Overview of Our Approach

In this paper, we introduce probabilistic memory-based collaborative filtering (PMcollaborative filtering), a probabilistic framework for collaborative filtering systems that is similar in spirit to the classical memory-based col-

laborative filtering approach. A schematic drawing of the components of PMCF is shown in Fig. 2.1.

As the basic ingredient, we present a probabilistic model for user preferences in Sec. 2.2. We use a mixture model built on the basis of a set of stored user profiles; thus the model clearly links with memory-based collaborative filtering methods.

Various heuristics to improve memory-based collaborative filtering have been proposed in the literature. In contrast, extensions to PMCF can be based on a principled probabilistic way. We argue that this is one of the major advantages of PMCF. We use PMCF to derive solutions for two particularly important problems in collaborative filtering.

The first one concerns the new user problem. An active learning extension to the PMCF system can actively query a user for additional information, in case the available information is insufficient.

The second major extension aims at reducing the computational burden in the prediction phase typically associated with memory-based collaborative filtering. PMCF allows us to select a small subset, called the *profile space*, from a (possibly huge) database of user ratings. The selection procedure is derived directly from the probabilistic framework and ensures that the small profile space leads to predictions that are as accurate as predictions made by using the whole data base of user ratings.

2.1.3 Structure of this Chapter

This paper is organized as follows. In Sec. 2.2, we describe the framework of probabilistic memory-based collaborative filtering (PMCF). In Sec. 2.3, we present an active learning extension of PMCF to gather information about a new user in a particularly efficient way that requires a minimum of user interaction. In Sec. 2.4, we show how to construct the profile space for the PMCF model, which is a small subset of the available user rating data. We present experimental results that demonstrate the effectiveness of PMCF, the active learning extension and the profile space construction in Sec. 2.5. We end the paper by conclusions and an outlook in Sec. 2.6.

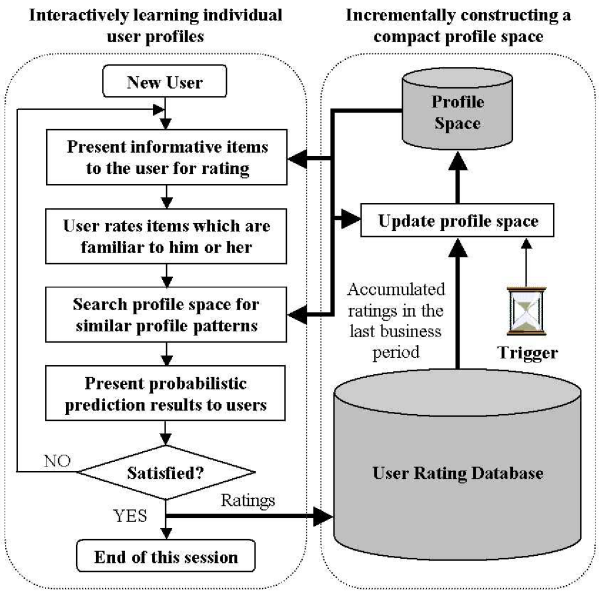


Figure 2.1: A schematic drawing of the components of probabilistic memory-based collaborative filtering (PMCF). Through an active learning scheme (presented in Sec. 2.3), the profile of a new user can be inferred with a minimum of required user effort. User ratings are stored in a database, from which a compact representation—the profile space—can be constructed in order to make fast predictions (presented in Sec. 2.4)

2.2 Probabilistic Memory-Based collaborative filtering

In this section a general probabilistic memory-based collaborative filtering (PMCF) approach is introduced. Probabilistic collaborative filtering has been a vivid research topic. Examples include Bayesian networks [BHK98], dependency networks [HCM+00], latent class models or mixture models [HP99, Lee01], and hybrids of memory- and model based systems [PHLG00]. The work presented here has been inspired by [PHLG00], in that we also aim at connecting memory- and model-based collaborative filtering in a probabilistic way. While [PHLG00] mainly focusses on making predictions, we use the probabilistic model for further extensions of the collaborative filtering system, some of which will be described in Sec. 2.3 and 2.4.

2.2.1 Notation

Suppose that we have gathered K users' ratings on a given item set $\mathcal{I}$ of size $M = |\mathcal{I}|$. Let $x_{i,j} \in \mathbb{R}$ be the rating of user i on item j and let $\mathcal{D}$ with $(\mathcal{D})_{i,j} = x_{i,j}$ be the $K \times M$ matrix of all ratings. $\mathcal{R}_i$ is the set of items for which user i has actually given ratings, $\mathcal{R}_i \subseteq \mathcal{I}$. If an item has not been rated, we set $x_{i,j}$ to a neutral rating n_i, which we will define later. We denote by $\boldsymbol{x}_i$ the vector of all ratings of user i. In the following text, user i's ratings $\boldsymbol{x}_i$ are often referred as user i's *profile*. We also maintain a smaller set of user profiles, the *profile space* $\mathcal{P}$, which consists of a subset of rows of $\mathcal{D}$. Without loss of generality, we assume that the profile space is built up[1] from the ratings of the first N users, i.e. the first N rows of $\mathcal{D}$, where typically $N \ll K$.

In collaborative filtering terminology, the *active user* is the user that queries the collaborative filtering system for recommendations on some items. We denote the active user's ratings by $\boldsymbol{a}$. By $\boldsymbol{a}^r$, we denote the ratings

[1] We will show in Sec. 2.4 how a compact and accurate profile space $\mathcal{P}$ can be incrementally built from a given set of user ratings $\mathcal{D}$.

the active user has already provided (for items $\in \mathcal{R}_a$), and $\boldsymbol{a}^n$ are the yet unknown ratings. The total rating vector $\boldsymbol{a}$ is thus the union of $\boldsymbol{a}^r$ and $\boldsymbol{a}^n$.

As mentioned above, we use a neutral rating n_i for all items a user i has not given an explicit rating, i.e. $x_{i,j} = n_i$ if $j \notin \mathcal{R}_i$. In order to compute n_i, we assume a Gaussian prior for the neutral rating with mean m_0 which is estimated as the overall mean of user ratings. If we further assume that n_i is also Gaussian distributed with mean m_0 we can estimate the neutral rating as

$$n_i = \frac{\sum_{j \in \mathcal{R}_i} x_{i,j} + C m_0}{|\mathcal{R}_i| + C} \tag{2.1}$$

where C is the ratio of the variance of the ratings for user i and the variance of m_0. We determined a suitable value for C based on cross validation experiments. We found $C = 9$ to work effectively on the data we consider.

2.2.2 A Density Model for Preference Profiles

We assume a generative probabilistic model in which the ratings $\boldsymbol{a}$ of an active user are generated based on a probability density of the form

$$p(\boldsymbol{a}|\mathcal{P}) = \frac{1}{N} \sum_{i=1}^{N} p(\boldsymbol{a}|i), \qquad \boldsymbol{x}_i \in \mathcal{P} \tag{2.2}$$

where $p(\boldsymbol{a}|i)$ is the probability of observing the active user's ratings $\boldsymbol{a}$ if we assume that a has the same profile class as the ith profile prototype in $\mathcal{P}$, i.e. user i's profile. The density expressed by Eq. (2.2) models the influences of other like-minded users' preferences on the active user a. For the mixture components $p(\boldsymbol{a}|i)$, we use Gaussian[2] density functions. Assuming

[2]We are a little inaccurate here and assume for simplicity that our rating scale is continuous and unbounded, ignoring the fact that ratings are often given on a discrete scale. One might also chose mixture components that fit particular data, for example binomial distributions for discrete ratings.

that ratings on individual items are independent, given a profile i, we get

$$\begin{aligned} p(\boldsymbol{a}|i) &= \prod_{j\in\mathcal{I}} p(a_j|i) \\ &= \prod_{j\in\mathcal{I}} \frac{(2\pi)^{-1/2}}{\sqrt{\sigma^2 + d_{j\notin\mathcal{R}_i}\sigma_0^2}} \exp\left(-\frac{1}{2}\frac{(a_j - x_{i,j})^2}{\sigma^2 + d_{j\notin\mathcal{R}_i}\sigma_0^2}\right) \end{aligned} \tag{2.3}$$

Here, $d_{j\notin\mathcal{R}_i} = 1$ if $x_{i,j}$ is unrated and $d_{j\notin\mathcal{R}_i} = 0$ otherwise. This model can be motivated as a mixture model, with the prototype profiles $\boldsymbol{x}_i$ serving as cluster centers, or as a Parzen density model on the profile space $\mathcal{P}$. The additional variance for unrated items takes into account the uncertainty of the estimated rating.

In our experiments, we set σ_0^2 to be the overall variance of user ratings. σ^2 was optimized by maximizing the leave-one-out likelihood of profiles

$$\sum_{\boldsymbol{a}\in\mathcal{P}} p(\boldsymbol{a}|\mathcal{P}\setminus\boldsymbol{a}) \tag{2.4}$$

with respect to σ^2. σ^2 is tuned after constructing the profile space (see Sec. 2.4) and left constant thereafter. Note that, technically, profiles take on different meanings: If they are part of the data base, they represent prototype vectors defining the component densities in Eq. (2.3). If we consider the active user's profile, the profile corresponds to a sample generated from the probability density defined in the same equation.

2.2.3 A Probabilistic Approach to Estimating User Ratings

We can now calculate the posterior density of the active user a's ratings on not yet rated items, denoted by $\boldsymbol{a}^n$, based on the ratings $\boldsymbol{a}^r$ user a has already

given. Using the previously defined density model for user ratings, we find

$$
\begin{aligned}
p(\boldsymbol{a}^n|\boldsymbol{a}^r,\mathcal{P}) &= \frac{p(\boldsymbol{a}^n,\boldsymbol{a}^r|\mathcal{P})}{p(\boldsymbol{a}^r|\mathcal{P})} && (2.5)\\
&= \frac{\sum_{i=1}^{N} p(\boldsymbol{a}^n,\boldsymbol{a}^r|i)}{\sum_{i=1}^{N} p(\boldsymbol{a}^r|i)} && (2.6)\\
&= \sum_{i=1}^{N} p(\boldsymbol{a}^n|i)\,\Pr(i|\boldsymbol{a}^r,\mathcal{P}). && (2.7)
\end{aligned}
$$

$\Pr(i|\boldsymbol{a}^r,\mathcal{P})$ indicates the *a posteriori* probability of user a having the ith prototype profile, given the ratings user a already has provided. It thus models the "like-mindedness" of active user a to other users i in the profile space $\mathcal{P}$:

$$\Pr(i|\boldsymbol{a}^r,\mathcal{P}) = \frac{p(\boldsymbol{a}^r|i)}{\sum_{i=1}^{N} p(\boldsymbol{a}^r|i)}. \tag{2.8}$$

Within the PMCF model, predictions for the active user are thus made by combining the predictions based on other prototype users $\boldsymbol{x}_i$, weighted by their degree of like-mindedness to user a. This puts the key idea of memory-based collaborative filtering into a probabilistic framework.

Note that the computational complexity of prediction is $O(NM)$, i.e. it is linear in the size of the profile space. In Sec. 2.4 we will show how to obtain a profile space that is much smaller than the complete user rating database $\mathcal{D}$. Making predictions only on basis of the small profile space thus brings a significant reduction of overall computational cost.

2.3 An Active Learning Approach to Learning User Profiles

In the previous section, we introduced the PMCF framework and showed how predictions can be made. In this section we will use an active learning approach to efficiently learn the profile of an individual user. The active learning approach integrates smoothly into the PMCF framework and provides a solution for the "new user problem". By presenting a set of most

informative query items in an interactive process, we can learn about the profile of a new user with a minimum of user effort.

2.3.1 The New User Problem

For users that are new to a recommender system, no information about their preferences is initially known. Thus, the recommender system typically requests them to rate a set of query items. Using the ratings on these query items, the collaborative filtering system can then start making recommendations.

There are several important reasons why this set of query items should be selected carefully: (1) Users are not willing to rate a long list of items; (2) Users cannot rate items unknown to them; (3) Rating results for some items might be very informative for determining a user's profile whereas rating results for other items might not provide useful new information. So far little work has been done to address[3] the new user problem. [RAC+02].

In the next sections, we will present an approach for selecting query items that requires particularly little user effort, yet allows fast learning about the user's preferences.

2.3.2 Identifying Informative Query Items

To achieve an efficient interactive learning of user profiles, we put the selection of query items into a decision theoretic framework (see for example Sec. 4.3 of [Jen01]). First, one needs to define a loss function, evaluating the quality of the system before querying a new item $\lambda(\boldsymbol{a}^r, \mathcal{P})$ and after querying the user for item j, $j \notin \mathcal{R}_i$ and after having obtained rating a_j. We denote the loss after querying by $\lambda(a_j, \boldsymbol{a}^r, \mathcal{P})$. The goal is now to select the query item j such that the expected loss

$$E_{p(a_j|\boldsymbol{a}^r,\mathcal{P})}\big[\lambda(a_j, \boldsymbol{a}^r, \mathcal{P})\big] \tag{2.9}$$

[3] A method for improving the accuracy of collaborative filtering systems by adding extra query items has been presented in [BZ03]. This approach might also be adapted to solve the new user problem.

is minimized. The expectation is calculated here with respect to the predicted probability of user a's ratings for item j.

The most important ingredient is the loss function $\lambda(a_j, \boldsymbol{a}^r, \mathcal{P})$. We propose to use the entropy of the like-mindedness $\Pr(i|\boldsymbol{a}^r, \mathcal{P})$ as the loss function. $\Pr(i|\boldsymbol{a}^r, \mathcal{P})$ describes the like-mindedness of a user i in the profile space $\mathcal{P}$ with active user a, given a's ratings $\boldsymbol{a}^r$. In an extreme case, $\Pr(i|\boldsymbol{a}^r, \mathcal{P})$ has a uniform distribution, which means that the profile of user a is completely unclear. In contrast, a sharp peak in the distribution of $\Pr(i|\boldsymbol{a}^r, \mathcal{P})$ indicates that user a has similar preferences as a small group of like-minded users. It thus seems natural to choose those query items that minimize the uncertainty (thus, the entropy) of user a's like-mindedness.

Putting this into a formal setting, we can write for the loss function

$$\lambda(a_j, \boldsymbol{a}^r, \mathcal{P}) = -\sum_{i=1}^{N} \Pr(i|a_j, \boldsymbol{a}^r, \mathcal{P}) \log \Pr(i|a_j, \boldsymbol{a}^r, \mathcal{P}). \tag{2.10}$$

By $\Pr(i|\boldsymbol{a}^r, a_j, \mathcal{P})$ we denote like-mindedness, computed with an updated vector of ratings for the active user, who now also has rated the (previously unrated) item j.

We can now define the expected benefit (Sec. 4.3.2 of [Jen01]) for querying item j as

$$E[B(j)] = E_{p(a_j|\boldsymbol{a}^r, \mathcal{P})} \left[\lambda(a_j, \boldsymbol{a}^r, \mathcal{P})\right] - \lambda(\boldsymbol{a}^r, \mathcal{P}) \tag{2.11}$$

and terminate the query process if the expected benefit is less than a threshold related to the cost of querying.

Our algorithm for query item selection is myopic in the sense that the algorithm only looks one step ahead. In contrast, a hyperopic algorithm would aim at finding the optimal *sequence* of query items to be presented. However, since hyperopic optimization is computationally intractable, myopia is a standard approximation used in sequential decision-making problems [HBR94, Ton01].

2.3.3 Identifying the Items Possibly Known to the Active User

If we wanted to use the active learning approach described in the previous section directly, we would most often get a "don't know" as the answer to most of the query items. Users of a collaborative filtering system can typically provide ratings for only few of the items. For example, in a recommender system for movies, users may typically have seen a few dozen movies out of the several hundred movies contained in the data base. It may be quite informative to know the user's opinion on an unusual movie, yet it is likely that the user will not be able to give this movie any rating.

Thus, we must also predict the probability that a user is able to rate[4] a given query item. This can be achieved by again referring to the like-mindedness of users. In Eq. (2.5), predictions for active user a were built from a sum of other users' ratings, weighted by their degree of like-mindedness $\Pr(i|\boldsymbol{a}^r, \mathcal{P})$. Similarly, we can predict the probability of user a being able to rate item j, given his or her other ratings $\boldsymbol{a}^r$, by checking user a's like-minded users:

$$\Pr(\text{user } a \text{ can rate item } j|\boldsymbol{a}^r, \mathcal{P}) = \sum_{i=1}^{N} \Pr(\text{user } a \text{ can rate item } j|i) \Pr(i|\boldsymbol{a}^r, \mathcal{P}) \tag{2.12}$$

$\Pr(\text{user } a \text{ can rate item } j|i)$ is the probability that a can rate item j, given that users a and i (as described by prototype profile $\boldsymbol{x}_i$) agree on which items they are able to rate. We assume for simplicity that user a can rate exactly the same[5] movies as user i:

$$\Pr(\text{user } a \text{ can rate item } j|i) = \begin{cases} 1 & \text{if user } i \text{ has rated item } j \\ 0 & \text{otherwise} \end{cases} \tag{2.13}$$

[4] Another way of solving this problem would be to integrate this probability into the loss function Eq. (2.10) for the active learning approach. We do not pursue this solution in the present article.

[5] This is a strong assumption, yet due to the weighting introduced by the like-mindedness we obtain meaningful results

2.3.4 A Summary of the Active Learning Process

Using the ideas described in the previous sections, we propose the following iterative scheme to learn the profile of the active user a:

1. Out of the set of items that have not yet been rated by user a, find those k_1 items with the highest probability of being known to user a, i.e. those items with the highest value for Eq. (2.12).

2. Out of these k_1 items, select a subset of k_2 items that lead to the highest reduction of uncertainty about the user's profile, i.e. the items with the highest expected benefit in Eq. (2.11).

3. Display those k_2 items to the user for rating. Collect the ratings and update the vector of ratings $\boldsymbol{a}$.

4. Terminate if the user is not willing to answer any more queries or if the expected benefit of querying (as defined in Eq. (2.11)) is below a certain threshold. Otherwise, go to step 1.

In the very first step, where nothing is known about user a, we assume equal like-mindedness of user a with all profiles in $\mathcal{P}$. Thus, user a will be presented the k_2 most popular items as query items.

2.3.5 Implementation

Parameters for Active Learning

The value of k_1 (see step 1 of Sec. 2.3.4) should be carefully selected. If k_1 is too small, for example, as small as k_2, then the selection procedure is too much biased by Eq. (2.12), and thus might miss out informative items—the system performs too little exploration. If k_1 is too large, too many items will be presented to the user which the user is not able to rate. In cross validation experiments, we found that $k_1 = 50$ gives the best results for the data we consider. The value for k_2 is rather uncritical. We used $k_2 = 10$, because it seems reasonable to display 10 items on a normal-sized PC screen. Thus, at each iteration, we first find the 50 candidate items with largest probability

of being known, and then identify 10 query items according to the expected reduction of uncertainty in like-mindedness.

Computational Complexity

The most costly part in this active learning approach is the evaluation of Eq. (2.11), where the expected reduction of uncertainty in like-mindedness is computed. The algorithm needs to exhaust $O(ck_1)$ possibilities of user feedbacks at each iteration (where c is the number of ratings a user might possibly give to a presented query item, and k_1 is the number of candidate items) and calculate the entropy of the like-mindedness for each case. This again requires evaluating Eq. (2.2) with changed preference vector $\boldsymbol{a}$. Fortunately, Eq. (2.2) factorizes along items, thus the distances only need to be re-calculated along the dimensions of the newly rated items. This greatly reduces the overall computational cost.

Alternative Methods

Several of the approaches proposed in the active learning literature may be adopted for collaborative filtering. A common approach is *uncertainty sampling* [LC94], which has been successfully applied to text categorization [LC94] and image retrieval [Ton01] to reduce the number of training examples. The general idea behind all proposed variants of uncertainty sampling is to present the unlabeled examples for which the outcome is most uncertain, based on the current predictions. In a collaborative filtering scenario, one is interested in predicting a user's ratings for non-rated items. Thus, the variance of predictions $\operatorname{var} p(a_j|\boldsymbol{a}^r, \mathcal{P})$ is an appropriate measure of uncertainty. An advantage of this approach lies in its low computational cost, since we only have to compute the predictions $p(a_j|\boldsymbol{a}^r, \mathcal{P})$ for all yet unrated items.

Another low complexity method for query item selection is *entropy sampling* [RAC+02]. Here, we consider $\Pr_j(s)$, the fraction of users who had given a particular rating $s \in \{s_1, \ldots, s_c\}$ for item j. Query items are selected such that the entropy of $\Pr_j(s)$ is maximized.

We will show in Sec. 2.5.5 that the method based on uncertainty of like-

mindedness (as outlined in Sec. 2.3.2) achieves best results, both in terms of achieved accuracy and in terms of required user input.

2.4 Incrementally Constructing Profile Space

In Sec. 2.2 we introduced a probabilistic model for describing user preferences. This model was based on a given set of user profiles, the profile space $\mathcal{P}$. In this section, we will show how this profile space can be constructed, by selecting informative user profiles from the overall database of user ratings $\mathcal{D}$. Since the profile space typically contains only a low number of user profiles (as compared to the often huge $\mathcal{D}$), it allows us to build compact models and make predictions efficiently, while maintaining a high accuracy. It thus solves the well-known problem that predictions of traditional memory-based collaborative filtering methods are rather time-consuming.

2.4.1 Kullback-Leibler Divergence for User Profile Sampling

Let's assume that there exists an optimal density model for user ratings, which we denote by $p^{\text{opt}}(\boldsymbol{x})$. Naturally we do not have access to this optimal model but we work with a non-optimal model $p(\boldsymbol{x}|\mathcal{P})$, as given in Eq. (2.2), based on some profile space $\mathcal{P}$. The key idea of our proposed selection procedure is to select the profile space $\mathcal{P}$ such that the density $p(\boldsymbol{x}|\mathcal{P})$ is as close as possible to the optimal density $p^{\text{opt}}(\boldsymbol{x})$.

To measure the distance of these two distributions, we use the Kullback-Leibler divergence (KL-divergence [CT91]). We denote the KL-divergence of the two distributions by

$$D\left(p(\boldsymbol{x}|\mathcal{P})||p^{\text{opt}}(\boldsymbol{x})\right) = \int p^{\text{opt}}(\boldsymbol{x}) \log \frac{p^{\text{opt}}(\boldsymbol{x})}{p(\boldsymbol{x}|\mathcal{P})} d\boldsymbol{x} \tag{2.14}$$

where the integral is over the whole space of user rating vectors. The KL-divergence is always non-negative and is zero when two compared distributions are identical. Assuming that the total set of user ratings $\mathcal{D}$ constitutes

a set of independent samples drawn from $p^{\text{opt}}(\boldsymbol{x})$, we can approximate the KL-divergence by Monte-Carlo integration [Fis96]:

$$\tilde{D}\left(p(\boldsymbol{x}|\mathcal{P})||p^{\text{opt}}(\boldsymbol{x})\right) = \frac{1}{K}\sum_{i=1}^{K}\log\frac{p^{\text{opt}}(\boldsymbol{x}_i)}{p(\boldsymbol{x}_i|\mathcal{P})} \tag{2.15}$$

$$= \frac{1}{K}\log\frac{p^{\text{opt}}(\mathcal{D})}{p(\mathcal{D}|\mathcal{P})} \tag{2.16}$$

where K is the number of users in $\mathcal{D}$.

As stated above, we wish to minimize the KL-divergence $\tilde{D}(p(\boldsymbol{x}|\mathcal{P})||p^{\text{opt}}(\boldsymbol{x}))$ so that the density $p(\boldsymbol{x}|\mathcal{P})$ best approximates $p^{\text{opt}}(\boldsymbol{x})$. Since $p^{\text{opt}}(\mathcal{D})$ is constant, Eq. (2.15) can be minimized by maximizing the likelihood of the user rating database $\mathcal{D}$ with respect to the profile space $\mathcal{P}$. Finding the optimal profile space $\mathcal{P}$ is clearly an intractable task, we thus switch to an iterative greedy approach for constructing $\mathcal{P}$.

