Inf Retrieval (2006) 9:357–382
DOI 10.1007/s10791-006-4651-1

A study of mixture models for collaborative filtering

Rong Jin · Luo Si · Chengxiang Zhai

Received: 20 July 2004 / Revised: 1 May 2005 / Accepted: 29 August 2005

Abstract Collaborative filtering is a general technique for exploiting the preference patterns of a group of users to predict the utility of items for a particular user. Three different components need to be modeled in a collaborative filtering problem: users, items, and ratings. Previous research on applying probabilistic models to collaborative filtering has shown promising results. However, there is a lack of systematic studies of different ways to model each of the three components and their interactions. In this paper, we conduct a broad and systematic study on different mixture models for collaborative filtering. We discuss general issues related to using a mixture model for collaborative filtering, and propose three properties that a graphical model is expected to satisfy. Using these properties, we thoroughly examine five different mixture models, including Bayesian Clustering (BC), Aspect Model (AM), Flexible Mixture Model (FMM), Joint Mixture Model (JMM), and the Decoupled Model (DM). We compare these models both analytically and experimentally. Experiments over two datasets of movie ratings under different configurations show that in general, whether a model satisfies the proposed properties tends to be correlated with its performance. In particular, the Decoupled Model, which satisfies all the three desired properties, outperforms the other mixture models as well as many other existing approaches for collaborative filtering. Our study shows that graphical models are powerful tools for modeling collaborative filtering, but careful design is necessary to achieve good performance.

Keywords Collaborative filtering · Graphical model · Probabilistic model

Rong Jin
Department of Computer Science and Engineering, Michigan State University, East Lansing, MI 48824, USA
e-mail: rongjin@cse.cmu.edu

Luo Si (✉)
Department of Computer Science, Purdue University, West Lafayette, IN 47907-1398, USA
e-mail: lsi@cs.cmu.edu

Chengxiang Zhai
Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL 61801, USA
e-mail: czhai@cs.uiuc.edu

1. Introduction

The rapid growth of information over Internet demands intelligent information agents that can sift through all the available information and find out the most valuable to us. These intelligent systems can be categorized into two classes: Collaborative Filtering (CF) systems and Content-based Filtering (CBF) systems. The difference between them is that collaborative filtering systems utilize the given ratings of training users to make recommendation for test users while content-based filtering systems rely on contents of items for recommendation. In this paper, we focus on the collaborative filtering systems.

Most collaborative filtering methods fall into two categories: Memory-based algorithms and Model-based algorithms (Breese et al., 1998). In memory-based algorithms, rating examples of different users are simply stored in a training database, and the rating of a test user on a specific item is predicted based on the corresponding ratings of training users who share similar tastes as the test user. In contrast, in model-based algorithms, statistical models are learned from the given ratings of training users and ratings of test users are estimated using the learned model. In the previous studies, both types of approaches have been shown to be effective for collaborative filtering (Breese et al., 1998).

In general, most collaborative filtering approaches assume that users with similar "tastes" would rate items similarly, and the idea of clustering has been exploited in all approaches either explicitly or implicitly. Compared with memory-based approaches, model-based approaches provide a more principled way of performing clustering, and is also often much more efficient in terms of the computation cost at the prediction time. The basic idea of a model-based approach is to cluster items and/or training users into classes explicitly and predict ratings of a test user using the ratings of classes that fit in well with the test user and/or items.

Several different probabilistic models have been proposed and studied in the previous work (Breese et al., 1998; Hofmann and Puzicha, 1998; Pennock et al., 2000; Popescul et al., 2001; Ross and Zemel 2002; Si et al., 2003; Jin et al., 2003; Hofmann, 2003). These models have succeeded in capturing user/item similarities through probabilistic clustering in one way or the other, and have all been shown to be quite promising. Most of these methods can be represented as graphical models. However, there has been no systematic study and comparison of different graphical models proposed for collaborative filtering, which is necessary for both theoretical and empirical reasons: (1) Theoretically, different models make different assumptions. We need to understand the difference and connections among these models in terms of the underlying assumptions. (2) Empirically, these different models are evaluated with different experimental settings in previous studies; it would be useful to see how they are compared with each other using *identical* experimental settings. Moreover, a systematic study is necessary for explaining why some models tend to perform better than others.

In this paper, we conduct a systematic study of a large subset of graphical models – mixture models – for collaborative filtering. One of the fundamental difficulties with collaborative filtering is the sparse data issue. It arises when most users only provide ratings for a small number of items. As a result, even if two users have similar interests, there may be no common items rated by both of them. Mixture models are the natural remedy to the sparse data problem. By grouping items with similar ratings into clusters, the mixture models are able to estimate the similarity among different users based on their ratings on item clusters, not individual items. Since three components, namely users, items, and ratings, involved in the collaborative filtering, a good mixture model for collaborative filtering should be able to not only cluster each component, but also model the interactions between different components

appropriately. We propose three desirable properties that a reasonable graphical model for collaborative filtering should satisfy: (1) separate clustering of users and items; (2) flexibility for a user/item to be in multiple clusters; (3) decoupling of user preference from its rating patterns.

We thoroughly analyze five different mixture models, including Bayesian Clustering (Breese et al., 1998), Aspect Model (Hofmann and Puzicha, 1999), Flexible Mixture Model (Si et al., 2003), Joint Mixture Model (JMM) and the Decoupled Model (Jin et al., 2003) based on the three proposed properties. We also compare these models empirically. Experiments over two datasets of movie ratings under several different configurations show that in general, the fulfillment of the proposed properties tends to be positively correlated with the model's performance. In particular, the DM model, which satisfies all the three properties that we want, outperforms all the other mixture models as well as some other existing approaches to collaborative filtering. Our study shows that graphical models are powerful tools for modeling collaborative filtering, but careful design is necessary in order to achieve good performance.

The rest of the paper is arranged as follows: Section 2 gives a general discussion of using graphical models for collaborative filtering and presents the three desirable properties that any graphical model should satisfy. In Section 3, we present and examine five different mixture models in terms of their connections and differences. We discuss model estimation and rating prediction in Section 4 and Section 5. Empirical studies are presented in Section 6. Conclusions and future work are discussed in Section 7.