2.4.2 Incremental Profile Space Construction

For constructing the profile space $\mathcal{P}$ from a data base $\mathcal{D}$ of user ratings, we consider an incremental scenario. Given the current profile space $\mathcal{P}$, which profile pattern $\boldsymbol{x}_i \in \mathcal{D}$ should be included such that the updated profile space $\mathcal{P} \cup \boldsymbol{x}_i$ can achieve the maximum reduction in KL-divergence, according to Eq. (2.15)?

The reduction in KL-divergence caused by including $\boldsymbol{x}_i$ in $\mathcal{P}$ can be written as

$$\Delta_i = \tilde{D}\left(p(\boldsymbol{x}|\mathcal{P})||p^{\text{opt}}(\boldsymbol{x})\right) - \tilde{D}\left(p(\boldsymbol{x}|\mathcal{P}\cup\boldsymbol{x}_i)||p^{\text{opt}}(\boldsymbol{x})\right) \tag{2.17}$$

$$= \frac{1}{K}\log\frac{p(\mathcal{D}|\mathcal{P}\cup\boldsymbol{x}_i)}{p(\mathcal{D}|\mathcal{P})} \tag{2.18}$$

Mind that this step causes the optimal density $p^{\text{opt}}(\boldsymbol{x})$ to drop out. According to Bayes' rule, the likelihood of the overall data $\mathcal{D}$, given the updated profile space $\mathcal{P} \cup \boldsymbol{x}_i$ can be written as follows:

$$p(\mathcal{D}|\mathcal{P}\cup\boldsymbol{x}_i) = p(\mathcal{D}|\mathcal{P})\frac{p(\boldsymbol{x}_i|\mathcal{D})}{p(\boldsymbol{x}_i|\mathcal{P})} \tag{2.19}$$

where $p(\boldsymbol{x}_i|\mathcal{D})$ is the likelihood of $\boldsymbol{x}_i$, based on a model that uses the complete data as the profile space. Combining Eq. (2.17) and (2.19), the optimal profile $\boldsymbol{x}$ to be selected is given by:

$$\arg\max_i \Delta_i = \arg\max_{\boldsymbol{x}_i \in \mathcal{D}\setminus\mathcal{P}} \frac{p(\boldsymbol{x}_i|\mathcal{D})}{p(\boldsymbol{x}_i|\mathcal{P})} \tag{2.20}$$

An intuitive interpretation of this selection scheme is as follows: Eq. (2.20) suggests that profiles $\boldsymbol{x}_i$ with low $p(\boldsymbol{x}_i|\mathcal{P})$ but high $p(\boldsymbol{x}_i|\mathcal{D})$ will be selected. $p(\boldsymbol{x}_i|\mathcal{P})$ encodes how likely a profile $\boldsymbol{x}_i$ is, given our current knowledge $\mathcal{P}$, while $p(\boldsymbol{x}_i|\mathcal{D})$ encodes the likelihood and thus the "degree of typicalness" of profile $\boldsymbol{x}_i$ in the overall data $\mathcal{D}$. The profile selection scheme thus focusses on profiles that are novel to our current knowledge (encoded by the current profile space), but are in fact typical in the real world (represented by the whole data $\mathcal{D}$). Thus, this sampling scheme will result in removing redundancies (we only focus on novel data that is not yet included in the profile space) and in removing outliers (outliers can be considered untypical data).

Still, Eq. (2.20) does not give a practical algorithm, since it requires evaluating $O(K)$ profiles, $K = |\mathcal{D}|$, where each evaluation requires $O(K)$ steps to actually build $p(\boldsymbol{x}_i|\mathcal{D})$. This leads to the clearly impractical overall runtime of $O(K^2)$. Practical variants will be discussed in the next section.

2.4.3 Implementation

Constructing a profile space $\mathcal{P}$ according to Eq. (2.20) is sometimes referred to as *full greedy selection*. This can only be done efficiently if the associated objective function can be computed cheaply—which is not the case for the likelihood ratio we consider here. In related problems, it has been suggested to consider small subsets of candidates, evaluate the objective function for each candidate, and select the best candidate out of this subset (see, for example, Sec. 6.5 of [SS02]).

We thus obtain the following profile sampling scheme to build $\mathcal{P}$ from $\mathcal{D}$:

1. Select a subset $\mathcal{C}$ of candidate profiles at random from $\mathcal{D} \setminus \mathcal{P}$.
2. Compute the likelihood $p(\boldsymbol{x}_i|\mathcal{P})$ for each candidate profile $\boldsymbol{x}_i \in \mathcal{C}$, based on the current profile space $\mathcal{P}$.

3. Compute the likelihood $p(\boldsymbol{x}_i|\mathcal{D})$ for each $\boldsymbol{x}_i \in \mathcal{C}$, based on the complete data $\mathcal{D}$.

4. Include the best candidate profile in the profile space:

$$\mathcal{P} \leftarrow \mathcal{P} \cup \arg\max_{\boldsymbol{x}_i \in \mathcal{C}} \frac{p(\boldsymbol{x}_i|\mathcal{D})}{p(\boldsymbol{x}_i|\mathcal{P})} \tag{2.21}$$

5. Terminate, if the profile space has reached a given maximum size or if the reduction of KL-divergence is below a given threshold.

It has been suggested in [SS02] that subsets of size $|\mathcal{C}| = 59$ can be guaranteed to select profiles that are better than 95% of all other profiles with confidence 95%. In our experiments, we aim at achieving higher efficiency and thus use subsets of size $|\mathcal{C}| = 7$. This corresponds to selecting profiles that are better than 80% of all others with confidence 80%.

2.4.4 Constructing Profile Spaces in a Dynamic Environment

While the sampling approach presented in the previous section works fine in a static environment with a fixed database of user ratings, it needs to be refined to work in a dynamic environment. The dynamics arises from changing preferences patterns (for example, new styles of music in a music recommender system) and the ever growing database of user ratings. Since user profiles are typically collected incrementally, we suggest an incremental extension to the basic sampling scheme presented in Sec. 2.4.3. We assume that the profile space is being updated after a fixed period of time, e.g. each day or week. The new user profiles gathered during this period are being processed and some of them will be added to the profile space.

Assuming that we have a data base of user ratings $\mathcal{D}$. From $\mathcal{D}$, we have already constructed a profile space $\mathcal{P}$. After collecting user profile data for some time, we get an updated data base $\mathcal{D}^+$, with $\mathcal{D}^+ = \mathcal{D} \cup \Delta\mathcal{D}$. In order to build the according profile space $\mathcal{P}^+$, select the set of candidate items $\mathcal{C}$ from $\mathcal{D}^+$. Select the most informative profile and update the profile space

$\mathcal{P}^+$:

$$\mathcal{P}^+ \leftarrow \mathcal{P}^+ \cup \arg\max_{\boldsymbol{x}_i \in \mathcal{C}} \frac{p(\boldsymbol{x}_i|\mathcal{D}^+)}{p(\boldsymbol{x}_i|\mathcal{P}^+)} \tag{2.22}$$

Terminate if the new profile space $\mathcal{P}^+$ has reached a given size or if none of the candidate items $\boldsymbol{x}_i \in \mathcal{C}$ leads to a reduction of KL-divergence. Otherwise, select a new candidate set and proceed.

Through this straight-forward extension we can retain the basic idea of using a small profile space, as introduced in Sec. 2.4.2, while now being capable of incrementally processing new data.[6]

2.4.5 Computational Complexity

For the basic profile space construction, as outlined in Sec. 2.4.2, the computational complexity is as follows:

Evaluating the density function $p(\boldsymbol{x}_i|\mathcal{D})$ for a candidate profile $\boldsymbol{x}_i$ (see Eq. (2.20)) requires scanning the whole data base $\mathcal{D}$ with K user ratings. Its complexity is thus $O(K)$. Since all potential profile spaces $\mathcal{P}$ are subsets of $\mathcal{D}$, $\mathcal{P} \subseteq \mathcal{D}$, one can easily construct $p(\boldsymbol{x}_i|\mathcal{P})$ as a "by-product" when scanning the data base in order to find $p(\boldsymbol{x}_i|\mathcal{D})$. Both steps are thus $O(K)$, with $K = |\mathcal{D}|$. Constructing a profile space of size N requires a total of $O(KN)$ operations. Once the profile space, is constructed, one also needs to update the variance σ^2 according to Eq. (2.4). This is done with a leave-one-out scheme, its complexity is thus $O(N^2)$.

Since one would typically keep the profile space resident in memory, the memory consumption of the profile space construction is $O(N)$, with $N = |\mathcal{P}|$.

The suggested method for constructing a profile space $\mathcal{P}$ thus has the same complexity as making predictions in a traditional memory-based collaborative filtering method. Yet, as described in Sec. 2.4.4, profile space construction can be seen as a background process that is being triggered by time or when unused computing power is available. Thus, its time consumption is not visible to a user of the collaborative filtering system. We argue

[6] One might also consider the case of removing certain (outdated) user profiles from $\mathcal{P}$, yet we did not evaluate this idea in the present work.

that the so achieved *shift of workload* is important, since it greatly improves the efficiency of front-end processing, namely, making predictions.

2.5 Empirical Study

In this section we report results from applying the probabilistic memory-based collaborative filtering (PMCF) framework to two collaborative filtering benchmark data sets, EachMovie and Jester. We report results on prediction accuracy, efficiency of learning individual user profiles (based on the ideas presented in Sec. 2.3) and accuracy of the constructed profile spaces (using the incremental scenario of Sec. 2.4).

2.5.1 Data Sets

We apply the PMCF framework to the following two benchmark data sets:

- EachMovie[7] contains ratings from $72,916$ users on $1,628$ movies. User ratings were recorded on a discrete scale from zero to five. On average, each user rated about 30 movies. EachMovie is one of the most widely used data sets in recommender system research.
- Jester[8] contains ratings from $17,998$ users on 100 jokes, continuously valued from -10 to 10. On average, each user rated about 50 jokes. We transferred the ratings to a discrete scale $\{-10, -9, \ldots, 9, 10\}$.

2.5.2 Evaluation Metrics and Experimental Setup

In collaborative filtering research, one is typically interested in two types of accuracy, the accuracy for predicting ratings and the accuracy for making

[7] Available from the Digital Equipment Research Center at `http://www.research.digital.com/SRC/EachMovie/`

[8] Jester stems from a WWW-based joke recommender system, developed at the University of California, Berkeley [GRGP01]. It is available from `http://shadow.ieor.berkeley.edu/humor/`

recommendations. The first one measures the performance when explicitly predicting the active users ratings on some unseen items. The second one focusses on finding an accurate ordering of a set of unseen items, in order to recommend the top ranked items to the active user. These two scenarios require different experimental setups and metrics, which we will describe now.

Accuracy of Predicting Ratings

To evaluate the accuracy when the collaborative filtering system is asked to predict an active user's ratings, we use the mean absolute error (MAE, the average absolute difference between the actual ratings and the predicted ratings). This measure has been widely used in previous collaborative filtering research [BHK98, HKBR99, PHLG00, RIS+94, SM95].

We examine the accuracy of predictions in two experimental setups, AllButOne and Given5, which were introduced in [BHK98]:

- AllButOne evaluates the prediction accuracy when sufficient information about the active user is available. For each active user (from the test set[9]) we randomly hide one of the rated items and predict its rating, based on the ratings on other non-hidden items.

- Given5 evaluates the performance of a collaborative filtering system when only little information about a user is available. For each active user, we retain only 5 ratings. The collaborative filtering system predicts the ratings of hidden items, based on the 5 visible ratings.

It has been argued that the accuracy of a collaborative filtering system is most critical when predicting extreme ratings (very high or very low) for items [PHLG00, SM95]. Since the goal of a collaborative filtering system is to make recommendations, high accuracy on high and low rated items is of most importance. One would like to present those items (in particular,

[9]This naturally requires that we skip users in the test set that have only rated one single item, respectively users that rated less than 6 items in the Given5 setup.

products) that the active user likes most, and avoid anything the user dislikes. Therefore, for both of the above AllButOne and Given5 setups, we use two settings Extreme and All (see [PHLG00]). The All setting corresponds to the standard case where the collaborative filtering system is asked to predict any of the hidden ratings. In the Extreme setting, the collaborative filtering system only predicts ratings that are on the end of the rating scales. For EachMovie, these extreme ratings are $\{0, 1, 2, 4, 5\}$, and ratings below -5 or above 5 for Jester.

Accuracy of Recommendations

We use precision and recall to evaluate the accuracy of recommendations. These two metrics have been extensively used in information retrieval and collaborative filtering research [BP98, LAR02]. In our experiments, precision is the percentage of items recommended to a user that the user actually likes. Recall is the percentage of items the user likes that are also recommended by the collaborative filtering system. For the EachMovie data, we assume that users like those items (movies) which they had rated 4 or 5. For Jester, we assume that users like those jokes that had been given a rating larger than 5.

To compute precision and recall, we use the following setup. For each active user (from the test set[10]) we randomly hide 30 of the user's ratings[11]. The collaborative filtering system then predicts the ratings for these items, based on the remaining visible ratings. The top ranked items out of these 30 items are then recommended to the user and used to evaluate precision and recall. We compute precision and recall for two cases, where we either recommend the top 5 or the top 10 ranked items. These two cases will be labeled Top5 and Top10 in the table of results.

[10]The setup requires that we skip users who had rated less than 31 items.

[11]We experimented with different numbers here, for example, hiding 20 of the user's ratings. We found that the results were consistent throughout these experiments, thus we present only results for one setup.

Training and Test Sets

For comparing the accuracy of predictions of PMcollaborative filtering with that of Bayesian network collaborative filtering [BHK98] on the EachMovie data, we use exactly the same split as reported in [BHK98, PHLG00] with training and test sets of size 5000. To be able to evaluate the significance of our results, we use training and test sets (both of size 5000) drawn at random from the data, and repeat this five times.

Similarly, for evaluating the accuracy of prediction on the Jester data, we take the first 5000 users as the training set, and the next 5000 as the test set. Five random splits are used for significance tests.

As mentioned above, we skip all test users that have rated less than 31 items when computing precision and recall, respectively less than two (six) items when computing the MAE in the AllButOne (Given5) setup. Final results for MAE, precision and recall are always averaged over all users in the test set.

2.5.3 Comparison with Other Collaborative Filtering Methods

To compare the results of PMCF with other established collaborative filtering methods, we report results in terms of MAE, precision and recall for PMCF and for the following methods that have proven successful in the collaborative filtering literature.

- Memory-based collaborative filtering with Pearson correlation coefficient [RIS+94], one of the most popular memory-based collaborative filtering algorithms.

- Bayesian network collaborative filtering [BHK98]. Since we use exactly the same experimental setup and evaluation metrics for the EachMovie data as reported in [BHK98], we can directly compare the performance of Bayesian network collaborative filtering with other methods. We did not implement Bayesian network collaborative filtering for the Jester data.

- Naïve Bayesian collaborative filtering [MP00]. Despite its simplicity, the naïve Bayesian classifier has proven to be competitive with Pearson correlation collaborative filtering.

All methods are evaluated in the setup described in Sec. 2.5.2.

We compare the above listed methods with two variants of PMCF, which we label PMCF $\mathcal{P}$ and PMCF $\mathcal{D}$. For the PMCF $\mathcal{D}$ variant, we use the full training set to build the density model in Eq. (2.2), that is, the profile space is taken to be the full training data $\mathcal{P} = \mathcal{D}$. The other variant PMCF $\mathcal{P}$ is PMCF with profile space constructed from the training set $\mathcal{D}$ in the way described in Sec. 2.4. For both EachMovie and Jester, we constructed profile spaces with 1000 profiles (out of the training data of size 5000).

2.5.4 Evaluation of Accuracy

Tab. 2.1 and 2.2 summarize the performance of all evaluated collaborative filtering methods in terms of accuracy for prediction and recommendation.

Tab. 2.1 lists results for accuracy of prediction that are based on one particular split of the data into training and test set that has also been used in [BHK98]. It can be clearly seen that PMCF achieves an MAE that is about 7-8% lower than the MAE of the competing methods. The results also suggest that PMCF is particularly suitable for making predictions when only very little information about the active user is given: PMCF achieved a particularly high improvement of accuracy for the Given5 scenarios.

For the accuracy of predictions, we also evaluated all methods (except for the Bayesian network) with five different randomly drawn training and test sets of size 5000, and did a pairwise comparison of results using a paired t-test. The test confirmed that both variants of PMCF performed better than all of the competing method with a significance level of 99% or above. Comparing PMCF $\mathcal{P}$ and PMCF $\mathcal{D}$, we noted that both performed almost identical for the Given5 setups. For the two AllButOne setups, PMCF $\mathcal{D}$ achieved a slightly better performance.

The results for accuracy of recommendation listed in Tab. 2.1 are averages over five different random splits into training and test data, as described

above. The large advantage of PMCF in terms of accuracy of prediction does not fully carry over to the accuracy of recommendation. Still, a consistent and statistically significant gain in performance could be achieved. Precision and recall of PMCF are typically about 2-3% better than those of the competing methods. A larger performance gain was always achieved in the TOP5 setup. Again, a pairwise comparison of results in a paired t-test was conducted. Results for one of the two PMCF variants that are marked in bold in Tab. 2.1 are better than those of the two competing methods with a significance level of 95% or above. Similarly, results marked in italics achieve a significance level of 90% or above.

Overall, we could verify that our proposed probabilistic memory-based collaborative filtering framework achieves an accuracy that is comparable or superior to other approaches that have been proposed for collaborative filtering.

2.5.5 Evaluation of Profile Learning

In Sec. 2.3, we proposed an active learning approach to interactively learn user profiles. In this section we investigate the performance of this learning process in a series of experiments that simulate the interaction between users and the recommender system.

We use the training/test split described in Sec. 2.5.2. For each test user, ratings are randomly split into a set S of 30 items and the remaining items U. We assume that the test user initially has not rated any items, and we wish to infer his profile using the active learning approach. To obtain long learning curves, we restrict the test set to users who had rated at least 60 items. This leaves us with 972 and 1340 test users respectively for the EACHMOVIE and JESTER data sets.

The interactive sessions are simulated as follows: The recommender system selects the 10 most informative items[12] according to the criterion de-

[12]Query items might also be presented one by one, instead of using batches of 10 items. We chose the variant with 10 items since it seems more natural in an application scenario. Presenting items one by one can easily make users impatient.

Table 2.1: Accuracy of predictions, measured by mean absolute error MAE, of different collaborative filtering methods. Details on the individual experiments are given in Sec. 2.5.2 and 2.5.3. Both PMCF $\mathcal{P}$ and PMCF $\mathcal{D}$ consistently outperform the competing method, in particular when little information is given about the active user in the GIVEN5 scenario. The results shown here are based on the training/test split reported in Sec. 2.5.2. Additional experiments with 5 random splits and paired t-test confirmed that PMCF outperformed the competing methods at a significance level of 99% or above

	EACHMOVIE			
	ALL		EXTREME	
	ALLBUTONE	GIVEN5	ALLBUTONE	GIVEN5
Pearson correlation	0.996	1.150	1.130	1.290
Bayesian networks	1.066	1.154		
Naïve Bayes	0.987	1.162	1.096	1.223
PMCF $\mathcal{D}$	0.966	1.008	1.010	1.112
PMCF $\mathcal{P}$	0.984	1.008	1.040	1.110

	JESTER			
	ALL		EXTREME	
	ALLBUTONE	GIVEN5	ALLBUTONE	GIVEN5
Pearson correlation	3.927	4.258	5.062	5.730
Bayesian networks				
Naïve Bayes	4.132	4.263	4.753	5.512
PMCF $\mathcal{D}$	3.544	3.967	4.408	5.219
PMCF $\mathcal{P}$	3.724	3.972	4.523	5.464

Table 2.2: Accuracy of recommendations, measured by precision and recall, of different collaborative filtering methods. All results in this table are averaged over 5 runs, where training and test sets had been drawn at random from the total data sets. Marked in bold are PMCF results that are significantly better (with a significance level of 95% or above in a paired t-test) than the competing approaches. Marked in italic are PMCF results that are better than the competing approaches with a significance level of 90% or above. Further details on the individual experiments are given in Sec. 2.5.2 and 2.5.3

	EachMovie			
	Top5		Top10	
	Precision	Recall	Precision	Recall
Pearson correlation	0.703	0.284	0.656	0.510
Naïve Bayes	0.663	0.264	0.617	0.484
PMCF $\mathcal{D}$	**0.715**	**0.291**	**0.665**	**0.520**
PMCF $\mathcal{P}$	**0.713**	*0.288*	*0.659*	*0.512*

	Jester			
	Top5		Top10	
	Precision	Recall	Precision	Recall
Pearson correlation	0.703	0.284	0.656	0.510
Naïve Bayes	0.663	0.264	0.617	0.484
PMCF $\mathcal{D}$	**0.715**	**0.291**	**0.665**	**0.520**
PMCF $\mathcal{P}$	**0.713**	*0.288*	*0.659*	*0.512*

scribed in Sec. 2.3.4. User feedback is taken from the actual ratings the user has given on an item, if the item is in set U. Otherwise it is left unrated, simulating that the user is not able to give feedback on this particular item. We make a series of such simulated interactions, $t = 1, 2, \ldots$, gaining more and more knowledge about the user's profile. For test user a, we compute the MAE when predicting the ratings in set S and the precision for making recommendations in set S, denoted by MAE(a, t) and precision(a, t). By averaging over all users in the test set, we obtain MAE(t) and precision(t).

Using MAE and precision, we compare the following 5 methods for selecting the query items:

1. Query item selection by minimizing the entropy of the like-mindedness, as outlined in Sec. 2.3.4.

2. Uncertainty sampling, as described in Sec. 2.3.5

3. Entropy sampling, as described in Sec. 2.3.5

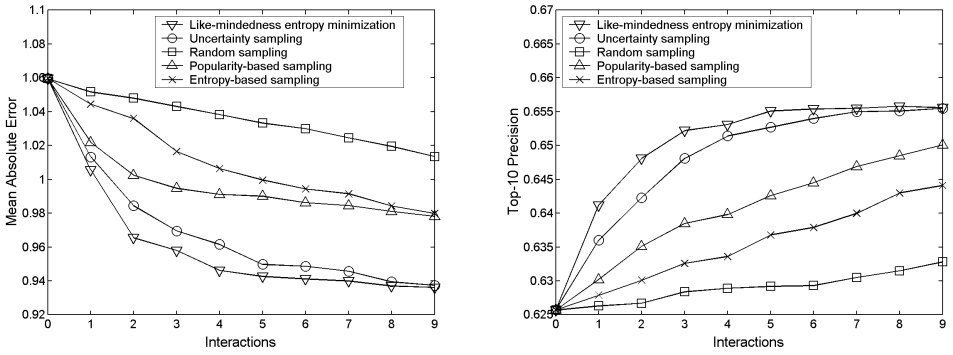
4. Popularity sampling: At each iteration, we present 10 of the most popular items to the test user

5. Random sampling: At each iteration t, we randomly select 10 query items

Methods 3, 4 and 5 have also been studied in [RAC+02].

The resulting learning curves MAE(t) and precision(t) for the above 5 methods are shown in Fig. 2.2 (for the EachMovie data) and in Fig. 2.3 (for Jester). The graphs clearly indicate that query item selection based on like-mindedness outperforms all other tested methods. Like-mindedness based selection is thus a method which achieves a maximum gain of information about a particular user with only a minimum of user effort.