2. Graphical models for collaborative filtering

2.1. Problem definition

We first introduce notations for formally describing the problem of collaborative filtering. Let $\tilde{\mathbf{X}} = \{x_1, x_2, \ldots\ldots, x_M\}$ be a set of items, $\tilde{\mathbf{Y}} = \{y_1, y_2, \ldots\ldots, y_N\}$ be a set of users, and $\{1, \ldots, R\}$ be the set of possible ratings. Let $\{(x_{(1)}, y_{(1)}, r_{(1)}), \ldots\ldots, (x_{(L)}, y_{(L)}, r_{(L)})\}$ be the training database that consists of ratings of different items from multiple training users. Each tuple $(\mathrm{x}_{(i)}, \mathrm{y}_{(i)}, \mathrm{r}_{(i)})$ represents that item $x_{(i)}$ is rated as $r_{(i)}$ by user $y(i)$. Let $R_y(x)$ be the rating of item x given by user y and $X(y)$ be the set of items rated by user y. In addition to the training database, each test user also provides a small number of ratings to indicate his/her interests and preference. The goal of collaborative filtering is to predict the rating r that a test user would give to an unrated item x given the training database and the additional rating information from the test user.

To cast this problem into graphical models, we treat each tuple $(\mathrm{x}_{(i)}, \mathrm{y}_{(i)}, \mathrm{r}_{(i)})$ as an observation that is randomly drawn from the joint distribution of three random variables – **X**, **Y**, and **R**. Random variables X and Y can take any value from the sets $\tilde{\mathbf{X}}$ and $\tilde{\mathbf{Y}}$, respectively. Random variable **R** will take any integer value ranging from 1 to R. Through the training database, we are able to model the interaction between the three random variables. There are three possible choices of likelihood that we can maximize for the training data: $p(r|x, y)$, $p(r, x \mid y)$ and $p(r,x,y)$. Although there is strong correlation between these quantities, maximizing data with a different likelihood models different aspects of the data. For the first choice, i.e., $p(r \mid x, y)$, we focus on modeling why item x is rated by user y as r.

The second choice, i.e., $p(r, x \mid y)$, differs from the first one in that it explains not only the observed ratings but also why item x is chosen to be rated by user y. As a result, movies that have been rated by many users will have more impact on the model estimation than movies that are only rated by a few users. The third choice, i.e., $p(r,x,y)$, models the joint distribution

between the three random variables. Under this choice, the model is also concerned with the behavior of users (e.g. some users rate a lot of movies and others only rate a few). In particular, users with more ratings tend to have larger impact on the final model than users that only rate a few items. Based on the above discussion, it is clear that the choice of likelihood function for training data can have a significant impact on model estimation and thus the performance of collaborative filtering. Most existing probabilistic approaches to collaborative filtering fall into one of these three cases. For example, the personality diagnosis method (Pennock et al., 2000) is a special case of the first one, where a Gaussian distribution is assumed for $p(r \mid x, y)$. The aspect model (Hofmann and Puzicha, 1999) can be regarded as a special case of the third choice, where a mixture model is used for estimating $p(r,x,y)$. In this paper, we will focus on the second and third cases and systematically examine the different choices of mixture models.

Note that we intentionally ignore the possibility of modeling conditional probability $p(r, y \mid x)$. This is because in collaborative filtering, it is the users who actively select items to rate, not the other way.

2.2. Major issues in designing a graphical model for collaborative filtering

In general, in order to model the similarity among different users, items and ratings given the difficulty of sparse ratings provided by users, we need to cluster each component into groups and model the interactions between different components appropriately. More specifically, the following three important issues must be addressed:

Issue 1: How should we model user similarity and item similarity? Generally, we may regard users and items as being from different types of entities and they couple with each other through rating information. Therefore, a good clustering model for collaborative filtering is expected to explicitly model both the classes of users and the classes of items and be able to leverage their correlations. This means that the choice of latent variables in our graphical model should allow for separate, yet coupled modeling of user similarity and item similarity. Of course, the separation of user similarity from item similarity will lead to complex clustering models that can be hard to estimate accurately with the data available. Models with different clustering strategies will be examined in this paper.

Issue 2: Should a user or an item be allowed to belong to multiple clusters? Since a user can have diverse interests and an item may have multiple aspects, intuitively, it is desirable to allow both items and users to be in multiple classes simultaneously. However, such a model may be too flexible to capture the similarity of users and items effectively with a limited amount of training data. Models with different assumptions about the membership of users and items will be examined in this paper.

Issue 3: How can we capture the variances in rating patterns among the users with similar interest of items? One common deficiency in most existing models for collaborative filtering is that they are all based on the assumption that users with similar interests would rate items similarly. This is incorrect because the rating pattern of a user is determined not only by his/her interests but also by the rating strategy/habit. For example, some users are more "tolerant" than others, and therefore their ratings of items tend to be higher than others although they share very similar tastes of items. Thus, it is important for a collaborative filtering method to capture the variance among rating patterns of users with similar interest of items. Methods for modeling such variances in a graphical model will be examined in this paper.

Based on the discussion above, we identify three important properties that a graphical model for collaborative filtering should satisfy:

- It should support clustering of both users and items
- It should allow both users and items to be in multiple clusters
- It should decouple the rating patterns from intrinsic preferences

In the following section, we will analyze a representative set of mixture models and examine them in terms of these desirable properties.

3. Mixture models for collaborative filtering

In this section, we discuss a variety of possible mixture models and examine their assumptions about user and item clustering and whether they address the variances in rating patterns.