For all of the tested methods, we also investigated the average number of items the user is being able to rate at a particular iteration t. The low performance of random and entropy based sampling, in particular on EachMovie, can be explained by the fact that users are not able to answer the posed queries. The remaining three methods all achieve similar results for the average number of rated items. Yet, like-mindedness sampling seems



(a) $MAE(t)$ (b) $Precision(t)$

Figure 2.2: Learning individual user profiles for the EachMovie data. Mean absolute error MAE(t) and precision(t) achieved after $t = 1, 2, \ldots$ steps of user interaction with different strategies for query item selection. Details of the experimental setup are given in Sec. 2.5.5

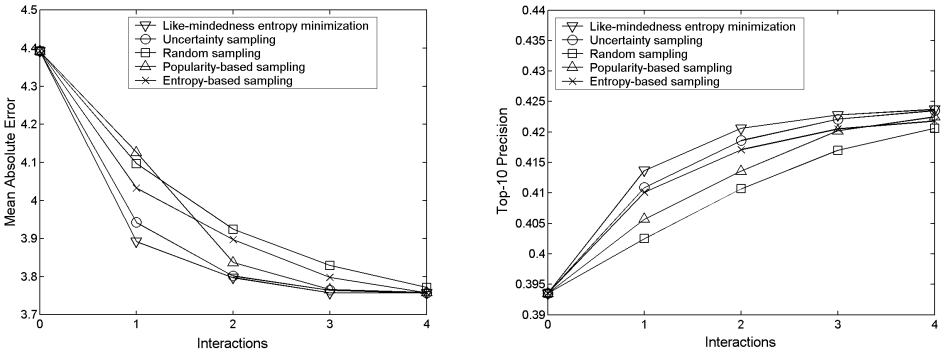
to ask more informative questions, leading to the steepest learning curves among all methods in Fig. 2.2 and 2.3.

From the presented results, we conclude that like-mindedness based sampling is a sensible and accurate method of inferring user profiles and requires only a minimum amount of user effort. It has a particularly good performance on data sets with high sparsity such as EachMovie, where only 3% of the items are rated, yet it also performs better than competing approaches on dense data sets (Jester).

2.5.6 Evaluation of Constructing Profile Spaces

We showed in Sec. 2.4 how a small profile space $\mathcal{P}$ for the PMCF model can be constructed out of a large data base of user ratings $\mathcal{D}$. In this section, we investigate how the profile space construction relates to the achievable accuracy for predictions and recommendations in the PMCF model.

To this aim, we use the split of training and test data described in



(a) $MAE(t)$ (b) $Precision(t)$

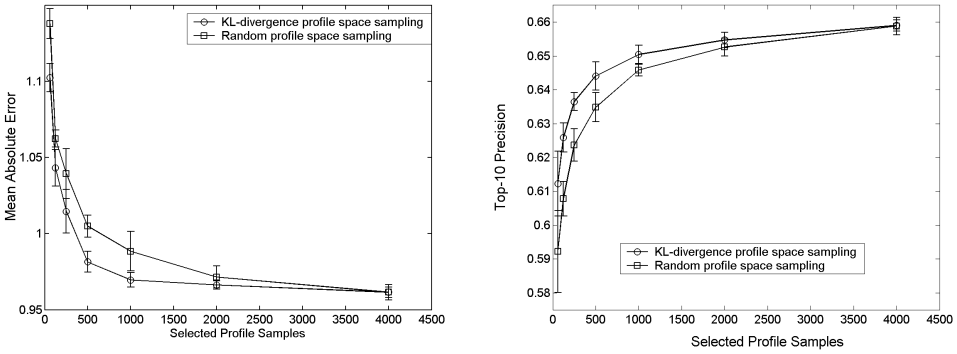
Figure 2.3: Learning individual user profiles for the JESTER data. Mean absolute error MAE(t) and precision(t) achieved after $t = 1, 2, \ldots$ steps of user interaction with different strategies for query item selection. Details of the experimental setup are given in Sec. 2.5.5

Sec. 2.5.2. From the training data $\mathcal{D}$, the profile space $\mathcal{P}$ is constructed iteratively as outlined in Sec. 2.4. At certain intervals[13], we evaluate the performance of the PMCF method, based on the profile space constructed so far, on the test set. We use the mean absolute error MAE in the ALLBUTONE setting and precision in the TOP10 setting as the measures of performance.

We so obtain a curve of performance versus size of the profile space. Since constructing the profile space uses a randomized strategy to select candidate profiles (see Sec. 2.4.3), we repeat this procedure 10 times. Thus, error bars for the performance of PMCF with a profile space of a given size can be plotted. As the baseline method, we use a PMCF model with a profile space drawn at random from the full training data $\mathcal{D}$.

The resulting curves for accuracy of prediction (MAE) and recommendation (precision) on the EACHMOVIE data are shown in Fig. 2.4, and in

[13]Evaluation is done when the profile space has reached a size of 60, 125, 250, 500, 1000, 2000 and 4000.



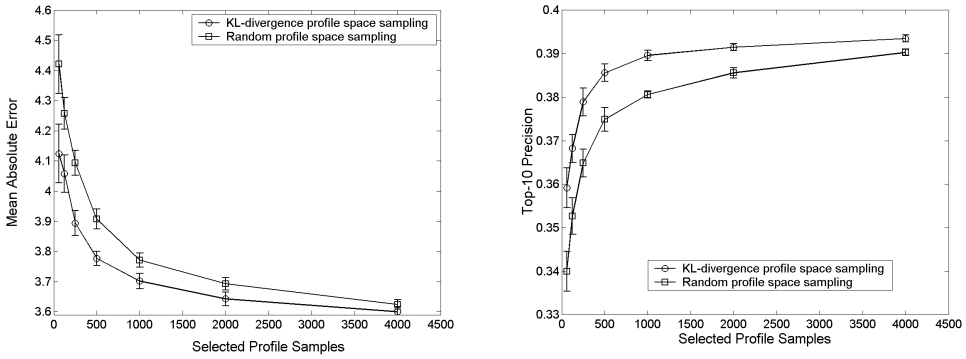
(a) $MAE(t)$ (b) $Precision(t)$

Figure 2.4: Evaluating the profile space construction for the EachMovie data set. Mean absolute error MAE and precision achieved with profile spaces of different size, that are either constructed based on KL-divergence (see Sec. 2.4) or drawn at random from the training data. The plot is averaged over 10 runs, with error bars

Fig. 2.5 for the Jester data. All plots clearly indicate that the profile space construction presented in Sec. 2.4 does bring significant advantages in terms of performance over a randomly chosen profile space. The gain in performance was particularly large for accuracy of recommendation on the Jester data.

2.6 Conclusions

In this chapter we proposed a probabilistic framework for memory-based collaborative filtering (PMCF). The PMCF is based on user profiles in a specially constructed profile space. With PMCF the posterior distribution of user ratings can be used to predict an active user's ratings. An experimental comparison with other collaborative filtering methods (memory-based collaborative filtering with Pearson correlation, Bayesian networks, naïve Bayes)



(a) $MAE(t)$ (b) $Precision(t)$

Figure 2.5: Evaluating the profile space construction for the JESTER data set. Mean absolute error MAE and precision achieved with profile spaces of different size, that are either constructed based on KL-divergence (see Sec. 2.4) or drawn at random from the training data. The plot is averaged over 10 runs, with error bars

showed that PMCF outperforms the competing methods both in terms of accuracy for prediction and recommendation.

As one of its major advantages, PMCF allows extensions to the basic model on a sound probabilistic basis. We showed in Sec. 2.3 how an active learning approach can be integrated smoothly into the PMCF framework. Through active learning, the collaborative filtering system can interactively learn about a new user's preferences, by presenting well selected query items to the user. Our results showed that the active learning approach performed better than other methods for learning user profiles, in the sense that it can make accurate predictions with only a minimum amount of user input.

In Sec. 2.4 we used the probabilistic framework to derive a data selection scheme that allows the recommender system to make fast and accurate predictions. Instead of operating on a possibly huge database of user preferences (as traditional memory-based collaborative filtering does), the data selection scheme allows us to use only a carefully selected subset, which we call the profile space. Using the so selected profile space in the PMCF model allows making fast predictions with only a small drop in performance over a PMCF model operating on the full data.

We believe that the PMCF framework will allow more extensions and thus can contribute to further improvements of recommender systems. A particularly promising research direction is the combination of collaborative filtering methods with content based filtering into hybrid systems. We are currently working on a PMCF based hybrid system for image and text retrieval [YST+03]. This system implicitly also solves the new item problem: If no user ratings are available for an item, predictions can still be made on the basis of the content description.

Our further work on the PMCF model will also include an improved model for user preferences. In Eq. (2.3), only items that were actually rated contribute to the model. An improved model could also take into account the information which items had not been rated. For example, in the EachMovie data, a movie may have been unrated because a friend had dissuaded the user from seeing the movie. Thus, one may be able to extract a certain degree of information from the set of unrated items as well and further improve the accuracy of a collaborative filtering system.

For the current PMCF system, as described in this article, the efficiency of the active learning scheme still needs to be improved. Active learning based on minimization of the entropy of like-mindedness achieves the best recommendation accuracy, yet the computational complexity is higher than that of competing methods such as uncertainty sampling.

Chapter 3

Content-Based Filter: A Generalized PCA Model

3.1 Introduction

Information retireval/filtering[1] systems typically operate on a database of information items, e.g. images, movies and articles, with each item described by many attributes. An image, for example, is associated with visual features (e.g. color, texture), categories, and introductory texts. As another example, a web page probably contains texts, images, user logs (implicitly), audio and video clips. Different types of attributes seem to be developed independently but actually *intrinsically related*. Organization and representation of heterogeneous attributes remain big research challenges, which can effectively influence a wide spectrum of problems like,

- Homogenizing the representation of heterogenous data
- Reducing dimensionality of attributes
- Highlighting latent relevant information
- Supporting fast and accurate retrieval/filtering

[1] In this chapter, we loose the distinguish between information retrieval and information filtering, since they are almost the same in the context of content-based framework (see Sec. 1.2.3).

The challenges generally arise in many application fields, including web information retrieval, multimedia databases, data mining, genetics and molecular biology, language modelling and machine translation.

Pattern recognition, machine learning and statistics communities have a long tradition to develop dimensionality reduction approaches for *continuous* data (e.g. principal component analysis (PCA), factor analysis). Recently similar models were applied to text data, which was referred as latent semantic analysis (LSA) or indexing (LSI). Studies indicate that LSA (or LSI) can reduce the dimensionality of document representations. The derived compact features summarize the dependencies between different terms and are often highly informative in characterizing the semantics of documents. Currently those related models mainly handle continuous data, lacking means to deal with non-continuous or mixed types of data.

This chapter will describe a different and novel paradigm, with emphasis on the *fusion* of as wide as possible information sources in content-based information filtering. As different attributes describe the information items from different perspectives, putting them all together is likely to add more relevant information in processing. This expansion, however, also makes it harder to handle the information due to the higher dimensionality and mixed types of attributes (i.e. continuous, binary and categorical). Clearly we need a dimensionality reduction algorithm dealing with vectors with heterogenous attributes. Traditionally few dimension reduction methods can do the job. Here we develop a new dimensionality reduction model, called *generalized probabilistic principle component analysis* (GPPCA), to transform heterogeneous attributes into reduced and unifying *continuous Gaussian variables*. Thus the model is not only dimensionality reduction, but also *feature fusion*. In addition to obvious benefits for computational concerns (as typical algorithms only easily work with low-dimension and continuous data), the reduced features are often highly indicative, as indicated by Vapnik,

> *if one can significantly compress the description of the given string, then the algorithm used describes intrinsic properties of the data.*

We will empirically demonstrate that the extraction of latent variables from heterogenous attributes can significantly improve the accuracy of content-

based retrieval/filtering on painting data.

The rest of this chapter will be organized as follows. Sec. 3.2 gives an overview on the research of latent variable analysis. Then in Sec. 3.3 we will describe our model of GPPCA and derive an efficient variational expectation-maximization (EM) algorithm to learn the model from data. we also discuss properties of the model and connections to previous work. Empirical studies, based on movie data and image data, are given in Sec. 3.4. At the end, we end up this chapter by some conclusive discussions in Sec. 3.5.

3.2 Latent Variable Analysis

We can often assume that high-dimensional observations are *indirect measurements* arising from one or several underlying *source(s)*, which typically cannot be directly measured. Latent variable analysis (also called hidden variable analysis) is a family of *unsupervised* data modelling approaches that factorizes high-dimensional observations with a reduced set of latent variables. The latent variables offer explanations of the dependencies between observed variables and often explore the underlying sources.

This section will review some major latent variable models, including factor analysis (FA) and principal component analysis (PCA). The focus is not on a comprehensive coverage of the whole field, but on highlighting the the mechanism of using reduced latent variables to explain high dimensional data.

3.2.1 Factor Analysis

Factor analysis (FA) is a classical technique developed in the statistical literature that aims to identify latent uncorrelated continuous variables, which are assumed to be *Gaussian distributed*.

In order to find a representation for the distribution of observed data $\boldsymbol{t}$ in an M-dimensional space $\boldsymbol{t} = (t_1, \ldots, t_M)$ in terms of L ($L < M$) latent

variables (or factors) $\boldsymbol{x} = (x_1, \ldots, x_L)$, FA has the form

$$\begin{aligned} t_1 &= w_{11}x_1 + \cdots + w_{1L}x_L + b_1 + \varepsilon_1 \\ t_2 &= w_{21}x_1 + \cdots + w_{2L}x_L + b_2 + \varepsilon_2 \\ \vdots & \qquad\qquad \vdots \\ t_M &= w_{M1}x_1 + \cdots + w_{ML}x_L + b_M + \varepsilon_M \end{aligned} \tag{3.1}$$

or $\boldsymbol{t} = \boldsymbol{W}\boldsymbol{x} + \boldsymbol{b} + \boldsymbol{\varepsilon}$, where $\boldsymbol{W}$ is an $L \times M$ matrix of *factor loadings*, the ε_m are uncorrelated zero-mean Gaussian noises, and the b_m are constants accounting for the mean of observations $\boldsymbol{t}$. Assuming $\boldsymbol{x}$ are a priori Gaussian distributed with zero mean and unit covariance matrix, we can write down the generative process of observations as

$$\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{x}; \boldsymbol{0}, \boldsymbol{I}) \tag{3.2}$$

$$\boldsymbol{t}|\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{t}; \boldsymbol{W}^\top \boldsymbol{x} + \boldsymbol{b}, \boldsymbol{C}_\varepsilon) \tag{3.3}$$

where $\boldsymbol{C}_\varepsilon = \mathrm{diag}[\mathrm{Var}(\varepsilon_1), \ldots, \mathrm{Var}(\varepsilon_M)]$. FA assumes that observed multivariate data $\boldsymbol{t}$ are generated from some independent[2] zero-mean Gaussian sources $\boldsymbol{x}$ by linear transformation $\boldsymbol{W}$ (i.e. rotation and stretching of the coordinates), plus observation noises ε and bias of mean $\boldsymbol{b}$. The covariance of observations is therefore written as

$$\boldsymbol{C}_t = \boldsymbol{W}^\top \boldsymbol{W} + \boldsymbol{C}_\varepsilon \tag{3.4}$$

$\boldsymbol{C}_\varepsilon$ being diagonal indicates that ε_m are unique to each t_m and explain the uncorrelated variation between t_ms. Thus latent sources account for the *correlations* of coordinates of observations and $\boldsymbol{W}$ explains the correlations. Given multivariate $\boldsymbol{t}$, typically an expectation-maximization (EM) algorithm is applied to learn the model. Fig. 3.1 offers a graphical interpretation to FA.

3.2.2 Principal Component Analysis

Principal Component Analysis (PCA) has been proven successful in dimensionality reduction for many pattern recognition applications. For a set of N

[2]Variables being Gaussian and uncorrelated are statistically independent.

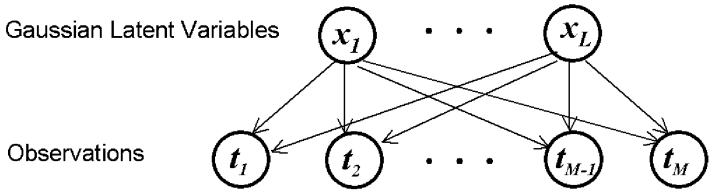


Figure 3.1: A graphical interpretation to factor analysis

zero-mean random observation vectors $\boldsymbol{t}_n \in \mathbb{R}^M, n = 1, \ldots, N$, PCA diagonalizes the covariance matrix [3]

$$\boldsymbol{C}_t = \frac{1}{N} \sum \boldsymbol{t}_n \boldsymbol{t}_n^\top \tag{3.5}$$

To do this, one has to solve the eigen-decomposition problem:

$$\begin{aligned} \boldsymbol{C}_t &= \sum_m \lambda_m \boldsymbol{v}_m \boldsymbol{v}_m^\top \\ &= \boldsymbol{V} \Lambda \boldsymbol{V}^\top \end{aligned} \tag{3.6}$$

where Λ is a diagonal matrix $\mathrm{diag}[\lambda_1, \ldots, \lambda_M]$ and $\boldsymbol{v}_m$ (i.e. each column of $\boldsymbol{V}$) is an eigenvector. PCA picks up the L leading eigenvectors with largest eigenvalues as *principle components*. It can be easily demonstrated that an eigenvalue is the variance of data distribution over its corresponding eigenvector (principal component). That is to say, PCA identifies the principal subspace where data distribution has the largest variance. An illustration of PCA on two-dimensional data is in Fig. 3.2.

Recent studies on PCA reveals that it has strong connections with statistical FA but takes a slightly different modelling assumption, where the measurement noises are assumed to be isotropic, i.e. having equal variance σ^2 [TB99, RG99]. Thus the generative model is

$$\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{x}; \boldsymbol{0}, \boldsymbol{I}) \tag{3.7}$$

$$\boldsymbol{t}|\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{x}; \boldsymbol{W}^\top \boldsymbol{x} + \boldsymbol{b}, \sigma^2 \boldsymbol{I}) \tag{3.8}$$

and the covariance matrix of $\boldsymbol{t}$ is

$$\boldsymbol{C}_t = \boldsymbol{W}^\top \boldsymbol{W} + \sigma^2 \boldsymbol{I} \tag{3.9}$$

[3] More precisely, the covariance matrix is the expectation of $\boldsymbol{t}\boldsymbol{t}^\top$

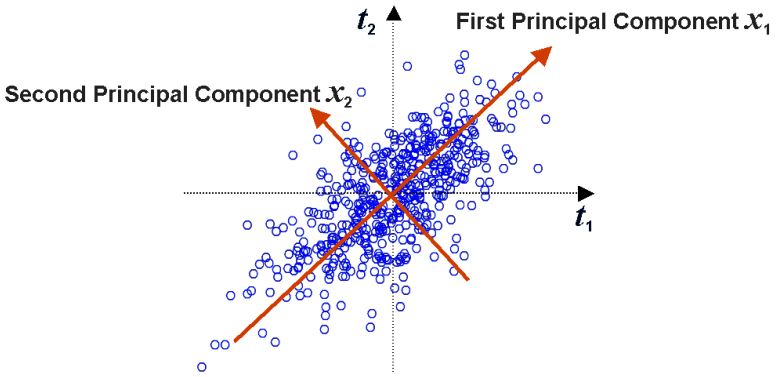


Figure 3.2: An illustration of PCA on two-dimensional data

It turns out that the maximum likelihood estimate of the factor loading matrix $\boldsymbol{W}$ in this case is given by the first L principle eigenvectors of the observation covariance matrix $\boldsymbol{C}_t$, with scaling determined by the eigenvalues and noise variances [TB99]. Classical PCA Eq. (3.7) can be obtained by assuming $\sigma \to 0$ in this latent-variable framework.

3.3 A Generalized Probabilistic PCA Model

We will present a probabilistic latent-variable model to fit observations with both continuous and binary attributes in this section. Since in practice categorical attributes can always be effectively encoded by sets of binary attributes (e.g. 1-of-c coding scheme [Bis95]), this model can be applied to a wide range of situations. We call the model as generalized probabilistic PCA (GPPCA).

3.3.1 Latent-Variable Modeling Mixed Types of Data

The goal of a latent variable model is to find a representation for the distribution $p(\boldsymbol{t})$ of observed data in an M-dimensional space $\boldsymbol{t} = (t_1, \ldots, t_M)$ in terms of a number of L latent variables $\boldsymbol{x} = (x_1, \ldots, x_L)$. In our setting of interest, we consider a *joint distribution* of M continuous and binary

attributes, as illustrated in Fig. 3.4. We use $m \in \mathcal{R}$ to indicate that the variable t_m is continuous-valued, and $m \in \mathcal{B}$ for binary variables (i.e. $\{0, 1\}$). The generative model is:

$$\begin{aligned} \boldsymbol{x} &\sim \mathcal{N}(\boldsymbol{x}; \mathbf{0}, \boldsymbol{I}) && (3.10)\\ \boldsymbol{y}|\boldsymbol{x} &= \boldsymbol{W}^T\boldsymbol{x} + \boldsymbol{b} && (3.11)\\ t_m|y_m &\sim \mathcal{N}(t_m; y_m, \sigma^2) \qquad m \in \mathcal{R} && (3.12)\\ t_m|y_m &\sim \mathrm{Be}\big(g(y_m)\big) \qquad m \in \mathcal{B} && (3.13) \end{aligned}$$

By $\mathrm{Be}(p)$ we denote a Bernoulli distribution with parameter p (the probability of giving a 1). $\boldsymbol{W}$ is an $L \times M$ matrix with column vectors $(\boldsymbol{w}_1, \ldots, \boldsymbol{w}_M)$, $\boldsymbol{b}$ an M-dimensional column vector, and $g(y_m)$ the sigmoid function (Also see Fig. 3.3)

$$p(t_m = 1) = g(y_m) = \frac{1}{(1 + \exp(-y_m))} \tag{3.14}$$

which models a Bernoulli distribution's dependence on a linear continuous quantity as a generalized linear model [HTF01]. We assume that the observed vectors $\boldsymbol{t}$ are generated from a prior Gaussian distribution with zero mean and identity covariance matrix[4]. Note that we assume a common noise variance σ^2 for all continuous variables. To match this assumption, we sometimes need to use scaling or whitening as a pre-processing step for the continuous data in our experiments.

The likelihood[5] of an observation vector $\boldsymbol{t}$ given the latent variables $\boldsymbol{x}$ and model parameters θ is

$$\begin{aligned} p(\boldsymbol{t}|\boldsymbol{x}, \theta) &= p(\boldsymbol{t}^{\mathcal{R}}|\boldsymbol{x}, \theta)p(\boldsymbol{t}^{\mathcal{B}}|\boldsymbol{x}, \theta) \\ &= \prod_{m \in \mathcal{R}} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\Big\{ -\frac{1}{2}\frac{(y_m - t_m)^2}{\sigma^2} \Big\} \prod_{m \in \mathcal{B}} g\big((2t_m - 1)y_m\big) \end{aligned} \tag{3.15}$$

[4] A non-zero mean and non-identity covariance matrix can be moved to parameters $\boldsymbol{W}$ and $\boldsymbol{b}$ without loss of generality.

[5] A full Bayesian treatment would require prior distributions for the parameters θ. We do not go for a full Bayesian solution here, thus implicitly assuming a non-informative prior.

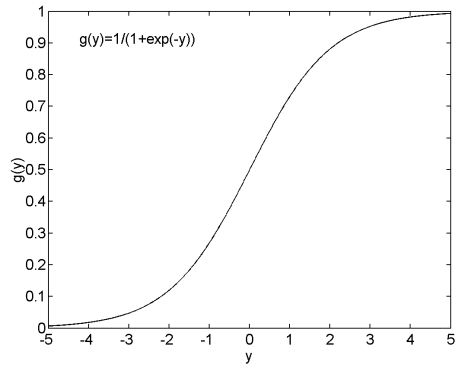


Figure 3.3: Sigmoid function

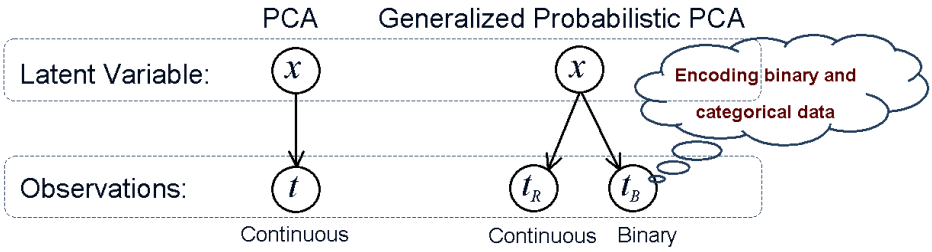


Figure 3.4: Graphical models of PCA and generalized probabilistic PCA

where $y_m = \boldsymbol{w}_m^T \boldsymbol{x} + b_m$. The distribution in $\boldsymbol{t}$-space, for a given value of θ is then obtained by integration over the latent variables $\boldsymbol{x}$

$$p(\boldsymbol{t}|\theta) = \int p(\boldsymbol{t}|\boldsymbol{x}, \theta) p(\boldsymbol{x}) d\boldsymbol{x} \tag{3.16}$$

3.3.2 Maximum-Likelihood Model Fitting

For a given set of N observation vectors, the log likelihood of data D is

$$\mathcal{L}(\theta) = \log p(D|\theta) = \sum_{n=1}^{N} \log p(\boldsymbol{t}_n|\theta) \tag{3.17}$$

We estimate the model parameters $\theta = \{\boldsymbol{W}, \boldsymbol{b}, \sigma^2\}$ by maximizing the log likelihood $\mathcal{L}(\theta)$, which can be typically achieved by the expectation-maximization (EM) algorithm [DLR77]. It starts from a random guess of θ and then iteratively repeats the E-step and M-step. Each iteration increases the likelihood of data.

E-step:

At the expectation step, we estimate the *a posteriori* distribution of hidden variables given the current estimate of model parameters.

$$p(\boldsymbol{x}|\boldsymbol{t}, \theta^k) = \frac{p(\boldsymbol{t}|\boldsymbol{x}, \theta^k) p(\boldsymbol{x})}{\int p(\boldsymbol{t}|\boldsymbol{x}, \theta^k) p(\boldsymbol{x}) d\boldsymbol{x}} \tag{3.18}$$

M-step:

At the maximization step, we update the estimate of parameters by maximizing the expected likelihood of observations over the estimated *a posteriori* distribution of hidden variables from the last E-step.

$$\theta^{k+1} = \arg\max_{\theta} \int p(\boldsymbol{t}|\boldsymbol{x}, \theta) p(\boldsymbol{x}|\boldsymbol{t}, \theta^k) d\boldsymbol{x} \tag{3.19}$$

However, given parameters θ^k estimated from the previous M-step, the posterior distribution $p(\boldsymbol{x}|\boldsymbol{t}, \theta^k)$ in E-step can not be analytically solved due to the integral. We thus have to resort to an approximated solution like

MCMC (Markov Chain Monte-Carlo) sampling [Fis96], which draws S random samples from $p(\boldsymbol{x}|\boldsymbol{t}, \theta^k)$. Then the integral at M-step can be approximated by a sum of predictive likelihood $p(\boldsymbol{t}|\boldsymbol{x}, \theta)$ evaluated at S random samples, with the complexity increased by S times. MCMC demonstrates good performance in many cases, however, it introduces a rather high computational cost. In the next section, we will present an analytical approximation to solve this problem.

3.3.3 A Variational EM Algorithm for Model Fitting

In order to obtain the parameters θ that maximize Eq. (3.17), we employ a *variational EM algorithm*[JGJS99]. A variational EM algorithm constructs a lower bound (the variational approximation) for the likelihood of observations, Eq. (3.17), by first introducing additional variational parameters $\boldsymbol{\psi}$. Then, it iteratively maximizes the lower bound with respective to the variational parameters (at the E-step) and the parameters θ of interest (at the M-step).

By adopting a variational lower bound previously used for Bayesian logistic regression [JJ00], an approximation for the likelihood contribution of binary variables, $t_m \in \mathcal{B}$ in Eq. (3.13) is given by

$$\begin{aligned} p(t_m|\boldsymbol{x}, \theta) &\geq \widetilde{p}(t_m|\boldsymbol{x}, \theta, \psi_m) \\ &= g(\psi_m)\exp\big\{(A_m - \psi_m)/2 + \lambda(\psi_m)(A_m^2 - \psi_m^2)\big\} \end{aligned} \quad (3.20)$$

where $A_m = (2t_m - 1)(\boldsymbol{w}_m^T\boldsymbol{x} + b_m)$ and $\lambda(\psi_m) = [0.5 - g(\psi_m)]/2\psi_m$. For a fixed value of $\boldsymbol{x}$, we get the perfect approximation where the lower bound is maximized to be $p(t_m|\boldsymbol{x}, \theta)$ by setting $\psi_m = A_m$. However, in the case of $\boldsymbol{x}$ distributed over a Gaussian distribution $\mathcal{N}(0, \boldsymbol{I})$, maximization of the corresponding lower bound with respect to ψ_m is not so straightforward. In the following we derive the variational approximation for the log likelihood Eq. (3.17) of data D as the following

$$\mathcal{L}(\theta) \geq \mathcal{F}(\theta, \Psi) = \log \prod_{n=1}^{N} \int \widetilde{p}(\boldsymbol{t}_n|\boldsymbol{x}, \theta, \boldsymbol{\psi}_n) p(\boldsymbol{x}) d\boldsymbol{x} \quad (3.21)$$

where

$$\widetilde{p}(\boldsymbol{t}_n|\boldsymbol{x},\theta,\boldsymbol{\psi}_n) = \prod_{m\in\mathcal{R}} p(t_{mn}|\boldsymbol{x},\boldsymbol{w}_m,\sigma) \prod_{m\in\mathcal{B}} \widetilde{p}(t_{mn}|\boldsymbol{x},\boldsymbol{w}_m,\psi_{mn}) \tag{3.22}$$

We denote the total set of $N \times |\mathcal{B}|$ variational parameters by Ψ. Since the variational approximation depends on $\boldsymbol{x}$ only quadratically in the exponent and the prior $p(\boldsymbol{x})$ is Gaussian, the integrals to obtain the approximation $\mathcal{F}(\theta,\Psi)$ can be solved in closed form.

The variational EM algorithm starts with an initial guess of θ and then iteratively maximizes $\mathcal{F}(\theta,\Psi)$ with respect to Ψ (E-step) and θ (M-step), respectively, holding the other fixed. Each iteration increases the lower bound, but will not necessary maximize the true log likelihood $\mathcal{L}(\theta)$. However, since the E-step results a very close approximation of $\mathcal{L}(\theta)$, we expect that, at M-step, the true log likelihood is increased. Details are given in the following:

E-step: $\Psi^{k+1} \leftarrow \arg\max_{\Psi} \mathcal{F}(\theta^k,\Psi)$

The optimization can be achieved by a normal EM approach. Given $\boldsymbol{\psi}_n^{\text{old}}$ updated from the previous step, the algorithm iteratively estimates the sufficient statistics for the posterior approximation $\tilde{p}(\boldsymbol{x}_n|\boldsymbol{t}_n,\theta^k,\boldsymbol{\psi}_n^{\text{old}})$[6], which is again a Gaussian with covariance and mean given by

$$\boldsymbol{C}_n = \big[\frac{1}{\sigma^2}\sum_{m\in\mathcal{R}} \boldsymbol{w}_m\boldsymbol{w}_m^T + \boldsymbol{I} - 2\sum_{m\in\mathcal{B}} \lambda(\psi_{mn}^{\text{old}})\boldsymbol{w}_m\boldsymbol{w}_m^T\big]^{-1} \tag{3.23}$$

$$\boldsymbol{\mu}_n = C_n\Big\{\frac{1}{\sigma^2}\sum_{m\in\mathcal{R}}(t_{mn}-b_m)\boldsymbol{w}_m + \sum_{m\in\mathcal{B}}\big[\frac{2t_{mn}-1}{2} + 2b_m\lambda(\psi_{mn}^{\text{old}})\big]\boldsymbol{w}_m\Big\} \tag{3.24}$$

and then updates $\boldsymbol{\psi}_n$ by maximizing $E_n\big\{\log\widetilde{p}(\boldsymbol{t}_n,\boldsymbol{x}_n|\theta^k,\boldsymbol{\psi}_n)\big\}$ where the expectation is with respect to $\tilde{p}(\boldsymbol{x}_n|\boldsymbol{t}_n,\theta^k,\boldsymbol{\psi}_n^{\text{old}})$. Taking the derivative of $E_n\big\{\log\widetilde{p}(\boldsymbol{t}_n,\boldsymbol{x}_n|\theta^k,\boldsymbol{\psi}_n)\big\}$ with respect to $\boldsymbol{\psi}_n$ and setting it to zero leads to the updates

$$\psi_{mn}^2 = E_n\big\{(\boldsymbol{w}_m^T\boldsymbol{x}_n + b_m)^2\big\} = \boldsymbol{w}_m^T E_n(\boldsymbol{x}_n\boldsymbol{x}_n^T)\boldsymbol{w}_m + 2b_m\boldsymbol{w}_m^T E_n(\boldsymbol{x}_n) + b_m^2 \tag{3.25}$$

[6]Based on Bayes' rule, the posterior approximation is derived by normalizing $\widetilde{p}(\boldsymbol{t}_n|\boldsymbol{x}_n,\theta^k,\boldsymbol{\psi}_n^{\text{old}})p(\boldsymbol{x}_n)$ and thus is a proper density, no longer a lower bound.

where $E_n(\boldsymbol{x}_n\boldsymbol{x}_n^T) = \boldsymbol{C}_n + \boldsymbol{\mu}_n\boldsymbol{\mu}_n^T$ and $E_n(\boldsymbol{x}_n) = \boldsymbol{\mu}_n$. The two-stage optimization updates ψ and monotonously increases $\mathcal{F}(\theta^k, \Psi)$. The experiments showed that this procedure converges rapidly, most often in only two steps.

M-step: $\theta^{k+1} \leftarrow \arg\max_\theta \mathcal{F}(\theta, \Psi^{k+1})$

Similar to the former E-step, this can also be achieved by iteratively first estimating the sufficient statistics of $\tilde{p}(\boldsymbol{x}_n|\boldsymbol{t}_n, \theta^{\text{old}}, \boldsymbol{\psi}_n^{k+1})$ through Eq. (3.23) and Eq. (3.24), and then maximizing $\sum_{n=1}^N E_n\big\{\log\widetilde{p}(\boldsymbol{t}_n, \boldsymbol{x}_n|\theta, \boldsymbol{\psi}_n^{k+1})\big\}$ with respect to θ, where $E_n(\cdot)$ denotes the expectation over $\tilde{p}(\boldsymbol{x}_n|\boldsymbol{t}_n, \theta^{\text{old}}, \boldsymbol{\psi}_n^{k+1})$. For $m \in \mathcal{R}$, we derive the following updates

$$\boldsymbol{w}_m^T = \Big[\sum_{n=1}^N (t_{mn} - b_m)E_n(\boldsymbol{x}_n)^T\Big]\Big[\sum_{n=1}^N E_n(\boldsymbol{x}_n\boldsymbol{x}_n^T)\Big]^{-1} \tag{3.26}$$

$$\sigma^2 = \frac{1}{N|\mathcal{R}|}\sum_{n=1}^N\Big\{\sum_{m\in\mathcal{R}}\big[\boldsymbol{w}_m^T E_n(\boldsymbol{x}_n\boldsymbol{x}_n^T)\boldsymbol{w}_m + 2(b_m - t_{mn})\boldsymbol{w}_m^T E_n(\boldsymbol{x}_n) + (b_m - t_{mn})^2\big]\Big\} \tag{3.27}$$

where b_m, $m \in \mathcal{R}$, is directly estimated by the mean of t_{mn}. For $m \in \mathcal{B}$, we have the following updates

$$(\boldsymbol{w}_m^T, b_m)^T = -\Big[\sum_{n=1}^N 2\lambda(\psi_{mn})E_n(\widehat{\boldsymbol{x}}_n\widehat{\boldsymbol{x}}_n^T)\Big]^{-1}\Big[\sum_{n=1}^N (t_{mn} - 0.5)E_n(\widehat{\boldsymbol{x}}_n)\Big] \tag{3.28}$$

where $\widehat{\boldsymbol{x}}_n = (\boldsymbol{x}^T, 1)^T$.

3.3.4 Inference with Complete and Incomplete Observations

Finally, given the trained generative model, we can infer the *a posteriori* distribution of hidden variables for a complete observation vector $\boldsymbol{t}$ by using Bayes' rule

$$p(\boldsymbol{x}|\boldsymbol{t}, \theta) = \frac{p(\boldsymbol{t}|\boldsymbol{x}, \theta)p(\boldsymbol{x})}{\int p(\boldsymbol{t}|\boldsymbol{x}, \theta)p(\boldsymbol{x})d\boldsymbol{x}} \tag{3.29}$$

However, since the integral is again infeasible, we need to derive a variational approximation by normalizing $\widetilde{p}(\boldsymbol{t}|\boldsymbol{x},\theta,\boldsymbol{\psi})p(\boldsymbol{x})$, where $\boldsymbol{\psi}$ is obtained by maximizing the lower bound $\tilde{p}(\boldsymbol{t}|\theta,\boldsymbol{\psi})$, as described by Eq. (3.23), Eq. (3.24), and Eq. (3.25) in the E-step of variational EM inference. The optimization goes on with model parameters θ fixed and converges very fast, normally whithin 2 iterations.

For a vector $\boldsymbol{t}$ with some missing attributes, we can still infer the posterior distribution with the described model, since essentially the attributes are independent given model parameters θ and latent variables $\boldsymbol{x}$. Let $\boldsymbol{t}^* \in \boldsymbol{t}$ denote the non-missing attributes, we can infer the posterior distribution of latent variables

$$p(\boldsymbol{x}|\boldsymbol{t}^*,\theta) = \frac{\prod p(t^*|\boldsymbol{x},\theta)p(\boldsymbol{x})}{\int \prod p(t^*|\boldsymbol{x},\theta)p(\boldsymbol{x})d\boldsymbol{x}} \tag{3.30}$$

Again, the inference resorts to an iterative optimization process with respect to variational parameters $\boldsymbol{\psi}$: starting from a random initialization of $\boldsymbol{\psi}$, and interatively first estimating the covariance and mean of the posterior distribution of hidden variables given the visible attributes $\boldsymbol{t}^*$

$$\boldsymbol{C} = \big[\frac{1}{\sigma^2}\sum_{m\in\mathcal{R}^*}\boldsymbol{w}_m\boldsymbol{w}_m^T + \boldsymbol{I} - 2\sum_{m\in\mathcal{B}^*}\lambda(\psi_m^{\text{old}})\boldsymbol{w}_m\boldsymbol{w}_m^T\big]^{-1} \tag{3.31}$$

$$\boldsymbol{\mu} = C\Big\{\frac{1}{\sigma^2}\sum_{m\in\mathcal{R}^*}(t_m - b_m)\boldsymbol{w}_m + \sum_{m\in\mathcal{B}^*}\big[\frac{2t_m-1}{2} + 2b_m\lambda(\psi_m^{\text{old}})\big]\boldsymbol{w}_m\Big\} \tag{3.32}$$

where $\mathcal{R}^*$ and $\mathcal{B}^*$ respectively denote the non-missing continous and non-missing binary attributes, and then updating variational parameters

$$\psi_m^2 = E\big\{(\boldsymbol{w}_m^T\boldsymbol{x} + b_m)^2\big\} = \boldsymbol{w}_m^T E(\boldsymbol{x}\boldsymbol{x}^T)\boldsymbol{w}_m + 2b_m\boldsymbol{w}_m^T E(\boldsymbol{x}) + b_m^2 \tag{3.33}$$

Obviously, with missing data, the uncertainty of latent varibles $\boldsymbol{x}$ is likely to be increased. At the end, we can use the estimated mean of $\boldsymbol{x}$ as the intrinsic representation of observations. Finally, we would note that the ability of handling missing data is one of the big advantages of probabilistic latent-variable models, while traditional dimensionality-reduction methods solve the problem in an ad-hoc way, just filling in default values. In the application of information filtering, there might be many missing data due to varous reasons. In our experiment we associate paintings with user logs, which typically have a large portion of missing values since each user only visited a subset of paintings.

3.3.5 Unbalanced Inference

We can extend GPPCA to the case of *unbalanced inference*, meaning that sometimes we wish the model fitting adapted to only a *subset* of attributes. Putting the problem in a general setting, let $\boldsymbol{t}_u$ be ordainary attributes (continous or discrete) and $\boldsymbol{t}_v$ be the important attributes that we wish to adapt to. The learning problem can be solved in two steps:

- Maximizing the likelihood $p(\boldsymbol{t}_v|\theta_v)$ via the variational EM algorithm, and deriving the sufficient statistics of the *a poseriori* distributions of latent variables $p(\boldsymbol{x}|\boldsymbol{t}_v, \theta_v)$ for each samples $\boldsymbol{t}_v$.

- Using only the M-step to estimate parameters θ_u by maximizing the likelihood $p(\boldsymbol{t}_u|\boldsymbol{x}, \theta_u)$ expected over $p(\boldsymbol{x}|\boldsymbol{t}_v, \theta_v)$.

This approach takes into account the coviarance structure of $\boldsymbol{t}_v$ and the covariance between $\boldsymbol{t}_v$ and $\boldsymbol{t}_u$, while ignoring the covariance of $\boldsymbol{t}_u$.

For a better understanding of the problem, let us think about a content-based image retrieval problem using low-level visual features and texts. It makes sense to assume that visual features are pretty weak while texts are always relevant to people's searching intentions. Thus we can train the model with higher emphasis on the texts based on some images with mannully annotated texts. Since in the trained model the mapping from low-level features to latent space has been 'forced' by textual information, thus the resultant model is likely to project images without textual information into semantically meaningful latent spaces. This is a nice property because textual annotation costs many human efforts.

Let us consider another problem called *supervised dimensionality reduction*, in which given some labelled training examples, GPPCA treats the target labels (or values) as additional attributes $\boldsymbol{t}_v$ and adapts the derived latent varibles $\boldsymbol{t}$ accounting for targets. After the model is learned, given a new input vector with targets being missing, with parameters θ_u we can transform the input into a meaningful latent space.

3.3.6 Properties of GPPCA

Multivariate Data Analysis, Fusion and Visualization

The rows of the ML estimator $\boldsymbol{W}$ that relates latent variables to observed variables span the principal subspace of the data. The GPPCA model allows a unified probabilistic modelling of continuous, binary and categorical observations, which can bring great benefits in real-world data analysis. Also, it can serve as a visualization tool for high-dimensional mixed data in a two-dimensional latent variable space. Existing models currently only visualize either continuous [BSW98] or binary data [Tip99]. Also, like PPCA [TB99], GPPCA specifies a full generative model, it can also handle missing observations in a principled way.

For pattern recognition tasks, GPPCA can provide a principled data transformation for general learning algorithms (which most often rely on continuous inputs) to handle data with mixed types of attributes. In information filtering, the advantage is even more apparent. Suppose that we are building a web-based image search system. A common way to characterize and index images is to perform PCA on high-dimensional visual measures (e.g. color, texture, shape and segments). The proposed generalized GPPCA can do this by taking into account not only the visual features, but also image categories, textual descriptions, and even user visit logs. This scenario has the potential power of finding the projections of images that actually reflect the semantics and high-level perceptual properties.

Supervised Dimensionality Reduction

Also, GPPCA can provide a principled approach to *supervised* dimensionality reduction, by allowing the target values as additional observation variables. GPPCA explores the dependence between inputs and targets via the hidden variables and maximizes the joint likelihood of both. It actually discovers a subspace of the joint space in which the projections of inputs have small projection loss and also have clear class distributions. A large number of methods have been developed to handle issue of supervised data reduction (see [HTF01]), like partial least squares, linear discriminant analysis (LDA).

However most of them, in general, can not handle missing data. Moreover, for a problem with c classes, LDA only finds a $c-1$ independent projective directions. For problems with few classes and high input dimensionality this may result in a reduction of dimensionality that is too drastic (see [Bis95]).

Relation to Previous Work

Jaakkola & Jordan [JJ00] proposed a variational likelihood approximation for Bayesian logistic regression, and briefly pointed out that the same approximation can be applied to learn the "dual problem", i.e. a hidden-variable model for binary observations. Tipping [Tip99] derived the detailed variational EM formalism to learn the model and used it to visualize high-dimensional binary data. Collins *et al.* [CDS01] generalized PCA to various loss functions from the exponential family, in which the case of Bernoulli variables is similar to Tipping's model. Latent variable models for mixed observation variables were also studied in statistics community [Mou96, SRL97]. In contrast to our variational approach, [Mou96] and [SRL97] used numerical integration methods to handle the otherwise intractable integral in the EM algorithm. Latent variable models for mixed data were already mentioned by Bishop [BSW98] and Tipping [Tip99], yet never explicitly implemented. Recently, Cohn [Coh03] proposed *informed projections*, a version of supervised PCA, that minimizes both projection loss and inner-class dissimilarities. However, this requires tuning a parameter β to weight the two parts of the loss function. To our best knowledge, little work has been done in information retrieval and filtering that mergeing heterogenous descriptive attributes into unifying continuous features.

3.4 Empirical Study

3.4.1 A Toy Problem

We first illustrate GPPCA on a simple problem, where 100 two-dimensional samples are generated from two Gaussian distributions with mean $[-1, 1]$ and

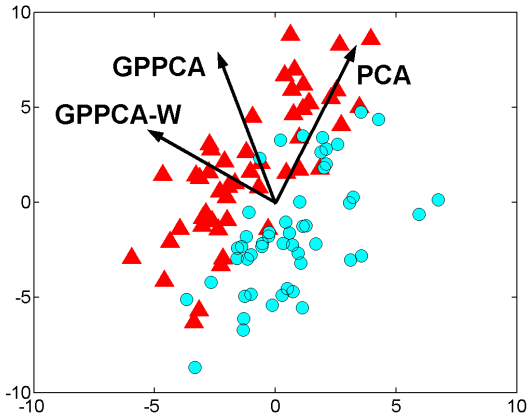
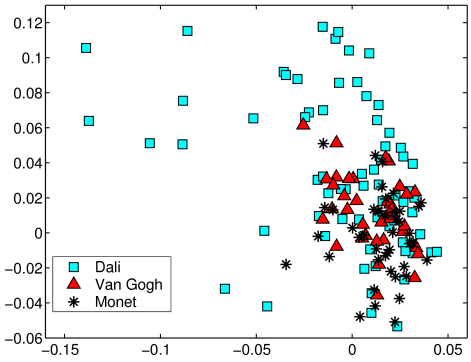


Figure 3.5: A toy problem: PCA, GPPCA and GPPCA-W solutions

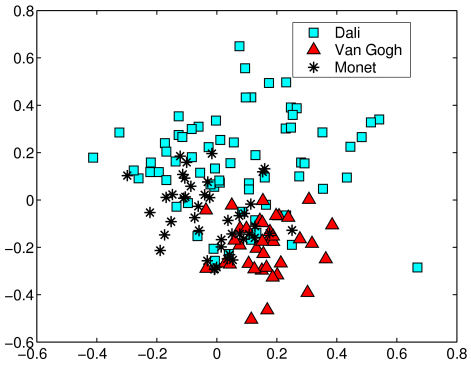
$[1, -1]$ respectively and equal covariance matrices. A third binary variable was added that indicates which Gaussian the sample belongs to. We perform GPPCA, as described in Sec. 3.3, and standard PCA on the data to identify the principal subspace. The results are illustrated in Fig. 3.5. As expected, the PCA solution is along the direction of largest variance. The GPPCA solution, on the other hand, also takes the class labels into account, and finds a solution that conveys more information about the observations. In an additional experiment, we pre-process the continuous variables with whitening and then perform GPPCA. We will refer to this as GPPCA-W in the following. With GPPCA-W, the solution even more clearly indicates the class distribution. Clearly, a change of the subspace in $\boldsymbol{W}$ corresponding to the whitened continuous variables will no longer change the likelihood contribution. Thus, the GPPCA EM algorithm will focus on the likelihood of binary observations only and thus lead to a result with clear class distribution.