3.1. Bayesian clustering (Breese et al., 1998)

In Bayesian Clustering (Breese et al., 1998), we assume that the same type of users would rate items similarly, thus users can be automatically grouped together into a set of user clusters, or user classes, according to their ratings of items. Formally, given a user class 'z', the preferences for different items expressed as ratings are independent, and the joint probability of user class 'z' and the ratings of items can be written as the standard naïve Bayes formulation:

$$P(z, r_1, r_2, \ldots, r_M) = P(z) \prod_{i=1}^{M} P(r_i|z) \tag{1}$$

where r_i represents the rating for item x_i. Thus, the joint probability for the ratings given by user y, denoted as $P(\{R_y(x)\}_{x \in X(y)}|y)$, can be written as:

$$P(\{R_y(x)\}_{x \in X(y)}|y) = \sum_{z} P(z) \prod_{x \in X(y)} P(R_y(x)|z) \tag{2}$$

According to Equation (2), this model will first select a user class 'z' from the distribution $P(z)$ and then rate all the items using the *same selected class* 'z'. In another word, this model assigns each user to a single user class and therefore does not allow a user to be in multiple user classes. Parameters $P(r \mid z)$ can be learned automatically using the Expectation-Maximization (EM) algorithm. More details of this model can be found in (Breese et al., 1998).

According to the three criteria mentioned in the previous section, Bayesian Clustering appears to be the simplest mixture model: only cluster the users; each user is assumed to be in a single cluster; no separation for preference and rating patterns. Figure 1 illustrates the basic idea of the Bayesian Clustering.

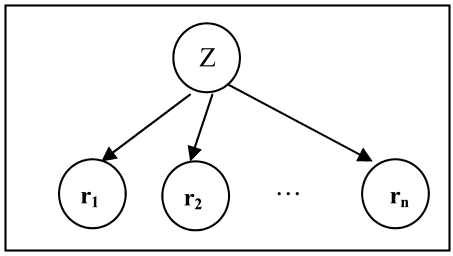

Fig. 1 Graphical model representation for bayesian Clustering

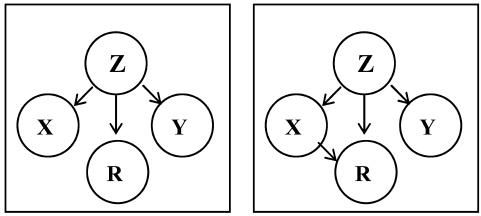

Fig. 2 Graphical models for the two extensions of aspect model in Eqs. (4) and (5)

3.2. Aspect model (AM)

The aspect model is a probabilistic latent space model, which models individual preferences as a convex combination of preference factors (Hofmann and Puzicha 1999). It introduces a latent variable $z \in \mathbf{Z} = \{z_1, z_2, \ldots\ldots, z_K\}$ for each user-item pair (x, y), and writes the joint probability for each pair as:

$$P(x, y) = \sum_{z \in Z} P(z)\, P(x \mid z)\, P(y \mid z) \tag{3}$$

where $P(z)$ is the class prior probability, $P(x \mid z)$ and $P(y \mid z)$ are class-dependent distributions for items and users, respectively. Intuitively, this model means that the preference pattern of a user is modeled by a combination of typical preference patterns, which are represented in the distributions of $P(z)$, $P(x \mid z)$ and $P(y \mid z)$.

There are two ways to incorporate rating information 'r' into the basic aspect model:

$$P\left(x_{(l)}, y_{(l)}, r_{(l)}\right) = \sum_{z \in Z} P(z)\, P\left(x_{(l)} \mid z\right) P\left(y_{(l)} \mid z\right) P\left(r_{(l)} \mid z\right) \tag{4}$$

$$P\left(x_{(l)}, y_{(l)}, r_{(l)}\right) = \sum_{z \in Z} P(z)\, P\left(x_{(l)} \mid z\right) P\left(y_{(l)} \mid z\right) P\left(r_{(l)} \mid z, x_{(l)}\right) \tag{5}$$

The corresponding graphical models are shown in Fig. 2. Compared to the first approach in Eq. (4), the second approach in Eq. (5) has to estimate the conditional probability $P(r_{(l)} \mid z, x_{(l)})$, which corresponds to a larger parameter space and may not be estimated reliably.

Unlike the Bayesian Clustering algorithm, which only models ratings, the aspect model is able to model both users and items with conditional probabilities $P(y \mid z)$ and $P(x \mid z)$. Furthermore, unlike the Bayesian Clustering algorithm, where the joint probability for a set of ratings by an individual user is modeled directly, the aspect model models the joint probability $P(x,y,r)$ separately for each rated item. As a result, the aspect model allows each rating triplet to choose its own appropriate class while in Bayesian Clustering the same user class is used to rate all the items. However, the aspect model only introduces a single set of class variables for items, users, and ratings. This essentially encodes the clustering of users, the clustering of items, and the correlation between them *together*, thus the separate clustering of users and items is not attempted. Furthermore, no efforts have been made in aspect model to separate users' rating patterns from their intrinsic interests. Therefore, according to the criterion stated in Section 2.2, the aspect model is still a preliminary model: a simple way to model users and items but without clustering them separately; allowing each user and item to be in multiple clusters; no attempt for modeling intrinsic preference of users separately from their rating patterns.

3.3. Joint Mixture Model (JMM) and Flexible Mixture Model (FMM)

In this section, we examine two additional graphical models for collaborative filtering, namely Joint Mixture Model (JMM) and Flexible Mixture Model (FMM) (Si et al., 2003). They differ from both the Bayesian Clustering algorithm and the Aspect Model in that users and items are clustered separately.

For both graphical models, the goal is to model the joint probability $P(\{R_y(x)\}_{x\in X(y)} \mid y)$. They differ in the way of decomposing the joint probability. In the Joint Mixture Model, the joint probability is expanded as:

$$P(\{R_y(x)\}_{x\in X(y)}|y) = \sum_{z_y} P(z_y|y) \prod_{x\in X(y)} P(x, R_y(x)|z_y) \tag{6}$$

where variable z_y stands for the class for user 'y'. According to Eq. (6), to estimate the joint probability, user class z_y is first chosen according to distribution $P(z_y|y)$, and then the likelihood of every rated item is computed using the same user class z_y. Thus, similar to the Bayesian clustering algorithm, the Joint Mixture Model assumes that each user belongs to a single user class. In contrast, the Flexible Mixture Model first expands $P(\{R_y(x)\}_{x\in X(y)}|y)$ into a product of likelihoods for rated items, followed by the introduction of hidden variables for user class for each item:

$$P(\{R_y(x)\}_{x\in X(y)}|y) = \prod_{x\in X(y)} \sum_{z_y} P(x, R_y(x)|z_y)P(z_y|y) \tag{7}$$

Compared to Joint Mixture Model, the Flexible Mixture Model allow each item to choose the appropriate user class for its rating while the Joint Mixture Model enforces a single user class to be used throughout the ratings of every user.