3.4.2 Visualization of Painting Images

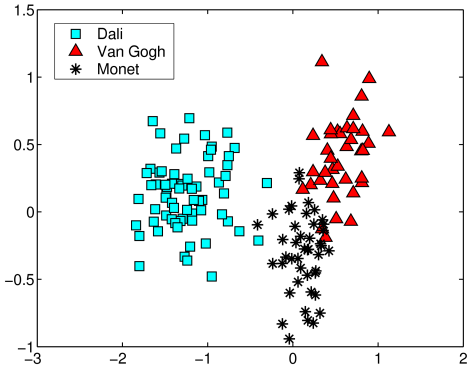
Next, we show an application of GPPCA to visualizing image data. We consider a data set of 642 painting images from 47 artists. An often encountered problem in the research on image retrieval is that low-level visual features



(a) PCA solution

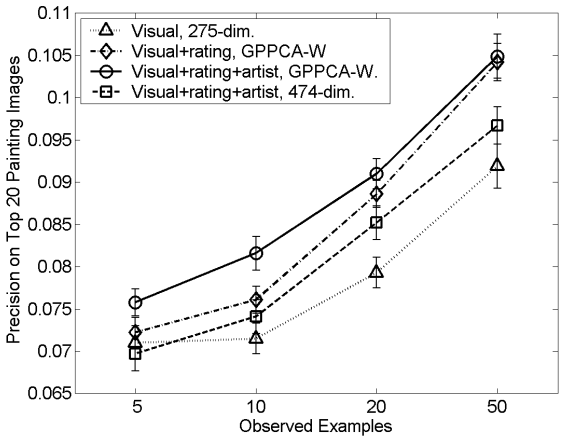


(b) GPPCA solution

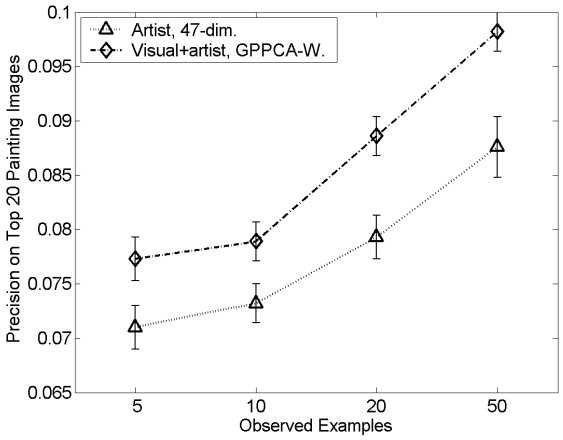


(c) GPPCA-W solution

Figure 3.6: Visualization of painting images



(a)



(b)

Figure 3.7: Precision on painting image recommendation, based on different features

(like color, texture, and edges) can hardly capture high-level information of images, like concept, style, etc. GPPCA allows to characterize images by more information than just those low-level features. In the experiment, we examine if it is possible to visualize different styles of painting images in a 2-dimensional space by incorporating the information about artists.

As the continuous data describing the images, we extract 275 low-level features (correlagram, wavelet texture, and color moment) for each image. We encode the artists in 47 binary attributes via a 1-of-c scheme, and obtain a 322-dimensional vector with mixed data for each image. The result of projecting this data to a 2-dimensional latent space is shown in Fig. 3.6, where we limit the shown data to the images of 3 particular artists.

The solution given by normal PCA does not allow a clear separation of artists. In contrast, the GPPCA solution, in particular when performing an additional whitening pre-processing for the continuous features, shows a very clear separation of artists. Note furthermore, that the distinction between Van Gogh and Monet is a bit fuzzy here—these artists do indeed share similarities in their style, in particular brush stroke, which is reflected by texture features.

3.4.3 Recommendation of Painting Images

Due to the deficiency of low-level visual features, building recommender systems for painting image is a challenging task. Here we will demonstrate that GPPCA allows a principled way of deriving compact and highly informative features. Thus the accuracy of recommender systems based on the new image features can be significantly improved.

We use the same set of 642 painting images as in the previous section. 190 users' ratings (like, dislike, or no rated) were collected through an online survey [7]. For each image, we combine visual features (275-dim.), artist (47-dim.), and a set of M *advisory* users' ratings on it (M-dim.) to form an $(322 + M)$-dimensional feature vector. This feature vector contains continuous, binary and missing data (because on average each user only rated 89

[7] http://honolulu.dbs.informatik.uni-muenchen.de:8080/paintings/index.jsp

images). We apply GPPCA to map the features to a reduced 50-dimensional feature space. The rest of $190 - M$ users are then treated as test users. For each test user, we hide some of his/her ratings and assume that only 5, 10, 20, or 50 ratings are observed. We skip one particular case if a user has not given that many ratings. Then we use the rated examples, in form of input (image features) – output(ratings) pairs, to train an RBF-SVM model to predict the user's ratings on unseen images and make a ranking. The performance of recommendation is evaluated by the top-20 precision, which is the fraction of actually liked images among the top-20 recommendations. We equally divide the 190 users into 5 groups, pick one group as the group of test users and treat the other 152 users as advisory users. For each tested case, we randomize 10 times and calculate the mean and error bars. The results are shown in Fig. 3.7.

Fig. 3.7(a) shows that GPPCA improves the precision in all the cases by effectively incorporating richer information. This is not surprising since the information about artists is a good indicator of painting styles. Advisory users' opinions on a painting actually reflect some high-level properties of the painting from a different individual's perspective. GPPCA here provides a princpled way to represent different information sources into a unified form of continuous data, and allows accurate recommendations based on the reduced data. Interestingly, as shown in Fig. 3.7(a), a recommender system working with direct combination of the three aspects of information shows a much lower precision than the compact form of features. This indicates that GPPCA effectively detects the 'signal subspace' of high dimensional mixed data, while eliminating irrelevant information. Note that there are over 80 percent of missing data in the user ratings. GPPCA also provides an effective means to handle this problem. Fig. 3.7(b) shows that GPPCA incorporating visual features and artist information significantly outperforms a recommender sytem that only works on artist information. This indicates that GPPCA working on the pre-whitened continuous data does not remove the influence of visual features.

3.5 Conclusions

This section describes generalized probabilistic PCA (GPPCA), a latent-variable model for mixed types of data, with continuous and binary observations. By adopting a variational approximation, an EM algorithm can be formulated that allows an efficient learning of the model parameters from data. The model generalizes probabilistic PCA and opens new perspectives for multivariate data analysis and machine learning tasks. For content-based filtering, mixed sources of attributes can be merged into low-dimensional continuous feature vectors, which can be easily processed by most of the existing content-based filtering algorithms. Since the derived data account for the dependence of original content features and thus often reflect the semantics of items, the quality of filtering can also be improved.

We demonstrated the advantages of the proposed GPPCA model on toy data and data from painting images. GPPCA allows an effective visualization of data in two-dimensional hidden space that takes into account both information from low-level image features and artist information. Our experiments on an image retrieval task show that the model provides a principled solution to incorporating different information sources, thus significantly improving the achievable precision. Currently the described model reveals the linear principal subspace for mixed high dimensional data. It might be interesting to pursue non-linear hidden variable model to handle mixed types of data. Also, how to deceide the number of hidden variables is also an open question.

Chapter 4

Hybrid Filter: A Hierarchical Bayesian Model

4.1 Introduction

Content-based filtering (CBF) and collaborative filtering (CF) represent the two major information filtering technologies. CBF systems analyze the contents of a set of items, together with the ratings provided by an individual user (called the "active user"), to infer which of the yet unseen items might be of interest for the active user. Examples include [BS97, MR00, PMB96, YT04, XYT+03, XXY+03]. In contrast, collaborative filtering methods [RIS+94, SM95, BP98, YST+04, YXEK03, YXS+02, YXT+02, YXEK01, YWXE01] typically accumulate a database of item ratings cast by a large set of users. The prediction of ratings for the active user is solely based on the ratings provided by all other users. These techniques do not rely on a description of item content.

One major difficulty in designing CBF systems lies in the problem of formalizing human perception and preferences based on content analysis. There is a large gap between low-level content features (visual, auditory, or others) and high-level user interests (like or dislike a painting or music). Fortunately, the information on personal preferences and interests are all carried in (explicit or implicit) user ratings. Thus CF systems can make use of these high level features rather easily, by combining the ratings of other like-minded users.

On the other hand, pure CF only relies on user preferences, without incorporating the actual content of items. CF often suffers from the extreme sparsity of available data, in the sense that users typically rate only very few items, thus making it difficult to compare the interests of two users. Furthermore, pure CF can not handle items for which no user has previously given a rating. Such cases are easily handled in CBF systems, which can make predictions based on the content of the new item.

4.1.1 Recent Work on Hybrid Filtering

Therefore, recently several hybrid approaches have been proposed to compensate the drawbacks of each method. The key challenge is how to *smoothly* combine the two types of filters. However, almost all the existing approaches did it in a heuristic or ad-hoc way. They may works well in some cases, but can hardly be applicable everywhere due to lacking of deep insights to the problem.

A family of approaches, e.g. [Paz99, CGM+99], treat content-based filtering and collaborative filtering separately and present the weighted average of both predictions as the final outcome. As another example, Fab [BS97] is a distributed implementation of a hybrid system. It maintains user profiles based on content analysis, and directly compare these profiles to determine similar users for collaborative filtering. An item is recommended to a user both when it scores highly against the user's own profile, and when it is also rated highly by users with similar profile. This family of methods combine two basic filtering approaches in a straightforward way. It needs heuristics to balance the weights of two parts. More seriously, they completely ignores the interaction between content effect and social effect and thus are not likely to achieve the optimal performance.

Basu et al [BHC98] proposed a classification approach that extends content-based filtering based on not only normal descriptive content features but also collaborative features (i.e. , other users' ratings on items). In this paradigm, difficulty comes from the extreme sparsity of collaborative features. Also the classifier need to be able to handle heterogenous input data (Interestingly, our proposed GPPCA in Ch. 3 is a way to solve the two problems.).

In another approach, a different combination strategy was taken in *content-boosted collaborative filtering* [MMN02], where content-based filters are built for each user and then applied to reduce the sparsity by generating *pseudo ratings* for non-rated items. The augmented user rating data is used to feed a collaborative filtering algorithm. It is yet unclear how much we should trust the the pseudo ratings. Obviously treating it equally with true ratings is not a good choice.

There are only few examples of a unifying framework for these two basic information filtering ideas, one being the three-way aspect model [PUPL01], which builds a probabilistic generative model assuming that both terms (i.e. textual content features) and user logs are generated from some hidden variables. This approach, however, is only applicable to text data and suffers from the sparsity of data. Our recent work [YST+03, YMT+03] made a step forward by suggesting a hierarchical Bayesian model. In this chapter we describe a nonparametric theoretical framework [YTY04] that generalizes our previous work.

4.1.2 Overview of Our Work

This chapter introduces the idea of *nonparametric hierarchical Bayesian modelling* to information filtering. The framework provides a deeper understanding on the *nature* of information filtering and leads to a principled hybrid filtering algorithm, which is general, simple and intuitively interpretable.

The framework assumes that each user's observed preferences data (i.e. annotations) are generated based on the user's own profile model, which itself is a random sample from a prior distribution of user profiles, shared by all the users and thus called the *common prior* in this paper. In this hierarchical Bayesian model each user's model is constrained by the common prior, through which the user is "communicating" with others.

The common prior is "learned" based on the observed annotations from a population of users. One may assume a parametric form (e.g. a Gaussian) for the common prior, and then estimate the associated parameters (e.g. the mean and variance in the Gaussian case). However, due to the complexity of the functional form of the learned prior, the true distribution of profiling

models can hardly be described by any known parametric distributions (e.g. a Gaussian). As a solution, this paper relaxes the parametric limitation on the common prior and adopts a nonparametric form—an infinite dimensional multinomial distribution—which itself is generated from a *Dirichlet Process* [EW95]. This model encompasses that, *a priori*, a new user may follow other users' interests, but may also have his/her own unique interests. The process enables a learning session for a new user to inherit knowledge from the sessions of other users, which leads to quite meaningful results for information filtering.

In the learning phase, typical Bayesian inference requires MCMC (Monte Carlo Markov Chain) sampling that is computationally expensive. This paper instead introduces novel approximations to learn the common prior effectively and efficiently. For a new user, the learned common prior is easily integrated into the prediction making in a Bayesian manner.

Finally, we would like to emphasize that the proposed work not only presents a novel and *principled* hybrid information filtering algorithm, but is also a quite *general framework* for information filtering and retrieval, since: (1) It unifies the CF and CBF in a single framework, where pure CF and pure CBF are special cases under certain circumstances; (2) Various existing algorithms combining CF and CBF can now be interpreted from a unified point of view, and their further improvements are also suggested; (3) Since CBF has its roots in information retrieval, the proposed work is also applicable to information retrieval, enabling retrieval sessions to inherit knowledge from each other. (4) The framework makes no requirements for the form of profiling models, it is thus applicable to a very wide range of user modelling applications in information filtering and retrieval (e.g. hidden Markov model for modelling user web browsing, and support vector machines for image retrieval).

4.1.3 Structure of This Chapter

The rest of this chapter is organized as follows. In Sec. 4.2, we will intensively explain the idea of modelling information needs of many users in a hierarchical Bayesian framework. Then we will introduce a nonparametric

solution to learn the prior distribution using Dirichlet process in Sec. 4.2.3. The realization of the theoretical framework using support vector machines will be presented in Sec. 4.5. In Sec. 4.6 we report results for applying the described algorithm to three data sets including images and texts. We end by giving conclusions and an outlook to future work in Sec. 4.7.

4.2 Modelling Information Needs via Hierarchical Bayes

In the following, we assume a set of M items, each item j being represented by a vector of features x_j, $j = 1, \ldots, M$. Also, we have annotation data for N different users. Annotation data for user i consists of a set of rated items $\mathcal{R}_i$, together with a set of ratings $\{y_{ij}\}, j \in \mathcal{R}_i$, where each rating y_{ij} is either $+1$ (liked that particular item) or -1 (disliked)[1]. The overall annotation data for user i, $i = 1, \ldots, N$ is denoted by $\mathcal{D}_i = \{(x_j, y_{i,j}) \mid j \in \mathcal{R}_i,\ y_{i,j} \in \{+1, -1\}\}$.

4.2.1 Non-Hierarchical Bayesian Models

Given observations $\mathcal{D}_i$ from user i, a statistical content-based approach will normally learn a predictive model, represented by parameters θ_i, by the maximum-likelihood (ML) principle,

$$\theta_i^{ML} = \arg\max_{\theta} p(\mathcal{D}_i|\theta). \tag{4.1}$$

Once the ML estimate θ_i^{ML} is achieved, $\mathcal{D}_i$ can be thrown away and predictions are made by $p(y|x, \theta_i^{ML})$, meaning distribution of user i's rating y on some item, described by a vector of features x, given profile model parameters θ_i^{ML}.

In contrast, a Bayesian approach will introduce a prior distribution $p(\theta)$, indicating, before observing something, a favor over different settings of

[1]We restrict the discussion here to models for binary annotation data. But this restriction can be released.

model parameters θ. Then the prediction is made by absorbing the uncertainty of model parameters:

$$p(y|x, \mathcal{D}_i) = \int_\theta p(y|x, \theta) p(\theta|\mathcal{D}_i) d\theta, \tag{4.2}$$

where the *a posteriori* distribution of θ is calculated by Bayes rule

$$p(\theta|\mathcal{D}_i) = \frac{p(\mathcal{D}_i|\theta)p(\theta)}{p(\mathcal{D}_i)}. \tag{4.3}$$

Here we can see the difference that traditional approach learns the ML estimate of model parameters θ_i^{ML} based on Eq. (4.1) and then use the model to make predictions $p(y|x, \theta_i^{ML})$, while Bayesian approach takes into account the uncertainty of models and makes predictions solely based on training data $p(y|x, \mathcal{D}_i)$.

When sufficient data are observed, then $p(\theta|\mathcal{D}_i)$ is likely to be governed by the likelihood $p(\mathcal{D}_i|\theta)$, becoming a sharp peak located at the ML estimate θ_i^{ML}. Thus $p(\theta|\mathcal{D}_i)$ can be approximated by a delta function $\delta(\theta_i^{ML})$. In this case Bayesian approaches Eq. (4.2) are equivalent to non-Bayesian approaches.

However, in information filtering applications we can hardly require a user to annotate many items. Given a small amount of data, the uncertainty of model parameter is very high and we can not say that the point estimate of θ is dominant in $p(\theta|\mathcal{D}_i)$. Thus the Bayesian solution with integral over the θ space can be viewed as a way of averaging predictions made by infinite number of settings of θ. This strategy can effectively prevent over-fitting.

It is necessary to note that, the prior distribution $p(\theta)$ plays an important role here, which reflects our prior knowledge about the domain and often prefers low-complexity models. The Bayesian rule Eq. (4.3) actually makes a trade-off between empirical knowledge conveyed by observations and the prior knowledge. When observations are limited, like the information filtering case, Bayesian rule makes $p(\theta|\mathcal{D}_i)$ more influenced by prior knowledge. However, if the user annotates more items, then the impact of empirical (training) data will be automatically increased. Thus Bayesian rule is a quite natural way to integrate our prior knowledge into the learning process.

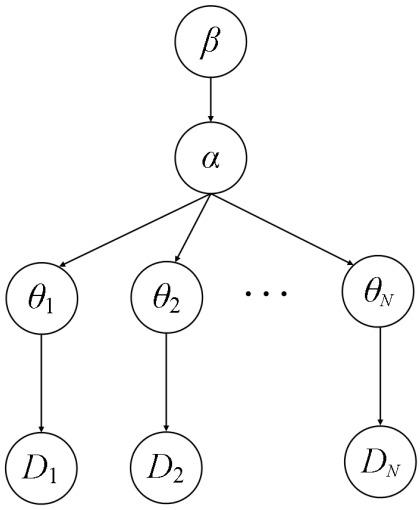


Figure 4.1: An illustration of the described hierarchical Bayesian model for information filtering

4.2.2 Hierarchical Bayesian Models

In normal Bayesian models, the prior distribution $p(\theta)$ is specified to reflect our prior belief. In this section we are going to answer two questions concerning $p(\theta)$, can we learn the prior from data, instead of specifying it before observing the data? Does it make sense to do so in information filtering? Our answers to them are both, YES.

We should keep in mind that information filtering systems are modelling a *population* of individuals, whose annotation data are different to each other but related in some way. The relations can be characterized by exploring the *structure of the data generation process*. It can be achieved in a quite natural way if we use a *common prior distribution*, from which each θ_i for user i is a random sample generated. Then the overall observations $\mathcal{D} = \{\mathcal{D}_1, \ldots, \mathcal{D}_N\}$ are modelled hierarchically, with each user's data $\mathcal{D}_i$ distributed conditionally on parameters θ_i, which themselves are distributed conditionally on the common prior distribution $p(\theta|\alpha)$ characterized by *hyperparameters* α. This hierarchical data generative process is described in Fig. 4.1. Then the likeli-

hood of overall observed annotations can be written as

$$p(\mathcal{D}|\alpha) = \prod_i \int_{\theta_i} p(\mathcal{D}_i|\theta_i)p(\theta_i|\alpha)d\theta_i, \tag{4.4}$$

where

$$p(\mathcal{D}_i|\theta_i) = \prod_{j\in\mathcal{R}_i} p(y_{i,j}, x_j|\theta_i) = \prod_{j\in\mathcal{R}_i} p(y_{i,j}|x_j, \theta_i)p(x_j), \tag{4.5}$$

in which we assume that the picking-up of items is random, independent of user profiles, and the ratings for picked items depends on user profiles.

With such a hierarchical thinking, we can learn the common prior distribution $p(\theta|\alpha)$ from observations of a population, even when samples of θ are not directly visible. It is again solved in a Bayesian manner, by specifying the prior distributions $p(\alpha|\beta)$ of hyperparameters β,

$$p(\alpha|\mathcal{D}, \beta) = \frac{p(\mathcal{D}|\alpha)p(\alpha|\beta)}{\int_\alpha p(\mathcal{D}|\alpha)p(\alpha|\beta)d\alpha}. \tag{4.6}$$

In rest of this chapter, we call $p(\alpha|\beta)$ the *hyper prior distribution*. By marginalizing out the uncertainty of α, one can learn the common prior distribution of profile model parameters θ as

$$p(\theta|\mathcal{D}, \beta) = \int p(\theta|\alpha)p(\alpha|\mathcal{D}, \beta)d\theta. \tag{4.7}$$

Then predictions for an active user a are made by

$$\begin{aligned} p(y|x, \mathcal{D}_a, \mathcal{D}, \beta) &= \int_\theta p(y|x, \theta)p(\theta|\mathcal{D}_a, \mathcal{D}, \beta)d\theta \\ &= \int_\theta p(y|x, \theta)\frac{p(\mathcal{D}_a|\theta)p(\theta|\mathcal{D}, \beta)}{\int_\theta p(\mathcal{D}_a|\theta)p(\theta|\mathcal{D}, \beta)d\theta}d\theta \\ &= \int_\theta p(y|x, \theta)\frac{p(\mathcal{D}_a|\theta)\int_\alpha p(\theta|\alpha)p(\alpha|\mathcal{D}, \beta)d\alpha}{\int_\theta p(\mathcal{D}_a|\theta)\int_\alpha p(\theta|\alpha)p(\alpha|\mathcal{D}, \beta)d\alpha d\theta}d\theta. \end{aligned} \tag{4.8}$$

Comparing Eq. (4.2) and Eq. (4.8), we can see that now the predictions for user a not only depend on his/her own existing data $\mathcal{D}_a$, but also depend on other users data $\mathcal{D}$. With the hierarchical thinking, our solution, with its root from the conventional idea of content-based filtering, is interestingly approaching the idea of collaborative filtering!

Finally, a fully Bayesian hierarchical generative process of data is described as the following

1. For the whole population of users, generate a sample of hyperparameters $\alpha \sim p(\alpha|\beta)$,

2. For each user $i = 1, \ldots, N$, generate a sample $\theta_i \sim p(\theta|\alpha)$,

3. Given θ_i and a set of randomly chosen items $\mathcal{R}_i$, generating annotations $y_{i,j} \sim p(y|x_j, \theta_i)$, for $j \in \mathcal{R}_i$.

Nonhierarchical models described in Sec. 4.2.1 treat users separately, which is a conventional way of content-based information filtering. The overall observed annotations are thus modelled by many independent parameters $\theta_i, i = 1, \ldots, N$, with the size scaled by the number of users N. Generally speaking, models with many free parameters may be too flexible and thus likely to overfit the data. In contrast, a hierarchical Bayesian model also apply so many parameters $\theta_i, i = 1, \ldots, N$ (thus with sufficient flexibilities to fit the data), but meanwhile put dependence into these parameters by using the common prior distribution $p(\theta|\alpha)$. In this way $p(\theta|\alpha)$ constrains the freedom of parameters θ_i and effectively avoid overfitting.

It is also necessary to comment the highest prior distribution $p(\alpha|\beta)$. Previously we introduce the common prior distribution $p(\theta|\alpha)$ with the aims to prevent overfitting in the sense that the learned model is able to predict well for user i's unseen items. However, we should also prevent another kind of overfitting that the learned common prior distribution $p(\theta|\alpha)$ only perfectly fit existing users while performs badly on future coming users. Thus we should use $p(\alpha|\beta)$ to constrain the adaptation of α.

4.2.3 Nonparametric Hierarchical Models

One may first specify some parametric form for the common prior $p(\theta|\alpha)$ and then learn the parameters α (or their posterior distribution using Eq. (4.6)). However, due to the nature of the problem[2], the common prior is normally complex and can hardly be described by any known parametric form (like a

[2](1) Profiling models must be tailored to applications, like hidden Markov model for web browsing or support vector machines for image retrieval; (2) The distribution of people's interests are very complex.

Gaussian). We thus relax the parametric assumption and introduce a highly flexible *nonparametric* common prior:

$$p(\theta|\alpha) = \sum_{l=1}^{\infty} \alpha_l \delta_{\theta_l}(\theta), \tag{4.9}$$

where $\delta_{\theta_l}(\theta)$ is the point mass distribution at θ_l (i.e. $\delta_{\theta_l}(\theta) = 0$ if $\theta \neq \theta_l$, and $\int \delta_{\theta_l}(\theta) d\theta = 1$). α are the parameters of an *infinite multinomial distribution*. For a better understanding, one can imagine that the space of θ is equally divided into K regions. For a region indexed by l, the corresponding mass probability is α_l. When $K \to \infty$, a large class of countably infinite distribution can be represented as Eq. (4.9). This way is called *nonparametric* in the sense that the distribution is described in a straightforward way and no simplifying parametric assumption is made.

To complete the model, we now define the hyper prior $p(\alpha|\beta)$ as a *Dirichlet process* [Ant74, EW95]:

$$\alpha|\beta \sim \mathrm{DP}(M, \alpha^*), \tag{4.10}$$

where $\beta = \{M, \alpha^*\}$, M is a nonnegative scalar, called *concentration parameter*, and α^* is called the *base distribution*, which is actually the mode of α. Dirichlet process is a generalization of Dirichlet prior (i.e. the conjugate prior for finite multinomial distribution) to the case of infinite dimensions (See the details about Dirichlet prior in [Hec95]). Intuitively, a Dirichlet process defines statistically how faraway a randomly generated distribution α differs from the base distribution α^*. The larger M is, the more likely α is close to α^*.

If we have directly observed the realizations of profile models $\theta_1, \ldots, \theta_N$, then the posterior distribution of α is again a Dirichlet process. By integrating over α, the common prior (i.e. the distribution of the next coming θ) becomes

$$p(\theta|\{\theta_i\}_{i=1}^N, \beta) \quad = \quad \frac{M p(\theta|\alpha^*) + \sum_{i=1}^N \delta_{\theta_i}(\theta)}{M + N}. \tag{4.11}$$

From the above equation we can see a very important property of a Dirichlet process. For an intuitive understanding, let us imagine a process of assigning persons into interest clubs. Suppose there are potentially infinite number of clubs, then

- The first person comes and creates a club θ_1 based on the base distribution $p(\theta|\alpha^*)$;
- The second person may either follow the first person to join in the same club with probability $1/(M+1)$, or create a new club from the distribution $p(\theta|\alpha^*)$ with probability $M/(M+1)$;
- As the process going on, N persons have chosen their own clubs $\{\theta_i\}_{i=1}^N$. Then a new person will join in a club by either following previous persons based on the distribution $\frac{1}{N}\sum_{i=1}^N \delta_{\theta_i}(\theta)$ with probability $N/(M+N)$, or creating a new club from distribution $p(\theta|\alpha^*)$ probability $M/(M+N)$.

The process is also known as a case of *Chinese restaurant process* in statistical literature [BGJT04]. In the process, a new user has a large chance to create a new club if M is very large (or to follow previous users when M is small). Thus in this paper, the hyper prior $p(\alpha|\beta)$—a Dirichlet process—reflects our prior knowledge about how strongly users are influenced by each other.

4.3 Learning the Nonparametric Hierarchical Model

However, in the application of information filtering, user profile models $\{\theta_i\}_{i=1}^N$ are not directly visible, but only associated annotations $\mathcal{D} = \{\mathcal{D}_i\}_{i=1}^N$ observed. Then the dependence among users, described by the common prior, should not only reflect our prior knowledge (i.e. the Dirichlet process), but also be adapted to empirical data $\mathcal{D}$. Thus the basic learning procedure is to first (1) estimate the common prior $p(\theta|\mathcal{D},\beta)$, and then (2) integrate it into individual profiling sessions. This section will introduce the details of our learning solution.

The integral over α is often infeasible in computing the common prior in Eq. (4.7). However, Markov Chain Monte Carlo (MCMC) approaches (like Gibbs sampling) allow us to directly sample θ. Here we briefly describe a Gibbs sampling procedure, considering the posterior $p(\theta|\mathcal{D}_i, M, \alpha^*)$:

1. for each user $i = 1, \ldots, N$, sample a $\theta_i \sim p(\theta|\mathcal{D}_i, \{\theta_j, i \neq j\}, M, \alpha^*)$, by either picking up an existing value θ_l^* with conditional probability:

$$b\frac{N^*_{-i,l}}{N-1+M}p(\mathcal{D}_i|\theta_l^*) \tag{4.12}$$

where $N^*_{-i,l}$ is occurrence of θ_l^* in all the users except i, or taking a new value from $p(\theta|\mathcal{D}_i, \alpha^*)$ with probability

$$b\frac{M}{N-1+M}\int p(\mathcal{D}_i|\theta)d\alpha^*(\theta) \tag{4.13}$$

where b is a normalizing term.

2. for each distinguished value $l = 1, \ldots, L$, sample $\theta_l^* \sim p(\theta_l^*|c_l, \alpha^*)$, $l = 1, \ldots, L$, where c_l are the set of users who are associated to the same θ_l^* at the last step. This step actually updates the distinguished values θ_l^* in a batch way.

The sampling proceeds iteratively over above steps until convergence, which ends up with a small number of values θ_l^*, somehow reflecting the clustering structure of user profiles.

MCMC sampling is often computationally expensive and sometimes technically difficult. This chapter instead goes for the *maximum a posteriori* (MAP) estimate of α:

$$\alpha^{MAP} = \arg\max_{\alpha} p(\alpha|\mathcal{D}, \beta) \tag{4.14}$$

and obtains the common prior as $p(\theta|\alpha^{MAP})$. Treating θ as latent variables, we apply expectation-maximization (EM) algorithm to estimate α^{MAP} as follows. At E-step, we re-estimate the posterior distribution of θ for each user, based on α^{k-1} achieved at the $(k-1)$-th step of EM:

$$p(\theta|\mathcal{D}_i, \alpha^{k-1}) = \frac{p(\mathcal{D}_i|\theta)p(\theta|\alpha^{k-1})}{p(\mathcal{D}_i|\alpha^{k-1})} \tag{4.15}$$

In the M-step, we re-estimate the parameters α of the common prior using the learning rule of multinomial (see [Hec95]), which is equivalent to update the common prior as

$$p(\theta|\alpha^k) = \frac{Mp(\theta|\alpha^*) + \sum_{i=1}^{N} p(\theta|\mathcal{D}_i, \alpha^{k-1})}{M+N}. \tag{4.16}$$

At the very beginning, we initialize α^0 to be the base distribution α^* and perform the described E-step and M-step iteratively. At the convergence we achieve $\alpha^{MAP} = \alpha^k$, and derive the learned common prior as follows:

$$p(\theta|\mathcal{D}, \beta) \approx p(\theta|\alpha^{MAP}) \tag{4.17}$$

By plugging the learned common prior into Eq. (4.8), we are able to predict users' information needs as follows

$$p(y|x, \mathcal{D}_a; \alpha^{MAP}) = \int_\theta p(y|x, \theta) p(\theta|\mathcal{D}_a; \alpha^{MAP}) d\theta \tag{4.18}$$

However, $p(\theta|\alpha^{MAP})$ can not be solved in a closed form, as the number of terms in it grows exponentially over EM iterations. In the following we discuss the details of our treatments in three different cases.

4.3.1 $M \to \infty$: Content-Based Filtering

This case indicates that a very strong hyper prior $p(\alpha|\beta)$ has been imposed. The base distribution $p(\theta|\alpha^*)$ thus dominates the learned common prior, no matter we adopt the fully Bayesian solution Eq. (4.7) or MAP estimate Eq. (4.17). This gives

$$p(\theta|\mathcal{D}, \beta) = p(\theta|\alpha^*), \tag{4.19}$$

which implies that the observations $\mathcal{D}$ can hardly change our knowledge about the prior distribution of profile models. Then we can predict an active user's interests by

$$p(y|x, \mathcal{D}_a; \mathcal{D}, \beta) \quad = \quad \int p(y|x, \theta) p(\theta|\mathcal{D}_a; \alpha^*) d\theta \tag{4.20}$$

in which the dependence of predictions for user a on $\mathcal{D}$ is removed. We shall be aware of that in this case the hierarchical model actually degenerates to the non-hierarchical Bayesian model, which is actually the pure content-based filtering.

The model treats different users separately, which can be explained from two perspectives. First, since the prior distribution of θ does not adapt to the observations from other users, the learning process for a new user a is

independent to other users' opinions. Second, usually a rather wide $p(\theta|\alpha^*)$ is assumed before the learner observes something. Such a wide common prior implies *a priori* that random samples (i.e. profiling models) are likely to be different from each other.

4.3.2 $M \to 0$: Content-enhanced Collaborative Filtering

The impact of base distribution $p(\theta|\alpha^*)$ vanishes in this situation. According to Eq. (4.15) and Eq. (4.16), the estimated common prior is "reshaped" by multiplying $p(\mathcal{D}_i|\theta)$ at each iteration, and therefore repeatedly gets enhanced at positions θ_i^{ML} where $p(\mathcal{D}_i|\theta)$ has the maximum, while being suppressed in rest positions. At the convergence, $p(\theta|\alpha^k)$ ends up with a number of sharp peaks. Therefore we take a variational approximation:

$$p(\theta|\mathcal{D}_j, \alpha) \approx \sum_{i=1}^{N} \xi_{i,j} \delta_{\theta_i^{ML}}(\theta), \tag{4.21}$$

where $\theta_i^{ML} = \arg\max_\theta p(\mathcal{D}_i|\theta)$, and $\xi_{i,j}$ are variational parameters to be specified. Thus we can apply an EM algorithm, which directly factorizes $p(\theta|\alpha)$ with parameters $\xi_{i,j}$, and then estimates α^{MAP} by maximizing $p(\alpha|\mathcal{D}, \beta)$ as follows:

1. E-step: Based on $p(\theta|\alpha^{k-1})$ derived from the last step, we can calculate $\xi_{i,j}^k$

$$\xi_{i,j}^k = \frac{p(\mathcal{D}_j|\theta_i^{ML}) p(\theta_i^{ML}|\alpha^{k-1})}{\sum_{i=1}^{N} p(\mathcal{D}_j|\theta_i^{ML}) p(\theta_i^{ML}|\alpha^{k-1})} \tag{4.22}$$

2. M-step: Then we update the common prior

$$p(\theta|\alpha^k) = \sum_{i=1}^{N} \xi_i^k \delta_{\theta_i^{ML}}(\theta). \tag{4.23}$$

   where $\xi_i^k = \frac{1}{N} \sum_j^N \xi_{i,j}^k$.

At the beginning of iterations, we initialize $\xi_i = 1/N$ for each i. At the convergence we obtain the estimate of α^{MAP}, and the estimated common prior $p(\theta|\alpha^{MAP})$ as well.

By plugging the obtained $p(\theta|\alpha^{MAP})$ into Eq. (4.18), a very simple solution is finally derived to predict the active user a's interest for an item with features x,

$$p(y|x, \mathcal{D}_a; \alpha^{MAP}) = \sum_{i=1}^{N} w_i p(y|x, \theta_i^{ML}). \tag{4.24}$$

Since $\xi_i = p(\theta_i^{ML}|\alpha^{MAP})$, w_i can be rewritten as

$$w_i = \frac{1}{Z}\xi_i p(\mathcal{D}_a|\theta_i^{ML}) = p(\theta_i^{ML}|\mathcal{D}_a), \tag{4.25}$$

where Z is a normalization term. The derived prediction is simply a *weighted average* of predictions made by profiling models (i.e. ML estimates) of other users.

Let us take a closer look at the weighting terms Eq. (4.25): ξ_i suggests the *a prior* probability or the *typicalness* of θ_i^{ML}; $p(\mathcal{D}_a|\theta_i^{ML})$ indicates how well profile model θ_i^{ML} can *explain* the active user a's interests; Then the weighting term w_i models how likely the active user has user i's profiling model. Eq. (4.24) indicates that persons like-minded to the active user should have more impacts in predicting a's interests, which is essentially also the major assumption of CF, but here expressed in a probabilistic way.

However, the derived algorithm Eq. (4.24) is not simply CF, but *content-enhanced* CF, in the sense that many content-based predictors $p(y|x, \theta_i^{ML})$ are combined to make predictions. In certain conditions, the algorithm will degenerate to pure collaborative filtering (see Sec. 4.4).

4.3.3 M is medium: Hybrid Filtering

Previously we discussed two extreme situations where M is either very large or very small. To complete our discussion, we shall examine the third case where M is not either too large or too small. We again apply the same variational approximation and take the EM algorithm, in which the E-step remains the same as Eq. (4.22), while M-step becomes:

$$p(\theta|\alpha^k) = \frac{M p(\theta|\alpha^*) + N \sum_{i=1}^{N} \xi_i^k \delta_{\theta_i^{ML}}(\theta)}{M + N}. \tag{4.26}$$

where $\xi_i^k = \sum_j^N \xi_{i,j}^k$. At the end we obtain the estimate of $p(\theta|\alpha^{MAP})$. Finally predictions for the active user are made as follows

$$p(y|x, \mathcal{D}_a; \alpha^{MAP}) = w_0 p(y|x, \mathcal{D}_a; \alpha^*) + \sum_{i=1}^{N} w_i p(y|x, \theta_i^{ML}) \tag{4.27}$$

where

$$w_0 = \frac{1}{Z} M p(\mathcal{D}_a|\alpha^*), \tag{4.28}$$

$$w_i = \frac{1}{Z} N \xi_i p(\mathcal{D}_a|\theta_i^{ML}), \qquad \text{if} \quad i \neq 0 \tag{4.29}$$

where Z is the normalizer to make $w_0 + \sum_{i=1}^{N} w_i = 1$. The final predictive model Eq. (4.27) averages all the existing users' predictive models (ML predictors) and the active user's own predictor (a Bayesian content-based filter). The additional weighting term w_0 allows contributions directly from user a's own data, which is particularly useful when other existing profile models do not fit the active user a's interests very well.

4.4 Connections to Related Work

Our work not only offers a principled hybrid information filtering approach, but also generalizes a bunch of existing information filtering algorithms. We already know that when we impose a very strong hyper prior, the algorithm degenerates to the pure CBF.

Now let us examine its connections to pure CF which —in contrast to our content enhanced CF— would also give valid predictions without useful features. Without useful features, we can rely on the fact that if a user would be required to re-rate an item the user had already rated, the user would be consistent in that both ratings would be (nearly) identical. This fact can be implemented by using the previous rating of an already rated item instead of using the prediction of the user model. Then the Eq. (4.24) becomes very similar to memory-based CF [RIS+94, SM95, PHLG00]. Our methods differs in that we treat cases (i.e. users) with different typicalness (indicated by ξ_i) while other CF methods assume cases are equally typical. Interestingly, a similar effect can be mimicked by simply overfitting the model!

Furthermore, our work also generalizes or improves on many hybrid filtering algorithms. Melville et al [MMN02] suggest to build content-based model for each user and then generate *pseudo ratings* for non-rated items. The augmented data are used to feed a memory-based CF algorithm. Since pseudo ratings may be not accurate, heuristics like harmonic mean weighting are developed to incorporate the confidence of pseudo ratings. Our algorithm Eq. (4.24) essentially shares the same idea, but is derived in a principled way, in which the confidence of pseudo ratings are smoothly handled by *predictive distribution* of y. Moreover, Eq. (4.27) further suggests that the predictor conditioned on the active user's own data should also be included.

A big family of hybrid filtering algorithms (e.g. [Paz99, CGM+99]) firstly treat CBF and CF separately and then average both results to make final predictions. Eq. (4.27) improves them in two aspects: (1) the weighting terms to balance two parts can now be computed; (2) the CF part can be content-enhanced. As another example, Fab [BS97] maintains user profiles based on content analysis, and directly compare these profiles to determine similar users for collaborative filtering. An item is recommended to a user both when it scores highly against the user's own profile, and when it is also rated highly by users with similar profile. Eq. (4.27) expresses the spirit of Fab system.

4.5 Collaborative Ensemble Learning with Support Vector Machines

So far we have studied the general theoretical framework of nonparametric hierarchical Bayesian solutions to information filtering, but have not yet specified the detailed model $p(y|x,\theta)$. In principle, $p(y|x,\theta)$ can be implemented as any kind of parametric probabilistic predictive models. In the following we will introduce a version of realization with support vector machines (SVMs), called collaborative ensemble learning.

4.5.1 Support Vector Machines

Support vector machines (SVMs) are a classification technique with strong backing in statistical learning theory [Vap95]. They have been applied with great success in many challenging classification problems, including text categorization [Joa98] and image retrieval [TC01].

We consider SVM models for the preferences of user i, based on the ratings $\mathcal{D}_i$ this user has previously provided. A standard SVM would predict user i's rating on some item x, represented by its feature vector, by computing

$$y = \text{sign}(f^i(x)) = \text{sign}\Big(\sum_{j \in \mathcal{R}_i} y_{i,j}\alpha_{i,j}k(x_j, x) + b_i\Big) \tag{4.30}$$

$k(\cdot,\cdot)$ denotes the kernel function, which computes the pairwise similarities of two items. We will later use θ to stand for the SVM preference model for user i, with θ containing all SVM model parameters $\alpha_{i,j}$ and b_i. The weights $\alpha_{i,j}$ of the SVM are determined by minimizing the cost function

$$C\sum_{j \in \mathcal{R}_i}(1 - y_{i,j}f^i(x_j))_+ + \frac{1}{2}\boldsymbol{\alpha}_i^T K^i \boldsymbol{\alpha}_i \tag{4.31}$$

By $(\cdot)_+$, we denote a function with $(x)_+ = x$ for positive x, and $(x)_+ = 0$ otherwise. K^i is the matrix of all pairwise kernel evaluations on the training data $\mathcal{D}_i$, and $\boldsymbol{\alpha}_i$ is a vector containing all parameters $\alpha_{i,j}$.

4.5.2 Probabilistic Extensions to SVMs

In their standard formulation, SVMs do not output any measure of confidence for their prediction. Probabilistic extensions of the SVM, where an associated probability of class membership is output, have been independently suggested by several authors. For our work, we follow the idea of [Pla99], and compute the probability of membership in class y, $y \in \{+1, -1\}$ as

$$p(y|x, \theta_i) = \frac{1}{1 + \exp(yA_i f^i(x))} \tag{4.32}$$

A_i is the parameter[3] to determine the slope of the sigmoid function. This modified SVM retains exactly the same decision boundary $f^i(x) = 0$ as

[3]Platt's original formulation used an additional bias term in the denominator $1 + \exp(y(A_i f^i(x) + b_i))$. Since we typically only have very few training data available, we

defined in Eq. (4.30), yet allows an easy approximation of posterior class probabilities.

4.5.3 PSVM Parameter Tuning

The PSVM profile model has a few parameter that need to be set: The SVM models in Eq. (4.31) require the constant C that gives a weighting of errors on the training data. Furthermore, the kernel function $k(\cdot, \cdot)$ may be parameterized. As the last parameter, we need to tune the slope parameter A_i of the PSVM model in Eq. (4.32).

In our experiments, we use an SVM with the radial basis function (RBF) kernel for working on image data, and a linear kernel for text data. The kernel parameters, as well as the constant C, are chosen to minimize the leave-one-out error on the training data. Since the training set for most users is very small, this typically leads to overfitting. Thus, the kernel parameters are shared between models, and the optimization is with respect to the average leave-one-out error on all models. For choosing the slope A_i of the sigmoidal function Eq. (4.32), we follow the three-fold crossvalidation strategy suggested by [Pla99].

4.5.4 Collaborative Ensemble Learning

So far we have described a model for the preferences of an individual user, based on probabilistic SVMs (PSVMs). Given some training data containing items the user likes and dislikes, this model can predict—based on a description of items using a set of features—an individual user's preferences. SVM models are known for their excellent performance in many challenging classification problems. However, using only the models for individual users would pose the same problems as common CBF methods, in that the models have very high variance (due to the insufficient amount of training data from each individual) and only a poor generalization ability.

Now, we improve the performance of information filtering systems by

restrict the model to containing only one additional parameter A_i.

applying Eq. (4.27). We maintain a set of profile models θ_i from users who have used the system[4]. Then we use the EM learning to estimate ξ_i, as discussed in Sec. 4.3. Now suppose an active user a comes and gives annotates $\mathcal{D}_a$, we first learn his/her profile model θ_a and them make predictions by

$$p(y|x, \mathcal{D}_a) = w_0 p(y|x, \theta_a) + \sum_{i=1}^{N} w_i \cdot p(y|x, \theta_i) \tag{4.33}$$

where the weights w_0 and w_i are computed as Eq. (4.28) and Eq. (4.29), the concentration parameter M is set by cross-validation. To compute $p(\mathcal{D}_a|\alpha^*)$ in Eq. (4.28), we assume that α^* specifies a flat distribution where each item has equal chance to be liked or disliked. Eq. (4.33) realizes Eq. (4.27) via substituting $p(y|x, \mathcal{D}_a, \alpha^*)$ by $p(y|x, \theta_a)$ and θ_i^{ML} by learned PSVMs θ_i. The next section will demonstrate the success of this approach on several data sets.

4.6 Empirical Study

Empirical evaluations of our learning method are conducted in the following two experimental settings:

- *Simulation on 4533 painting images.* From *Meisterwerke der Malerei* CDs we collected 4533 painting images, covering antique Egyptian and Arab frescos, Chinese traditional paintings, India arts, European classical paintings, impressionism paintings, and modern arts in early 1900s. To enable an extensive objective measure of performance, we categorized them into 58 categories, mainly according to their respective artists. One artist corresponds to one category. We did not distinguish those artists for antique Arab, Egyptian, Chinese, and India paintings and just put them into four categories.

- *Simulation on news articles.* Reuters-21578 text data set is a collection of news articles that has been widely used in the research on information

[4]One may select a subset of users to get a compact model. Our current work does not discuss this issue

retrieval. Each article has been assigned a set of categories (which may be empty). From the total data, we eliminate articles without categories, titles or main texts, and categories with less than 50 articles. The main text for each article was pre-processed by removing stop words, stemming, and calculating TF-IDF (i.e. term frequency inverse document frequency) weights for the stemmed terms. The final data are 36 categories covering a total of $10,034$ articles, where $1,152$ articles belong to more than one category.

- *Online survey on 642 painting images.* We collected 642 painting images from Internet, mainly impressionism paintings and modern arts from 30 artists. To evaluate the algorithm performance on completely true user preferences, we performed a web-based online survey[5] to gather user ratings for 642 images. In the survey, each user gave ratings, i.e. "like", "dislike", or "not sure", to a randomly selected set of painting images. We so collected data from more than 200 visitors. After removing users who had rated less then 5 images, and users who had rated all of their images with one class (only like resp. only dislike), we retain a total of $L = 190$ users. On average, each of them had rated 89 images.