The key component for both models is the estimation of $P(x, r|z_y)$, i.e., the likelihood for the user class z_y to rate item x as r. Directly estimating $P(x, r|z_y)$ from training data may lead to a severe sparse data problem. This is because the number of different $P(x, r|z_y)$ can be quite big given a large number of items and user classes. To alleviate this problem, hidden variable z_x is further introduced to represent the classes for items, which leads to a new expression for $P(x, r|z_y)$:

$$P(x, r|z_y) = \sum_{z_x} P(x, r, z_x|z_y) \approx \sum_{z_x} P(z_x)P(x|z_x)P(r|z_x, z_y) \tag{8}$$

where $P(z_x)$ is the class prior for item class z_x and $P(r|z_x, z_y)$ is the likelihood for user class z_y to rate item class z_x as r. The above expression assumes that the class variable z_x for items is independent from the class variable z_y for users, or $P(z_x|z_y) \approx P(z_x)$. Through the introduction of item class, the number of parameters for $P(x, r|z_y)$ is decreased from $M \times R \times |Z_y|$ to $|Z_x| \times (1 + M + |Z_y| \times R)$, where $|Z_y|$ and $|Z_x|$ are the number of classes for users and items respectively, and M and R are the number of items and rating categories, respectively. This is a significant reduction when the number of user classes is large. For example, given 1000 different movies, 5 different rating categories, and 20 different user types, the number of parameters for $P(x, r|z_y)$ is 100,000. However, by grouping movies into 10 different classes, the number of parameters drops to around 11,000, which is only one tenth of the original parameter space. Of course, the introduction of item classes may flat the difference between similar items and thus lead to errors in predicting ratings. This is the tradeoff between alleviating data sparseness and maintaining data diversity. We will examine

Table 1 Parameters for the Joint Mixture Model (JMM) and the Flexible Mixture Model (FMM)

$P(z_y \mid y)$	Likelihood of assigning user 'y' to the user class z_y
$P(z_x)$	Class prior for item class z_x
$P(x \mid z_x)$	Likelihood for item x to be in class z_x
$P(r \mid z_x, z_y)$	Likelihood for any user in class z_y to rate any item in class z_x as 'r'

this issue later in experiments. Table 1 summarizes the parameters used by both models and the diagrams of corresponding graphical models are displayed in Fig. 3.

As for the three properties in Section 2.2, both models apply separate clustering to users and items and thus satisfy the property 1. The Flexible Mixture Model satisfies the second property since it leaves each rated item the freedom to choose the appropriate user class while the Joint Mixture Model does not. Neither of the two models makes any attempt to explicitly model the difference between the rating patterns and the intrinsic preference of users.

3.4. Decoupled models for rating patterns and intrinsic preference (DM)

All mixture models that have been discussed so far fail to explicitly account for the fact that users with similar interests may have very different rating patterns. In this section, we discuss decoupled model (Jin et al., 2003), which extends the Flexible Mixture Model by introducing two hidden variables Z_P and Z_R that account for rating patterns and intrinsic preference of users, respectively. Figure 4 displays the graphical representation for the decoupled model. According to Fig. 4, the decoupled model first determines the class Z_x for item 'X', the class Z_P and Z_R for user 'Y'. Class Z_P accounts for the intrinsic preference of user 'Y', namely, the types of items that 'Y' likes and the types of items that he/she does not like. Class Z_R accounts for the rating patterns of user 'Y', namely how user 'Y' rates items according to his interests. Unlike the previous mixture models where the user type is modeled by a single class variable Z_y, in this model, users are clustered from two different perspectives, i.e., the clustering of intrinsic preference by hidden variable Z_P and the clustering of rating patterns (or habits) by hidden variable Z_R. To decide the rating category for item 'X', the new model first determines the value of the binary random variable Z_{pref} that indicates whether user 'Y'

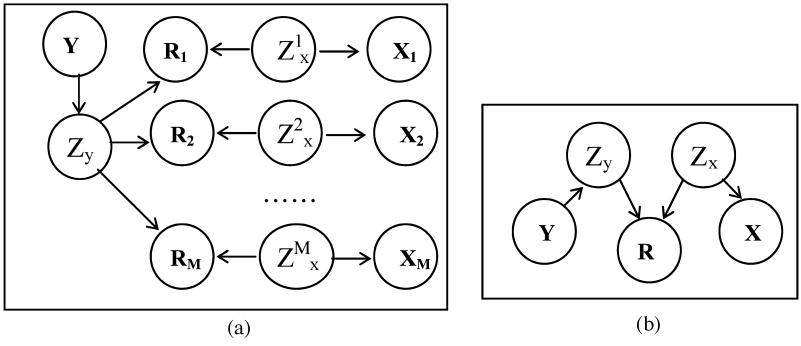

Fig. 3 Graphical model representation for the Joint Mixture Model and Flexible Mixture Model. Diagram (a) represents the joint mixture model (JMM) and (b) for flexible mixture model (FMM)

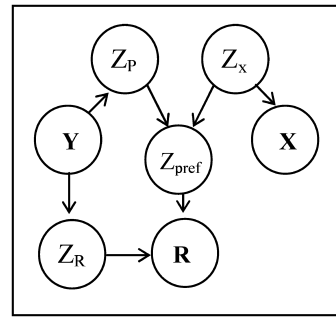

Fig. 4 Graphical model representation for the decoupled model (DM)

likes item 'X', and the rating variable 'R' is jointly determined by the preference variable Z_{pref} and the rating class Z_R of user 'Y'. Thus, the actual rating value is affected not only by whether the user likes an item (i.e., Z_{pref}), but also by the specific rating patterns of the user (i.e., Z_R). Therefore, even if a user appears to like a certain type of items, the rating value can still be low if he has a very 'tough' rating criterion. In summary, the new model has addressed the problem of large rating variance among users of similar interests with two heuristics:

1. It models the rating patterns and intrinsic preference of users separately;
2. The rating category of an item is decided not only by whether a user likes the item but also by the rating strategy of the user.