For all the images, we extract and combine *color histogram* (216-dim.), *correlagram* (256-dim.), *first and second color moments* (9-dim.) and *Pyramid wavelet texture* (10-dim.) to form 491-dimensional feature vectors to represent images. We use SVMs with RBF (radius basis function) kernel for images and SVMs with linear kernel for news articles. In our empirical study, we will mainly examine the accuracy of collaborative ensemble learning in terms of predicting users' interests in images or articles, and compare it with other two competitive algorithms:

- *SVM content-based retrieval* trains a SVM model on a set of examples given by an active user, and then apply the model to predict the active user's preferences. This algorithm represents a typical CBIR approach.

[5]The survey can be found on `http://honolulu.dbs.informatik.uni-muenchen.de:8080/paintings/index.jsp`.

- *Collaborative filtering* combines a society of advisory users' preferences to predict an active user's preferences. The combination is weighted by *Pearson correlation* between test user and other advisory users' preferences. The algorithm applied here is described in [BHK98].

4.6.1 Simulation with 4533 Painting Images

In this study, we will examine the retrieval accuracy of collaborative ensemble learning in cases that users have heterogenous interests for art images based on the 4533 painting images.

To enable objective evaluation, we need to "mimic" many users' preferences for the images. We assume that each user is interested in n categories. Since painting images from the same artist (e.g. one category) typically share similar painting styles, the assumption reflects the real-world cases to some extent, where one is interested in heterogeneous styles of paintings. We further assume that, without loss of generality, for a setting of n, there is $\mathcal{P}_n$, a set of profile types containing $58 - n + 1$ profile types and the p-th profile type is interested in n adjacent categories from the p-th to the $(p + n - 1)$-th one.[6] Then we stimulate a user's preference data in the following steps:

1. Randomly choose the value of n, where n can be 1, 2, or 3. Each possibility has equal chance.

2. Randomly assign a profile type in $\mathcal{P}_n$ to the user, where each profile type has equal chance.

3. Randomly produce 5 liked art images and 10 disliked art images based on the profile type assigned.

We repeat the procedure 1000 times and thus produce 1000 users' preference data. The detailed setting-up is based on some assumptions, however, we believe that it approaches real-world cases from certain perspectives. Since it is not easy to gather the *ground truth*, i.e. sufficient true-user preferences for an art image base, it is necessary to perform simulations at this early stage.

[6]The image categories are sorted in alphabet order of artist names.

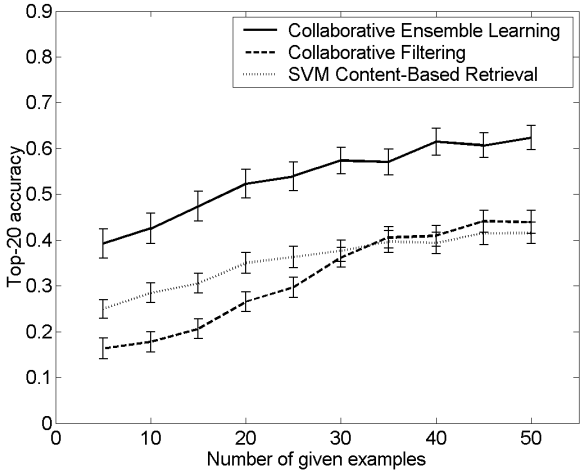


Figure 4.2: Top-20 accuracy with various number of given examples for each active user. For each advisory user, we assume that 5 liked and 10 disliked images are given (Simulation on 4533-painting data)

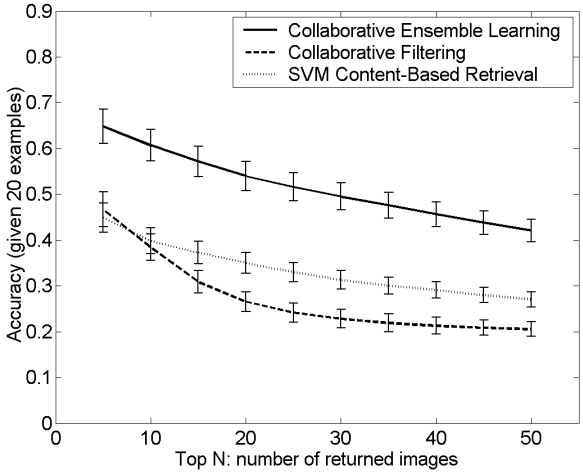


Figure 4.3: Accuracy with various number of returned images. For each active user, we fix the number of given examples to 20. For each advisory user, we assume that 5 liked and 10 disliked images are given (Simulation on 4533-painting data)

Our experiments take a leave-one-out scheme, in which a user is picked up as a test user (i.e. active user) and the remaining ones serve as *advisory users*. Then the test user's profile type serves as the ground truth for evaluation. Based on the profile type, we generate a number of examples, with approximately 1/3 liked images and 2/3 disliked ones, to feed the art image retrieval system. We use top-N accuracy to measure the performance, i.e. the fraction of truly liked images among the N top ranked images. We change the number of given examples for each active user, i.e. 5, 10, 15, 20, 25, 30, 35, 40, 45 and 50, to study the learning curve of the three compared methods. For one learning curve, we repeat the procedure for 10 times with different random seeds and each run will go through all the active users. Finally we compute the mean and standard deviation of the mean over the 10 runs. The obtained final results have been shown in Fig. 4.2. Collaborative ensemble learning significantly outperforms the other two algorithms, which indicates that the algorithm effectively captures simulated users' diverse interests for art images. While the SVM content-based retrieval shows a poor accuracy. The results confirm our analysis that although SVM demonstrate excellent learning performances in many real-world problems, it suffers the problems of modelling users' diverse interests due to the deficiency of low-level features. Collaborative filtering performs the worst in our simulation, because the preference ratings given by advisory users are very sparse, i.e. only 0.33% of the images are rated for each user. Collaborative filtering heavily relies on the user ratings while ignoring the descriptive features of images. It cannot compute reliable Pearson correlation between two users if they have few commonly rated examples. While our proposed collaborative ensemble learning generally overcomes the weaknesses of SVM content-based approach and collaborative filtering by incorporating wider information and thus achieves the best accuracy.

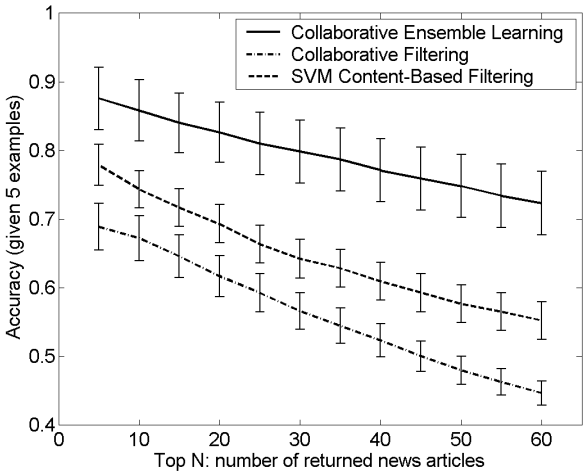
In the following, we fix the number of given examples for each active user to 20 and vary N, the number of top ranked results that are returned. Accuracy is then computed for all the active users and the procedure is repeated for 10 times with different random seeds. Finally the mean and error bar of the mean are calculated and demonstrated in Fig. 4.3. Accuracies of the three approaches are all decreasing as we increase the number of N, indicat-

ing that all the three methods present ranking which is better than random guess (which should be a flat line with accuracy insensitive to the value of N). However, collaborative ensemble learning clearly demonstrates the best performance. Interestingly, collaborative filtering's accuracy decreases the most quickly with N increasing. This is because that collaborative filtering is not able to generalize examples to similar cases (i.e. images distributed very close to the given examples in the low-level feature space), and thus cannot make judgements on the images never visited by any advisory user (i.e. new images). Therefore, it "consumes out" those limited number of liked images which could be suggested by advisory users at the early stage and cannot present more truly liked images when N further increases. This observation indicates that content-based approach has the ability of generalizing examples to never-rated cases, and clearly collaborative ensemble learning takes over and further enhances this advantage.

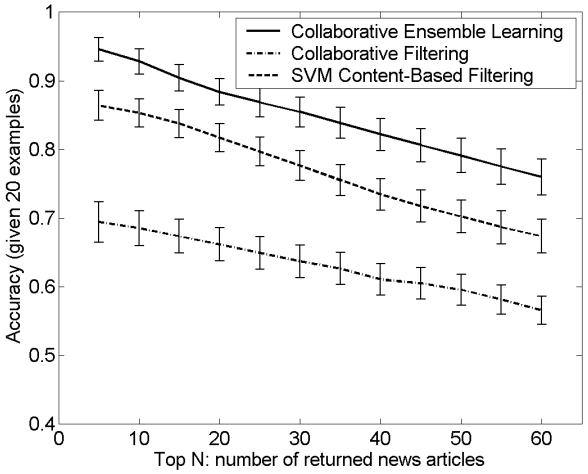
4.6.2 Text Retrieval on REUTERS-21578

Now we report results from a controlled simulation based on the Reuters-21578 text data set. For the experiments, we assume that each user is interested in exactly one of categories (an assumption that has been widely used in the literature on text retrieval). To further simulate the working environment of collaborative ensemble learning, we generate $N = 360$ users and assume that each user has annotated 30 articles. The annotation for an article is either $+1$ (if it falls into the assumed category for this particular user) or -1. In this way we get the training data $\mathcal{D}_i$, $i = 1, \ldots, N$. We report results for two scenarios, with 5 and 20 examples are given by the active users.

To evaluate collaborative ensemble learning, we examine the average accuracy of top N returned articles for a set of 180 active users, for whom we assume that 5 or 20 examples have been given by each active user. In both training and test data, categories have an equal chance to be assigned to users. We draw items (that is, articles) uniformly at random as well, but ensure that approximately 1/3 positive and 2/3 negative examples are selected for each user.



(a)



(b)

Figure 4.4: Accuracy with various number of returned news articles. (a) for each active user, we assume that 5 examples are given, (b) for each active user, we assume that 20 examples are given

Using the experimental setup, we compare collaborative ensemble learning with two other methods for information filtering: (1) collaborative filtering using Pearson correlation [BHK98], and (2) content-based filtering using SVM with linear kernel [DSG01, Joa98].

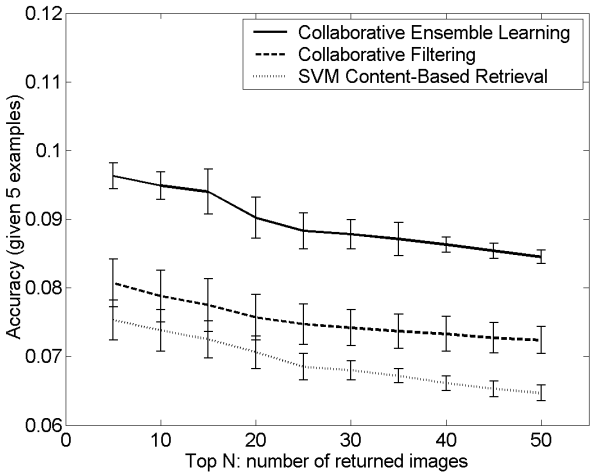
By changing the number of returned top N articles, we obtained two curves in Fig. 4.4. In this setting we find that, in contrast to the earlier image case, content-based filtering performs much better than collaborative filtering for text data. This is mainly because that the TF-IDF textual features are very effective in representing the topics of articles.

Collaborative filtering greatly suffers from the sparsity of user annotations because for two curves respectively only 0.05% and 0.3% articles are rated by advisory users. There are a large amount of articles which can not be processed by collaborative filtering at all because they have never been rated by any user.

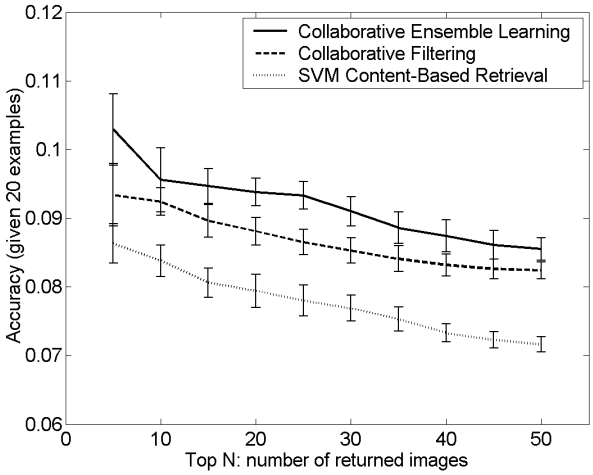
Again, collaborative ensemble learning significantly outperforms both other methods, with accuracy $10\% - 20\%$ better than content-based filtering.

4.6.3 Experiments with the Online Survey Data

Although we get impression that collaborative ensemble learning presents excellent performances, however, simulation can not replace the real-world cases. In this section, we will examine the performance of the three approaches based on 190 user's preference data on 642 painting images, which are gathered from the on-line survey. Again, we use top-N accuracy to evaluate the performance. Since we can not require a user to rate all of the 642 painting images in the survey, for each user we just partially know the "ground truth" of preferences. As a result, the true precision cannot be computed. We thus adopt the accuracy measure that is the fraction of *known* liked images in top ranked N images. The quantity is smaller than true accuracy because *unknown* liked images are missing in the measurement. However, in our survey, the presenting of images to users is completely random, thus the distributions of rated/unrated images in both unranked and ranked lists are also random. This randomness does not change the relative values of compared methods but just the absolute values. Thus in our follow-



(a)



(b)

Figure 4.5: Accuracy with various number of returned images. (a) for each active user, we assume that 5 examples are given, (b) for each active user, we assume that 20 examples are given

ing experiment it still makes sense to use the adopted accuracy measurement to compare the three retrieval methods.

Our experiment takes the leave-one-out scheme again, in which we pick up each user as the active user and treat all other users as collected advisory users. We fix the number of given examples for each active user to 5 and 20 respectively, and examine the retrieval accuracy in the cases of returning various N top ranked images. We take the same methodology as Fig. 4.3 and demonstrate the results in Fig. 4.5-(a) (given 5 examples) and Fig. 4.5-(b) (given 20 examples). We find that collaborative ensemble learning achieves the best accuracy in both cases. Since in the data user ratings are much denser than the simulation case, collaborative filtering outperforms the SVM content-based method. Interestingly, the accuracy improvement of collaborative ensemble learning over the other two approaches are more impressive in the given-5 case. This is a very nice property for art image retrieval because users are normally not patient at the initial information-gathering stage and it is much desired to get satisfactory accuracy with only a few examples. Theoretically, this nice property can be explained from the Bayesian perspective, where we use "an informative prior" learned from all the users to constraint the Bayesian inference. Such a prior knowledge gained from population promises a good accuracy even when limited examples are fed to the learning system.

In the next, we take a closer look at a case study. As shown in Fig. 4.6, we let a user input a positive and a negative examples to run the collaborative ensemble learning algorithm. The returned top 20 results look quite diverse and meanwhile very different from the positive example. Surprisingly, the user loves 18 out of the 20 images and there is no strongly disliked image. As a comparison, we present the results of SVM content-based approach trained on the same examples in Fig. 4.7. We find that 8 results are actually from the same artist as the positive example is. The user told us that he strongly dislikes the images (1,4), (3,2) (3,5), (4,1), (4,3), (4,4) and (4,5).[7] This case study is quite interesting, which demonstrates that, in the studied case where a user gives examples that only partially convey his preferences, collaborative ensemble learning effectively infer the user's comprehensive interests while

[7]Here we treat the presented 20 images as a 4 by 5 matrix.

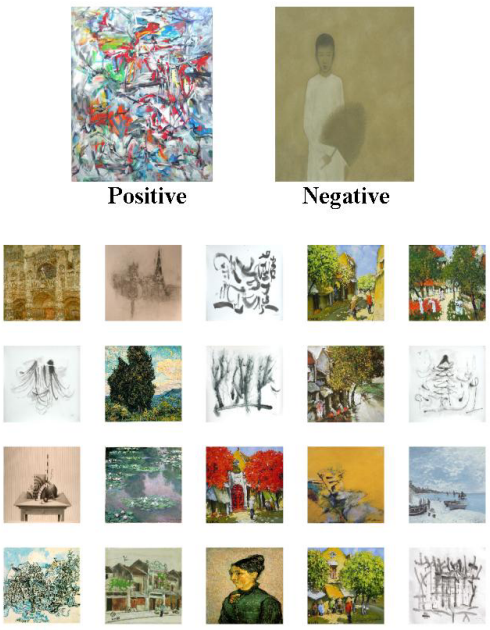


Figure 4.6: Case study: Two images on the top are examples given by a user. The lower 20 images are the top-20 results returned by collaborative ensemble learning.

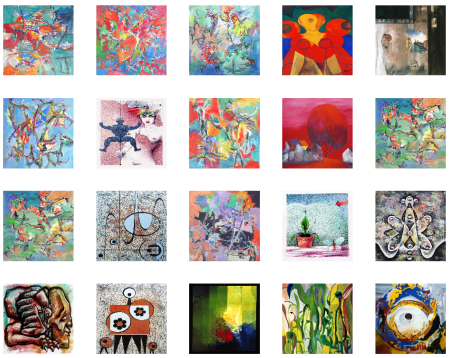


Figure 4.7: Top-20 results returned by SVM content-based retrieval. Examples are the same as the ones shown in Fig. 4.6.

SVM content-based approach only returns images that are similar to the positive example(s). In the art image retrieval application, presenting interesting but *novel* images to active users is a very nice property because a user can easily find images from the same artist (by category-based search) while has difficulties in locating potentially interesting images which are currently unknown to the user.

4.7 Conclusions

This chapter describes a theoretical framework—*nonparametric hierarchical Bayesian approaches* to hybrid information filtering. Traditionally, most of the information retrieval and filtering systems apply non-hierarchical content-based models. These methods ignore the connections between different users' information needs. Then a session of information service can not inherit knowledge from other sessions. In our work, each user is modelled by a parametric content-based profile model, whose parameters θ are generated from a common prior distribution $p(\theta)$, which is shared by all the users. Then users are connected to each other statistically via the common prior.

To complete a fully Bayesian model and enable the learning of such a

common prior from data, we assume the common prior is a sample generated from a *hyper prior*, i.e. "a prior distribution of the common prior distribution".

Since the high complexity of the common prior can hardly be covered by any parametric distribution, we describe a nonparametric form for the common prior—an infinite multinomial distribution—which is a sample generated from a *Dirichlet process*, i.e. the hyper prior.

We derive effective EM algorithms to learn the common prior from data annotated by users. In particular, various approximations are developed to solve analytically infeasible computations. The finally achieved predictive approaches are surprisingly simple and intuitively understandable.

- If a very strong hyper prior is assigned, then the learned common prior distribution can hardly be influenced by our empirical observations and remains the same as the base distribution. Therefore different users' information needs can not be connected to each other via the common prior distribution. In this case the hierarchical modelling degenerates to conventional non-hierarchical modelling, which is actually the pure content-based filtering, assuming users are independent.

- If a very weak hyper prior is assigned, then the impact of base distribution vanishes and the learned common prior is completely adapted to empirical data. As a direct result, predictions for an active user are made by a committee of *other* users' profile models (ML estimates). Users who are more like-minded to the active user will have more impacts in the committee. Here a principled hybrid filtering algorithm is derived since many content-based models are combined in a collaborative way. Interestingly, this method also leads to the pure collaborative filtering algorithm described in Ch. 2.

- If a normal hyper prior is assigned, the learned common prior is a trade-off between the base distribution and the empirical distribution. When existing profile models can not well explain the active user's data, the model will automatically give high chances to other settings of models. This is a very general framework for hybrid information filtering,

which explains a large family of existing hybrid filtering algorithms and suggest further improvements.

Finally we design the collaborative ensemble learning algorithm with SVMs, which is a realization of hybrid approach combining the basic idea of content-based filtering and collaborative filtering. The performance of collaborative ensemble learning has been extensively tested. As compared to pure content-based and collaborative filtering, collaborative ensemble learning achieved excellent performance for various data sets.

Our work not only presents a hybrid information filtering solution, but also unifies pure content-based filtering, pure collaborative filtering, and hybrid filtering in a single theoretical framework. Most existing information filtering algorithms can also be explained in the framework, and principled improvements are suggested. To our best knowledge, we have not seen similar work in the literature.

Moreover, the nonparametric hierarchical model provides a general methodology for modelling a population of related objects, like costumers in marketing analysis, hospitals in clinical analysis, patients in heath care, automachines for banks. We believe the work is a strong contribution to a wide range of data modelling tasks.

Chapter 5

Conclusions and Future Work

This thesis has been focusing on an important technology, information filtering, which aims to understand people's information needs and find desired information items. The work extensively studies major branches of information filtering approaches, including collaborative filtering, content-based filtering, and hybrid filtering. Our emphasis is not only on exploring novel and effective algorithms for solving various real-world problems, but also building a theoretical framework that provides a unifying view for information filtering. Besides showing the technical soundness of our solutions, the unifying view make us deeply understand the the relations between people in a community and pave the way for further developments. Particularly, probability theory and Bayes theory have been intensively used throughout the whole thesis, as the natural language to build flexible models, encode the uncertainty of user profiles, model their intrinsic dependence, and integrate our prior knowledge into learning processes. In the following we conclude the major aspects of this thesis and point out some future directions as well.

5.1 Probabilistic Memory-Based Collaborative Filtering

Collaborative filtering has recently been widely applied in recommender systems. It maintains a database of user ratings (or annotations) and explores

the similarity between users' opinions for making predictions. Due to its simplicity, memory-based approach is the most popular collaborative filtering technique. Various heuristics for memory-based methods have been proposed in the literature. In contrast, our work focuses on a probabilistic version of memory-based collaborative filtering (PMCF). The model describes connections between users' interests probabilistically. More importantly, various principled extensions are then readily supported.

One such an extension is active learning. To make predictions for an active user, a recommender system must previously know something about this user's interests. The easiest way is of course present some information items to the user for feedbacks. Conventional approaches did it in a passive way. Instead, our work does one earliest attempt to actively present unrated items to users. The choice of unrated items is made to maximize some *expected* gain according to the current knowledge we have known about the user.

The other extension aims to reduce the computational cost of prediction making via working with a small subset of stored users. By choosing typical preference prototypes, predictions can be made accurately and efficiently. The selection procedure is derived directly from the PMCF framework and aims to preserve a minimum loss of the data density (measured by K-L divergence). The derived data selection algorithm focuses on the data that are novel to our current knowledge, but are in fact typical in the real world, which is quite intuitively understandable.

With the development of PMCF framework, we interestingly demonstrate a learning system that actively *queries* the objects that we want to know (e.g. asks questions to users), and *samples* the data that we have gathered (e.g. chooses the subset of data). In general, one wishes that learning systems are able to know what to learn, where to learn and how to learn. We make one step towards this direction, but there is still a long way to go.

In memory-based collaborative filtering, predictions for one active user are made by taking data from like-minded users, who themselves are often not completely known to us (as typically we only observe limited data from each user). Obviously, a better understanding of other users will boost our predictions for current active user, while a better understanding of current

user will again benefit for others. In future work we should consider this *propagation* (or mutual enhancement) effects in collaborative filtering.

5.2 Generalized Probabilistic Principal Component Analysis for Content-based Filtering

Content-based filtering has its root back in information retrieval and aims at helping users find interesting items (e.g. documents, images) by exploring the content similarity of items. Typically, one item is associated with different content features that seems to be developed independently but are intrinsically related. To better characterize each item, one has to combine different perspectives of features together, which brings major challenges to content-based filtering, like homogenizing heterogenous data, reducing dimensionality, and exploring intrinsic factors.

In Ch. 3 we develop a generalized probabilistic principal component analysis model (GPPCA), which describes the dependence of high dimensional feature vector with heterogenous attributes (i.e. continuous and binary) by low-dimensional hidden variables that are distributed as a multivariate Gaussian. Since categorical data can be encoded by binary data, the model can also handle categorical attributes. We adopt a variational approximation to the likelihood of data and describe a variational EM algorithm to fit the model. The model allows a unified treatment to mixed types of attributes and thus brings great benefits for multivariate data analysis, visualization, and dimensionality reduction. For content-based filtering, mixed sources of attributes can be merged into low-dimensional continuous feature vectors, which can be easily processed by most of the existing content-based filtering algorithms. Since the derived data account for the dependence of original content features and thus often reflect the semantics of items, the quality of filtering can thus be improved. The advantages of the proposed model are illustrated on toy data and real-world painting image data for both visual-

ization and recommendation.

There are several directions to improve the work. Currently the hidden variables are described as a multivariate Gaussian distribution, which is sometimes a restrictive assumption. To relax this limitation, we may consider a mixture of Gaussian for the hidden variables. This modification actually assume that items can be structured into clusters. Clustering data with mixed types of attributes is also an interesting research topic.

Our work is also closely related to multi-output regression or classification. A straightforward solution might treat the problem as many independent regression/classification problems. In contrast, some algorithm like GPPCA can map inputs into hidden variables that really account for the multivariate outputs. Thus the dependence between each prediction problems are explored in this new framework. Actually, the problem is closely related to information filtering, since predicting for one user can be treated as a one-output problem and predicting the interests of many users then becomes a multi-output predicting problem. Formulating information filtering in this way will be an interesting future work.

So far the mapping from observations to hidden variables has been assumed to be linear. However, a more general case is of course nonlinear mapping. Learning a nonlinear hidden variable model is a fundamentally important problem.

5.3 Hierarchical Bayesian Framework for Hybrid Filtering

As we already know, collaborative filtering makes predictions mainly by exploring the connections between users, while content-based filtering mainly does the job by considering the similarity of contents between items. One natural extension is to combine the two methods together, in order to take the maximum advantages of data that we have. However, how to realize this combination remains an fundamentally unsolved problem.

In Ch. 4 we want to propose a principled solution to combine collaborative

filtering and content-based filtering. Finally, it turns out that actually a unifying framework for information filtering, including collaborative filtering, content-based filtering, and hybrid filtering has been developed. In addition, various previous work on hybrid filtering can be explained in this framework and further improvements are suggested.

The nonparametric hierarchical Bayesian framework has been designed, in which each user is modelled by a parametric content-based profile model, whose parameters θ are generated from a common prior distribution $p(\theta)$, which is shared by all the users. Then users are connected to each other statistically via the common prior. Since the high complexity of the common prior can hardly be covered by any parametric distribution, we describe a nonparametric form for the common prior—an infinite multinomial distribution—which itself is a sample generated from a *Dirichlet process*, i.e. the hyper prior distribution.

We derive effective EM algorithms to learn the common prior from data annotated by users. In particular, various approximations are developed to solve analytically infeasible computations. The finally achieved predictive approaches are surprisingly simple and intuitively understandable.

Finally we design the collaborative ensemble learning algorithm with SVMs, which is a realization of hybrid approach combining the basic idea of content-based filtering and collaborative filtering. The performance of collaborative ensemble learning has been extensively tested. As compared to pure content-based and collaborative filtering, collaborative ensemble learning achieved excellent performance for various data sets including images, paintings and news articles.

Our work is quite general in the sense that more complex models for each individual can be adopted. For example, we may take into account the sequential factor of user annotations, then for each user we build a Markov predictive model with parameters generated from a common prior distribution, which is again from a Dirichlet process. As another example, in the described of realization with SVMs, we require users to explicitly give both positive and negative examples. We can let users just give positive examples, as in most cases users are only willing to annotate implicitly, like browsing web pages, clicking hyperlinks, purchasing books and so on. Then we can

model user annotations as points distributed in the content feature space and model this distribution in some way (e.g. mixture of Gaussian). Finally the a hierarchical model can be build on the top of this user profile model.

The nonparametric hierarchical model does something beyond information filtering. It provides a general methodology for modelling a population of related objects, like costumers in marketing analysis, hospitals in clinical analysis, patients in heath care, automachines for banks, hackers in intrusion detection and so on. We believe the work is a strong contribution to a wide range of data modelling tasks. In the future we are going to pursue its extensions in different application fields.

Bibliography

[Ant74] C. E. Antoniak. Mixtures of dirichlet processes with applications to Bayesian nonparametric problems. *Annals of Statistics*, 2(6), Nov. 1974.

[BGJT04] D. Blei, T. L. Griffiths, M. I. Jordan, and J. B. Tenenbaum. Hierarichical topic models and the nested chinese restaurant process. In *Advances in Neural Information Processing Systems 16*. MIT Press, 2004.

[BHC98] C. Basu, H. Hirsh, and W. W. Cohen. Recommendation as classification: Using social and content-based information in recommendation. In *Proceedings of the Fifteenth National Conference on Artificial Intelligencen AAAI/IAAI*, pages 714–720, 1998.

[BHK98] J. S. Breese, D. Heckerman, and C. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence*, pages 43–52, 1998.

[Bis95] C. M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, 1995.

[BP98] D. Billsus and M. J. Pazzani. Learning collaborative information filters. In *Proceedings of the 15th International Conference on Machine Learning*, pages 46–54. Morgan Kaufmann, San Francisco, CA, 1998.

[BP99] D. Billsus and M. J. Pazzani. A personal news agent that talks, learns and explains. In Oren Etzioni, Jörg P. Müller, and Jeffrey M. Bradshaw, editors, *Proceedings of the Third International Conference on Autonomous Agents (Agents'99)*, pages 268–275, Seattle, WA, USA, 1999. ACM Press.

[BS97] M. Balabanovic and Y. Shoham. Fab: Content-based, collaborative recommendation. *Communications of the ACM*, 40(3):66–72, 1997.

[BSW98] C. M. Bishop, M. Svensen, and C. K.I. Williams. GTM: The generative topographic mapping. *Neural Compuation*, 10(1):215–234, 1998.

[BYRN99] R Baeza-Yates and B Ribeiro-Neto. *Modern Information Retrieval*. ACM Press, New York, 1999.

[BZ03] C. Boutilier and R. S. Zemel. Online queries for collaborative filtering. In *Ninth International Workshop on Artificial Intelligence and Statistics*, 2003.

[CDS01] M. Collins, S. Dasgupta, and R. Schapire. A generalization of principal component analysis to the exponential family. In Todd K. Leen, Thomas G. Dietterich, and Volker Tresp, editors, *Advances in Neural Information Processing Systems*, 13. MIT Press, 2001.

[CGM+99] M. Claypool, A. Gokhale, T. Miranda, P. Murnikov, D. Netes, and M. Sartin. Combining content-based and collaborative filtering in an online newspaper. In *Proceedings of ACM SIGIR Workshop on Recommender Systems*, August 1999.

[CLL00] E. Chang, B. Li, and C. Li. Toward perception-based image retrieval. In *IEEE Content-Based Access of Image and Video Libraries*, pages 101–105, June 2000.

[CMOY96] I. J. Cox, M. L. Miller, S. M. Omohundro, and P. N. Yianilos. Pichunter: Bayesian relevance feedback for image retrieval. In

Proceedings of International Conference on Pattern Recognition, volume 3, pages 361–369, Austria, 1996.

[Coh03] D. Cohn. Informed projections. In Suzanna Becker, Sebastian Thrun, and Klaus Obermayer, editors, *Advances in Neural Information Processing Systems*, 15. MIT Press, 2003.

[CS98] L. Chen and K. Sycara. WebMate: A personal agent for browsing and searching. In Katia P. Sycara and Michael Wooldridge, editors, *Proceedings of the 2nd International Conference on Autonomous Agents (Agents'98)*, pages 132–139, New York, 9–13, 1998. ACM Press.

[CT91] T. Cover and J. Thomas. *Elements of Information Theory*. Wiley, 1991.

[DG02] M. Davy and S.J. Godsill. Audio information retrieval: A bibliographical study. CUED/F-INFENG 429, Cambridge University, Engineering Department, Feb. 2002.

[DKHR98] B.J. Dahlen, J.A. Konstan, J.L. Herlocker, and J. Riedl. Jumpstarting movielens: User benefits of starting a collaborative filtering system with dead data. Technical Report 7, University of Minnesota, 1998.

[DLR77] A.P. Dempster, N.M. Laird, and D.B. Rubin. Maximum likelihood from incomplete data via the em algorithm (with discussion). *Journal of the Royal Statistical Society B*, 39:1–38, 1977.

[DSG01] H. Drucker, B. Shahrary, and D.C. Gibbon. Relevance feedback using support vector machines. In *Proceedings of 18th International Conference on Machine Learning*, pages 122–129, 2001.

[EW95] M. D. Escobar and M. West. Bayesian density estimation and inference using mixtures. *Journal of the American Statistical Association*, 90(430), June 1995.

[FD92] P. W. Foltz and S. T. Dumais. Personalized information delivery: An analysis of information filtering methods. *Communications of the ACM*, 35(12):51–60, Dec 1992.

[Fis96] G. Fishman. *Monte Carlo Concepts, Algorithms and Applications*. Springer Verlag, 1996.

[FSN+95] M. Flickher, H. Sawhney, W. Niblack, J. Ashley, Q. Huang, B. Dom, M. Gorkani, J. Hafner, D. Lee, D. Petkovic, D.Steele, and P. Yanker. Query by image and video content: The qbic system. *IEEE Computer*, 28(9):23–32, 1995.

[GRGP01] K. Goldberg, T. Roeder, D. Gupta, and C. Perkins. Eigentaste: A constant time collaborative filtering algorithm. *Information Retrieval Journal*, 4(2):133–151, 2001.

[GSK+99] N. Good, J.B. Schafer, J. Konstan, A. Borchers, B. Sarwar, J. Herlocker, and J. Riedl. Combining collaborative filtering with personal agents for better recommendations. In *Proceedings of AAAI-99*, pages 439–446, 1999.

[HBR94] D. Heckerman, J. Breese, , and K. Rommelse. Troubleshooting under uncertainty. Technical Report MSR-TR-94-07, Microsoft Research, 1994.

[HCM+00] D. Heckerman, D.M. Chickering, C. Meek, R. Rounthwaite, and C. Kadie. Dependency networks for inference, collaborative filtering, and data visualization. *Journal of Machine Learning Research*, 1:49–75, 2000.

[Hec95] D. Heckerman. A tutorial on learning with Bayesian networks. Technical Report MSR-TR-95-06, Microsoft Research, 1995.

[HKBR99] J.L. Herlocker, J.A. Konstan, A. Borchers, and J. Riedl. An algorithmic framework for performing collaborative filtering. In *Proceedings of the ACM Conference on Research and Development in Information Retrieval (SIGIR'99)*, pages 230–237. ACM, 1999.

[HKR00] J.L. Herlocker, J.A. Konstan, and J. Riedl. Explaining collaborative filtering recommendations. In *Proceedings of computer supported cooperative work (CSCW'00) conference*, pages 241–250, 2000.

[HP99] T. Hofmann and J. Puzicha. Latent class models for collaborative filtering. In *Proceedings of IJCAI'99*, pages 688–693, 1999.

[HSRF95] W. Hill, L. Stead, M. Rosenstein, and G. Furnas. Recommending and evaluating choices in a virtual community of use. In *Proceedings of ACM CHI'95 Conference on Human Factors in Computing Systems*, pages 194–201, 1995.

[HTF01] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Satatistical Learning*. Springer Verlag, 2001.

[HTH00] P. Hong, Q. Tian, and T. Huang. Incorporate support vector machines to content-based image retrieval with relevant feedback. In *Proceedings of the 7th IEEE International Conference on Image Processing (ICIP00)*, pages 750–753, 2000.

[Jen01] F. V. Jensen. *Bayesian Networks and Decision Graphs*. Statistics for Engineering and Information Science. Springer, 2001.

[JFM97] T. Joachims, D. Freitag, and T. Mitchel. Webwatcher: A tour guide for the world wide web. In *Proceedings of the 15th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 770–775. Morgan Kaufmann, 1997.

[JGJS99] M. I. Jordan, Z. Ghahramani, T. Jaakkola, and L. K. Saul. An introduction to variational methods for graphical models. *Machine Learning*, 37(2):183–233, 1999.

[JJ00] T. Jaakkola and M. Jordan. Bayesian parameter estimation via variational methods. *Statistics and Computing*, pages 25–37, 2000.

[Joa98] T. Joachims. Text categorization with support vector machines: Learning with many relevant features. In *Proceedings of European Conference on Machine Learning*. Springer, 1998.

[Lan95] K. Lang. NewsWeeder: learning to filter netnews. In *Proceedings of the 12th International Conference on Machine Learning*, pages 331–339. Morgan Kaufmann publishers Inc.: San Mateo, CA, USA, 1995.

[LAR02] W. Lin, S.A. Alvarez, and C. Ruiz. Collaborative recommendation via adaptive association rule mining. *Data Mining and Knowledge Discovery*, 6(1):83–105, Jan 2002.

[LC94] D. Lewis and J. Catlett. Heterogeneous uncertainty sampling for supervised learning. In *Proceedings of the Eleventh International Conference on Machine Learning*, pages 148–156. Morgan Kaufmann, 1994.

[Lee01] W.S. Lee. Collaborative learning for recommender systems. In *Proc. 18th International Conf. on Machine Learning*, pages 314–321, 2001.

[Lie95] H. Lieberman. Letizia: An agent that assists web browsing. In Chris S. Mellish, editor, *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI-95)*, pages 924–929, Montreal, Quebec, Canada, 1995. Morgan Kaufmann publishers Inc.: San Mateo, CA, USA.

[LW03] J. Li and J. Wang. Automatic linguistic indexing of pictures by a statistical modeling approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(9):1075–1088, 2003.

[Mae94] P. Maes. Agents that reduce work and information overload. *Communications of the ACM*, 37(7):31–40, July 1994.

[MDH99] S. Macskassy, A. Dayanik, and H. Hirsh. Emailvalet: Learning user preferences for wireless email. In *Proceedings of Learning about Users Workshop, IJCAI'99*, 1999.

[MK93] P. Maes and R. Kozierok. Learning interface agents. In *Proceedings of the Eleventh National Conference on Artificial Intelligence*, pages 459–465. AAAI Press, 1993.

[MM96] B. S. Manjunath and W. Y. Ma. Texture features for browsing and retrieval of image data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(8):837–842, 1996.

[MM99] W. Y. Ma and B.S. Manjunath. Netra: A toolbox for navigating large image database. *ACM Mulitimedia Systems*, 7:184–198, 1999.

[MMN02] P. Melville, R. J. Mooney, and R. Nagarajan. Content-boosted collaborative filtering for improved recommendations. In *Proceedings of the Eighteenth National Conference on Artificial Intelligence (AAAI-2002)*, pages 187–192, Edmonton, Canada, 2002.

[Mou96] I. Moustaki. A latent trait and a latent class model for mixed observed variables. *British Journal of Mathematical and Statistical Psychology*, 49:313–334, 1996.

[MP00] K. Miyahara and M. J. Pazzani. Collaborative filtering with the simple Bayesian classifier. In *Proceedings of the Sixth Pacific Rim International Conference on Artificial Intelligence PRICAI 2000*, pages 679–689, 2000.

[MR00] R.J. Mooney and L. Roy. Content-based book recommending using learning for text categorization. In *Proceedings of the Fifth ACM Conference on Digital Libaries*, pages 195–204, San Antonio, US, 2000. ACM Press, New York, US.

[Paz99] M.J. Pazzani. A framework for collaborative, content-based and demographic filtering. *Artificial Intelligence Review*, 13(5–6):393–408, 1999.

[PHLG00] D. M. Pennock, E. Horvitz, S. Lawrence, and C.L. Giles. Collaborative filtering by personality diagnosis: A hybrid memory-

and model-based approach. In *Proc. of the 16th Conference on Uncertainty in Artificial Intelligence*, pages 473–480, 2000.

[Pla99] J. C. Platt. Probabilities for SV machines. In A. Smola, P. Bartlett, B. Scholkopf, and D. Schuurmans, editors, *Advances in Large Margin Classifiers*, pages 61–74, Cambridge, MA, 1999. MIT Press.

[PMB96] M. Pazzani, J. Muramastsu, and D. Billsus. Syskill and webert: Identifying interesting web sites. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence*, pages 54–61, Portland, OR, August 1996.

[PUPL01] A. Popescul, L. Ungar, D. Pennock, and S. Lawrence. Probabilistic models for unified collaborative and content-based recommendation in sparse-data environments. In *17th Conference on Uncertainty in Artificial Intelligence*, pages 437–444, Seattle, Washington, August 2–5 2001.

[RAC+02] A.M. Rashid, I. Albert, D. Cosley, S.K. Lam, S.M. McNee, J.A. Konstan, and J. Riedl. Getting to know you: Learning new user preferences in recommender systems. In *Proceedings of International Conference on Intelligent User Interface (IUI2002)*, San Fransisco, CA, 2002.

[RG99] S. Roweis and Z. Ghahramani. A unifying review of linear gaussian models. *Neural Computaion*, 11:305–345, 1999.

[RHOM98] Y. Rui, T. S. Huang, M. Ortega, and S. Mehrotra. Relevance feedback: A power tool in interactive content-based image retrieval. *IEEE Trans. Circuits and Systems for Video Tech.*, 8(5):644–655, 1998.

[RIS+94] P. Resnick, N. Iacovou, M. Sushak, P. Bergstrom, and J. Riedl. Grouplens: An open architecture for collaborative filtering of netnews. In *Proceedings of the 1994 Computer Supported Collaborative Work Conference*, pages 175–186. ACM, 1994.

[Roc71] J. J. Rocchio. Relevance feedback in information retrieval. In *The SMART Retrieval System: Experiments in Automatic Document Processing*, pages 313–323. Prentice Hall, 1971.

[SC96] J.R. Smith and S.-F. Chang. Automated image retrieval using color and texture. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, November 1996.

[SKKR00] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Analysis of recommendation algorithms for e-commerce. In *Proceedings ACM E-Commerce Conference*, pages 158–167, 2000.

[SKKR01] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of 10th World Wide Web (WWW10) conference*, pages 285–295, Hong Kong, 2001.

[SM83] G. Salton and M. McGill. *Introduction to Modern Information Retrieval*. McGraw-Hill, 1983.

[SM95] U. Shardanand and P. Maes. Social information filtering algorithms for automating 'word of mouth'. In *Proceedings of ACM CHI'95 Conference on Human Factors in Computing Systems*, volume 1, pages 210–217, 1995.

[SRL97] M. D. Sammel, L. M. Ryan, and J. M. Legler. Latent variable models for mixed discrete and continuous outcomes. *Journal of the Royal Statistical Society*, Series B 59:667–678, 1997.

[SS02] B. Schölkopf and A. J. Smola. *Learning with Kernels*. MIT Press, 2002.

[TB99] M. E. Tipping and C. M. Bishop. Probabilistic principal component analysis. *Journal of the Royal Statisitical Scoiety*, B(61):611–622, 1999.

[TC01] S. Tong and E. Chang. Support vector machine active learning for image retrieval. In *Proceedings of ACM conference on Multimedia*, pages 107–118, Ottawa, Canada, 2001.

[Tip99] M. E. Tipping. Probabilistic visualization of high-dimensional binary data. In M. S. Kearns, S. A. Solla, and D. A. Cohn, editors, *Advances in Neural Information Processing Systems*, 11, pages 592–598. MIT Press, 1999.

[Ton01] S. Tong. *Active Learning: Theory and Applicaitons*. PhD thesis, Stanford University, 2001.

[UZ98] A. L. Uitdenbogerd and J. Zobel. Manipulation of music for melody matching. In *ACM Multimedia*, pages 235–240, 1998.

[Vap95] V. Vapnik. *The Nature of Statistical Learning Theory*. Springer, New York, 1995.

[XXY+03] Z. Xu, X. Xu, K. Yu, V. Tresp, and J. Wang. A hybrid relevance-feedback approach to text retrieval. In *Proceedings of the 25th European Conference on Information Retrieval Research (ECIR'03), Lecture Notes in Computer Science (LNCS 2633)*. Springer Verlag, 2003.

[XYT+03] Z. Xu, K. Yu, V. Tresp, X. Xu, and J. Wang. Representative sampling for text classification using support vector machines. In *Proceedings of the 25th European Conference on Information Retrieval Research (ECIR'03), Lecture Notes in Computer Science (LNCS 2633)*. Springer Verlag, 2003.

[YMT+03] K. Yu, W. Y. Ma, V. Tresp, Z. Xu, X. He, H. J. Zhang, and H.-P. Kriegel. Knowing a tree from the forest: Art image retrieval using a society of profiles. In *Proceedings of 11th Annual ACM International Conference on Multimedia (ACM Multimedia'03)*, Berkeley, California, November 2003.

[YST+03] K. Yu, A. Schwaighofer, V. Tresp, W. Y. Ma, and H. J. Zhang. Collaborative ensemble learning: Combining collaborative and content-based information filtering via hierarchical Bayes. In *Proceedings of the 19th Conference on Uncertainty in Artificial Intelligence (UAI)*, 2003.

[YST+04] K. Yu, A. Schwaighofer, V. Tresp, X. Xu, and H.-P. Kriegel. Probabilistic memory-based collaborative filtering. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 16(1):56–69, 2004.

[YT04] K. Yu and V. Tresp. Heterogenous data fusion via a probabilistic latent-variable model. In *Proceedings of 17th International Conference on Architecture of Computing Systems - Organic and Pervasive Computing (ARCS 2004), Lecture Notes in Computer Science (LNCS 2981)*, pages 20–30. Springer Verlag, 2004.

[YTY04] K. Yu, V. Tresp, and S. Yu. A nonparametric hierarchical Bayesian framework for information filtering. In *Proceedings of 27th Annual International ACM Conference on Research and Development in Information Retrieval (SIGIR 2004)*, 2004.

[YWXE01] K. Yu, Z. Wen, X. Xu, and M. Ester. Feature weighting and instance selection for collaborative filtering. In *Proceedings of the 2nd Int. Workshop on Management of Information on the Web - Web Data and Text Mining (MIW'01)*, pages 285–290, 2001.

[YXEK01] K. Yu, X. Xu, M. Ester, and H.-P. Kriegel. Selecting relevant instances for efficient and accurate collaborative filtering. In *Proceedings of the ACM 10th Int. Conf. on Information and Knowledge Management (CIKM'01)*, pages 247–254, 2001.

[YXEK03] K. Yu, X. Xu, M. Ester, and H.-P. Kriegel. Feature weighting and instance selection for collaborative filtering: An information-theoretic approach. *International Journal of Knowledge and Information Systems (KAIS)*, 5(2), Springer Verlag, April 2003.

[YXS+02] K. Yu, X. Xu, A. Schwaighofer, V. Tresp, and H.-P. Kriegel. Removing redundancy and inconsistency in memory-based collaborative filtering. In *Proceedings of the ACM 11th Int. Conf. on Information and Knowledge Management (CIKM'02)*, McLean, VA, November 2002.

[YXT+02] K. Yu, X. Xu, J. Tao, M. Ester, and H.-P. Kriegel. Instance selection techniques for memory-based collaborative filtering. In *Proceedings of the 2nd SIAM Int. Conf. on Data Mining (SDM'02)*, Arlington, VA, 2002.

[ZI02] T. Zhang and V. S. Iyengar. Recommender systems using linear classifiers. *Journal of Machine Learning Research*, 2:313–334, 2002.