Note that the DM model satisfies *all* the three desirable properties: cluster users and items separately; allow each user to be in multiple clusters; and model the difference between preference patterns and rating patterns.

Following the above description, probability $P(x, r|y)$ is expressed as follows:

$$P(x, r|y) = \sum_{z_P, z_R, z_x} P(z_P|y)P(z_R|y)P(z_x)P(x|z_x) \left\{ \sum_{Z_{pref}=\{0,1\}} P(z_{pref}|z_P, z_x)P(r|z_R, z_{pref}) \right\} \tag{9}$$

where $P(z_{pref}|z_P, z_x)$ is the likelihood for users in class Z_P to like ($Z_{pref} = 1$) or dislike ($Z_{pref} = 0$) items in class Z_x, $P(r|z_R, z_{pref})$ is the likelihood for users in class Z_R to give rating 'r' given that they like ($Z_{pref} = 1$) or dislike ($Z_{pref} = 0$) the items. Combining Eq. (7) with Eq. (9), we have the full description for the decoupled model that determines the likelihood for a rating database. Table 2 summarizes the parameters for the decoupled model. Compared with Table 1, more parameters are introduced in the decoupled model to account for the new variables Z_P, Z_R, and Z_{pref}, which will raise the complexity of the model and thus has more chance to over-fit training data.

The decoupled model can be further improved by extending the binary hidden variable Z_{pref} to a variable with multiple values. Thus, instead of indicating whether or not a user likes an item, hidden variable Z_{pref} represents the level of preference that the user has for the item. For example, we can let the variable Z_{pref} have three discrete values, with zero for no preference, one for slight preference and two for strong preference. In our experiments, the

Table 2 Parameters for the decoupled model (DM)

$P(z_P \mid y)$	Likelihood of assigning user 'y' to the preference class z_P
$P(z_R \mid y)$	Likelihood of assigning user 'y' to the rating class z_R
$P(z_x)$	Class prior for item class z_x
$P(x \mid z_x)$	Likelihood for item x to be in item class z_x
$P(z_{pref} = 1 \mid z_P, z_x)$	Likelihood for users in preference class z_P to favor items in class z_x
$P(r \mid z_R, z_{pref})$	Likelihood for users in rating class z_R to rate items in class z_x as 'r' given that they either like the items (i.e., $Z_{pref} = 1$) or dislike the items (i.e., $Z_{pref} = 0$)

number of discrete values for variable Z_{pref} is set to be equal to the number of different rating categories. In this case, Eq. (9) will be rewritten as:

$$P(x, r \mid y) = \sum_{z_P, z_R, z_x} P(z_P \mid y) P(z_R \mid y) P(z_x) P(x \mid z_x) \left\{ \sum_{z_{pref}=1}^{R} P(z_{pref} \mid z_P, z_x) P(r \mid z_R, z_{pref}) \right\} \tag{9'}$$

Note that by setting $P(r \mid z_R, z_{pref}) = \delta(r, z_{pref})$, the above equation will be turned into Eq. (8), which leads to the Flexible Mixture Model. Thus, the extended decoupled model is a more general framework than the Flexible Mixture Model.

3.5. Summary and comparison

In this section, we have discussed five different mixture models:

- *Bayesian clustering* is the simplest approach and does not satisfy any of the three properties. It makes no effort to model either users or items and each user is restricted to a single class.
- *Aspect model* improves over Bayesian clustering by introducing a hidden variable that models the interaction between users and items. It satisfies the second property by allowing each user to be in multiple different classes. However, it does not apply separate clustering to users and items, and it does not address the problem of rating variance within users of similar interests. Therefore, it violates both the first and the third properties.
- Both the *Joint Mixture Model* (JMM) and *Flexible Mixture Model* (FMM) emphasize separate clustering of users and items. They differ from each other in that FMM allows each user to be in multiple classes while JMM restricts every user to be in a single class. Thus, FMM satisfies both the first and the second property while JMM only satisfies the first one.
- The *Decoupled Model* (DM) extends the Flexible Mixture Model (FMM) by separating the intrinsic preference of users from their rating strategies. In particular, the final rating of an item is affected not only by the interest of a user but also by his rating criteria. Thus, the decoupled model satisfies all three properties.

Table 3 summarizes the properties of each model. On one hand, we expect models that satisfy more properties to provide better description for the data and achieve more accurate prediction. On the other hand, to satisfy more properties, we have to increase the model complexity, which could degrade the accuracy of prediction particularly when the number of training users is small. As will be seen later in the experiment section, with a large number

Table 3 Properties of five different mixture models for collabosrative filtering

	Property 1	Property 2	Property 3
Bayesian Clustering (BC)	–	–	–
Aspect Model (AM)	–	x	–
Joint Mixture Model (JMM)	x	–	–
Flexible Mixture Model (FMM)	x	x	–
Decouple Model (DM)	x	x	x

Property 1 corresponds to separate clustering for users and items; Property 2 corresponds to the flexibility for a single user or an item to be in multiple clusters; Property 3 corresponds to the capture of difference between intrinsic preference and rating patterns.

of training users, models satisfying more properties usually perform better than models satisfying fewer properties. However, when the number of training users is small, the simple model may perform even better.

4. Model estimation

In this section, we describe the general Expectation Maximization (EM) algorithm that is used to estimate the mixture models for collaborative filtering, followed by the description of the prediction algorithms.

4.1. General approach–EM algorithms

In general, all the mixture models can be estimated using the EM algorithm (Demspter et al., 1977). As an example, we give details on the EM algorithm for the Joint Mixture Model (JMM), which is slightly more complicated than the others.

According to the maximum likelihood approach, parameters of JMM model are estimated by maximizing the log-likelihood of training data, which is written as

$$L = \sum_{y} \log P(\{x, R_y(x)\}_{x \in X(y)} \mid y) \tag{10}$$

Expand $P(\{x, R_y(x)\}_{x \in X(y)} \mid y)$ using Eqs. (6) and (8), we have

$$L = \sum_{y} \log \left\{ \sum_{z_y} P(z_y \mid y) \prod_{x \in X(y)} \sum_{z_x} P(z_x) P(x \mid z_x) P(R_y(x) \mid z_x, z_y) \right\} \tag{11}$$

To optimize the above objective function, the EM algorithm alternates between the expectation step and maximization step. In the expectation step, the posterior probabilities for latent variables, i.e., $P(z_y \mid \{x, R_y(x)\}_{x \in X(y)}, y)$ and $P(z_x \mid x, R_y(x), y, z_y)$, are computed as follows:

$$P(z_y \mid \{x, R_y(x)\}_{x \in X(y)}, y) = \frac{P(z_y \mid y) \prod_{x \in X(y)} P(x, R_y(x) \mid z_y)}{\sum_{z_y} P(z_y \mid y) \prod_{x \in X(y)} P(x, R_y(x) \mid z_y)} \tag{12}$$

$$P(z_x \mid x, R_y(x), y, z_y) = \frac{P(z_x) P(x \mid z_x) P(R_y(x) \mid z_x, z_y)}{\sum_{z'_x} P(z'_x) P(x \mid z'_x) P(R_y(x) \mid z'_x, z_y)} \tag{13}$$

In the maximization step, model parameters $P(z_y \mid y)$, $P(z_x)$, $P(x \mid z_x)$, and $P(r \mid z_x, z_y)$ are updated using the posterior probabilities that are estimated in the expectation step:

$$P(z_y \mid y) = P(z_y \mid \{x, R_y(x)\}_{x \in X(y)}, y) \tag{14}$$

$$P(z_x) = \frac{\sum_{z_y} \sum_y \sum_{x \in X(y)} P(z_x \mid x, R_y(x), y, z_y) P(z_y \mid y)}{\sum_{z_x} \sum_{z_y} \sum_y \sum_{x \in X(y)} P(z_x \mid x, R_y(x), y, z_y) P(z_y \mid y)} \tag{15}$$

$$P(x \mid z_x) = \frac{\sum_{z_y} \sum_y \sum_{x' \in X(y)} P(z_x \mid x', R_y(x'), y, z_y) P(z_y \mid y) \delta(x = x')}{\sum_{z_y} \sum_y \sum_{x' \in X(y)} P(z_x \mid x', R_y(x'), y, z_y) P(z_y \mid y)} \tag{16}$$

$$P(r \mid z_x, z_y) = \frac{\sum_y \sum_{x' \in X(y)} P(z_x \mid x', R_y(x'), y, z_y) P(z_y \mid y) \delta(R_y(x') = r)}{\sum_y \sum_{x' \in X(y)} P(z_x \mid x', R_y(x'), y, z_y) P(z_y \mid y)} \tag{17}$$

4.2. Smoothing mixture models

The EM algorithm is notorious for finding undesirable local optimal solutions. In this section, we discuss two techniques that can help avoid unfavorable solutions, both aiming at regularizing the EM algorithm in some way. Again, we use the JMM as an example, but the smoothing techniques can be applied to other models as well.

The first technique is called Annealed EM algorithm (AEM) (Hofmann and Puzicha, 1998). The idea can be described as follows: to prevent posterior distributions from being skewed at the early stage of EM iterations, a regularization variable 'b' is introduced into the expectation step. According to the Annealed EM algorithm, 'annealed' posteriors for the JMM model are written as:

$$P(z_y \mid \{x, R_y(x)\}_{x \in X(y)}, y) = \frac{\left[P(z_y \mid y) \prod_{x \in X(y)} P(x, R_y(x) \mid z_y)\right]^b}{\sum_{z_y} \left[P(z_y \mid y) \prod_{x \in X(y)} P(x, R_y(x) \mid z_y)\right]^b} \tag{18}$$

$$P(z_x \mid x, R_y(x), y, z_y) = \frac{[P(z_x) P(x \mid z_x) P(R_y(x) \mid z_x, z_y)]^b}{\sum_{z'_x} [P(z'_x) P(x \mid z'_x) P(R_y(x) \mid z'_x, z_y)]^b} \tag{19}$$

Note that when $b = 1$, the above equations return back to Eqs. (12) and (13), which correspond to the expectation step of the normal EM algorithm. On the other hand, when $b = 0$, the posteriors estimated in Eqs. (18) and (19) become uniform distributions that completely ignore any training data. By varying b between 0 and 1, we are able to adjust posteriors between uniform distributions and the distributions that are estimated from training data. In the Annealed EM algorithm, parameter b is increased slowly from 0 to 1. Thus, the posteriors initially start as uniform distributions. With the increasing value for b, the posteriors are more influenced by training data and move away from uniform distributions. The purpose of slowly increasing parameter b is to let the information from training data gradually being transferred into the model, which prevents the model from committing to training data at early stage and thereby helps the EM algorithm avoid undesirable local optimum. By viewing b as the inverse of so-called 'temperature', the process of increasing b is analogous to the annealing process that slowly drops system temperature.

The second smoothing strategy is to introduce a model prior into the mixture model. The mixture models presented in previous sections are based on the maximum likelihood

estimation, which determines the model parameters by maximizing the likelihood of training data. To regularize the mixture model, this approach maximizes the posterior of training data, which is a product of the prior and the likelihood. It is also called maximum a posterior approach (MAP). Compared to the maximum likelihood approach, MAP has the advantage in that the choice of parameters is affected not only by training data but also the prior preference of mixture models. It is particularly useful when the amount of training data is insufficient for learning a reliable model. Since a uniform distribution is our best guess when no rating information is exposed, in general, an appropriate prior should favor distributions with equal probabilities. One choice of such a prior is the Dirichlet prior with uniform means. Using the JMM as an example, the Dirichlet prior for the JMM can be written as:

$$P(\boldsymbol{\theta} \mid a, b, c, d) \propto \left\{\prod_{z_x} P(z_x)\right\}^a \left\{\prod_{x,z_x} P(x \mid z_x)\right\}^b \left\{\prod_{y,z_y} P(y \mid z_y)\right\}^c \left\{\prod_{z_x,z_y,r} P(r \mid z_x, z_y)\right\}^d \tag{20}$$

where $\boldsymbol{\theta} = \{P(z_x), P(x \mid z_x), P(z_y \mid y), P(r \mid z_x, z_y)\}$ represents the parameter space for the JMM, and a, b, c, and d are hyper parameters for Dirichlet distribution. Combining the prior in the above equation with the likelihood function in Eq. (11), we have the objective function for MAP written as:

$$\begin{aligned} L = &\sum_y \log \left\{\sum_{z_y} P(z_y \mid y) \prod_{x \in X(y)} \sum_{z_x} P(z_x) P(x \mid z_x) P(R_y(x) \mid z_x, z_y)\right\} \\ &+ a \sum_{z_x} \log P(z_x) + b \sum_{x,z_x} \log P(x \mid z_x) + c \sum_{y,z_y} \log P(z_y \mid y) + d \sum_{r,z_x,z_y} \log P(r \mid z_x, z_y) \end{aligned} \tag{21}$$

Our goal is to find the parameters that maximize the above objective function. In the above equation, hyper parameters a, b, c, and d have played the role of regularization, and their values reflect our confidence on the prior preference of the mixture models. When we are unconfident about the prior preference, all the hyper parameters will be set small and the resulting parameters will be mainly determined by the likelihood term. On the other hand, when we are confident about the prior preference, all the hyper parameters will be set large and the resulting parameters will be mainly determined by the prior term. Thus, by adjusting hyper parameters a, b, c, and d, we are able to make appropriate tradeoff between the training data and the prior knowledge of the models. The detailed EM algorithm for maximizing the objective function in Eq. (21) is listed in Appendix A.

5. Rating prediction

To predict the ratings of items for a test user y^t, we need to estimate distributions of latent variables that are related to the test user. In addition to the ratings provided by training users, each test user also provides a small number of rated items that can be utilized to discover distributions of related latent class variables for the test user. Let D_{train} and D_{test} stand for the rating data for training users and test user y^t, respectively. Let $\{h_i\}_{i=1}^m$ be the hidden variables. Let $\boldsymbol{\theta}_{train}$ and $\boldsymbol{\theta}_{test} = \{P(h_i \mid y^t)\}_{i=1}^m$ represent the parameter space that is related to training users and the test user, respectively. In order to predict the rating of an item x by the test

user y^t, we need to estimate the likelihood $P(r \mid D_{train}, D_{test}, x)$ for each rating category 'r', which can be approximated as follows:

$$\begin{aligned} P(r \mid D_{train}, D_{test}, x) &= \sum_{M_{test}} \sum_{M_{train}} P(r \mid \boldsymbol{\theta}_{test}, \boldsymbol{\theta}_{train}, x) P(\boldsymbol{\theta}_{train} \mid D_{train}) P(\boldsymbol{\theta}_{test} \mid \boldsymbol{\theta}_{train}, D_{test}) \\ &\approx P(r \mid \boldsymbol{\theta}^*_{test}, \boldsymbol{\theta}^*_{train}, x) P(\boldsymbol{\theta}^*_{train} \mid D_{train}) P(\boldsymbol{\theta}^*_{test} \mid \boldsymbol{\theta}^*_{train}, D_{test}) \end{aligned} \tag{22}$$

where $\boldsymbol{\theta}^*_{train}$ and $\boldsymbol{\theta}^*_{test}$ stand for the optimal parameters that maximizes likelihood $P(\boldsymbol{\theta}^*_{train} \mid D_{train})$ and $P(\boldsymbol{\theta}^*_{train} \mid \boldsymbol{\theta}^*_{test}, D_{test})$, respectively. In the above expression, we approximate the average with its optimal value. The advantage of the above approach is that, to learn $\boldsymbol{\theta}^*_{test}$, i.e., the parameters related to the test user, we no longer need the training data D_{train}. Instead, information inside the training data has been summarized into $\boldsymbol{\theta}^*_{train}$, i.e., parameters related to training users. Thus, $\boldsymbol{\theta}^*_{test}$ is decided only by $\boldsymbol{\theta}^*_{train}$ and D_{test}. Using this approximation, we will be able to efficiently predict ratings for the test user. Take the JMM model as an example, the parameter space related to the test user is $\boldsymbol{\theta}_{test} = \{P(z_y \mid y^t)\}$ and the optimal $P(z_y \mid y^t)$ is computed by simply maximizing the likelihood of rating data by the test user, i.e.,

$$\begin{aligned} L = \log \left\{ \sum_{z_y} P(z_y \mid y^t; \boldsymbol{\theta}_{test}) \prod_{x \in X(y^t)} \sum_{z_x} P(z_x; \boldsymbol{\theta}_{train}) P(x \mid z_x; \boldsymbol{\theta}_{train}) P(R_{y^t}(x) \mid z_x, z_y; \boldsymbol{\theta}_{train}) \right\} \\ + c \sum_{z_y} \log P(z_y \mid y^t; \boldsymbol{\theta}_{test}) \end{aligned} \tag{23}$$

In the above equation, we add either $\boldsymbol{\theta}_{train}$ or $\boldsymbol{\theta}_{test}$ into each probability to illustrate which parameter space it belongs to. Since Eq. (23) only involves the rated examples from test user y^t, finding optimal solution for $P(z_y \mid y^t)$ usually can be done efficiently.

6. Experiments

In previous sections, we have analyzed a number of mixture models with different complexity in terms of their analytical properties. In this section, we present experiment results that allow us to examine how their analytical difference is correlated with their empirical performance. Specifically, we address the following five issues:

1. *Is separate modeling of users and items important to collaborative filtering*? Recall that the JMM and the FMM differ from the Aspect Model and Bayesian Clustering in that they introduce two different class variables for modeling users and items separately. Thus, by comparing both the JMM and the FMM to the Aspect Model and the Bayesian Clustering, we will be able to see if separate clustering of users and items is effective for collaborative filtering.
2. *Is it beneficial to allow a user/item to belong to multiple classes*? The difference between the JMM and the FMM is that the JMM assumes a single class for each user while the FMM allows each user to be in multiple classes. By comparing these two models, we will be able to see which assumption is more appropriate for collaborative filtering.
3. *Which smoothing technique is more effective for collaborative filtering*? At the end of Section 4, we discussed two different methods for smoothing the EM algorithm, including an Annealed EM algorithm (AEM) and a MAP approach. Both methods prevent the

Table 4 Characteristics of movie rating and each movie

	Movie rating	Each movie
Number of Users	500	2000
Number of Items	1000	1682
Avg. Number of rated Items/User	87.7	129.6
The scale of Ratings	1–5	1–6

estimation of parameters from being skewed at the early stage of EM iterations. We will compare the effectiveness of the two smoothing methods for collaborative filtering.

4. *Would modeling the distinction between intrinsic preferences and rating patterns help improve the performance of collaborative filtering?* The Decoupled Model (DM) is similar to the Flexible Mixture Model (FMM) except that it models the intrinsic preferences and rating strategies of users separately by using two different sets of class variables. We will compare the Decoupled Model to the Flexible Mixture Model to see whether the distinction between intrinsic preferences and rating patterns helps improve the performance of collaborative filtering.
5. *How effective are the proposed models compared to other proposed models*? We compare all five mixture models to other approaches for collaborative filtering under various conditions. In previous studies, when compared with the memory-based approaches, the model-based approaches tend to have mixed results (Breese et al., 1998). It is thus interesting to see if some sophisticated models, such as the Decoupled Model that decouples the intrinsic preferences of users from their rating patterns, can outperform memory-based approaches.

Two datasets of movie ratings are used in our experiments, i.e., 'MovieRating'[1] and 'EachMovie'[2]. Specifically, we extracted a subset of 2,000 users with more than 40 ratings from 'EachMovie' since evaluation based on users with few ratings can be unreliable. The global statistics of these two datasets are summarized in Table 4.

A major challenge in collaborative filtering applications is for the system to operate effectively when it has not yet acquired a large amount of training data (i.e., the so-called "cold start" problem). To test our algorithms in such a challenging and realistic scenario, we vary the number of training users from a small value to a large value. To get a better sense of the data sparseness problem, we introduce the measurement called 'movie coverage', which measures the average number of times that each movie is rated in the training database. In particular, we consider three different cases of training data:

1. *Small Training Data*. In this case, the training database consists of 20 different users, and the 'movie coverage' for 'MovieRating' and 'EachMovie' is only 1.8 and 1.5, respectively. In another word, by average each movie is rated by less than 2 training users.
2. *Medium Training Data*. In this case, the training database consists of 100 different users for 'MovieRating' and 200 users for 'EachMovie'. Its 'movie coverage' is 8.8 and 15.4 'MovieRating' and 'EachMovie', which is substantially larger than the small training data.
3. *Large Training Data*. In this case, the training database consists of 200 different users for 'MovieRating' and 400 users for 'EachMovie'. The 'movie coverage' for this case is 17.7 and 30.8.

[1] http://www.cs.usyd.edu.au/∼irena/movie_data.zip

[2] http://research.compaq.com/SRC/eachmovie

By varying the number of training users from a 'small training data' to a 'large training data', we are able to examine the robustness of the *learning procedure*. The other dimension to examine is the robustness of mixture models with respect to the number of items rated by the test user. In this experiment, we test mixture models against test users with 5, 10, and 20 given items. By varying the number of given items, we can test the robustness of the *prediction procedure*.

For the aspect model (AM), we choose the variant in Fig. 2(a) as a baseline algorithm since it consists of a smaller number of parameters and appears to be more robust than the variant in Fig. 2(b). The number of clusters is set to be 10 for Bayesian Clustering and 20 for Aspect Model. The number of classes for users and items are set to be 10 and 20 respectively, for the Joint Mixture Model, the Flexible Mixture Model, and the Decoupled Model. These numbers are selected based on the results of cross validation. The number of classes for rating patterns in Decoupled Model is the same as the number of different rating category, which leads to 5 classes of rating patterns for 'MovieRating' and 6 for 'EachMovie'.

For evaluation, we look at the mean absolute deviation (Pennock et al., 2000) of the predicted ratings from the actual ratings on items by the test user, i.e.,

$$S = \frac{1}{m} \sum_{y^t} \sum_{x \in \bar{X}(y^t)} | R_{y^t}(x) - \hat{R}_{y^t}(x) | \tag{24}$$

where $\hat{R}_{y^t}(x)$ is the predicted rating on item x for test user y^t, $R_{y^t}(x)$ is the actual rating for test user y^t, and m is the total number of test items that have been rated by all test users. We refer to this measure as the mean absolute error (MAE) in the rest of this paper. There are some other measures like the Receiver Operating Characteristic (ROC) as a decision-support accuracy measure (Breese et al., 1998) and the normalized MAE. But since MAE has been the most commonly used metric and has been reported in most previous research (Breese et al., 1998; Herlocker et al., 1999; Melville et al., 2002; SWAMI, 2000; Pennock et al., 2000), we chose it as the evaluation measure in our experiments to make our results more comparable.

6.1. Experiments with clustering of users and items

In these experiments, we want to address the first two questions listed at the beginning of this section, namely whether modeling users and items separately is important to collaborative filtering and whether it is beneficial to allow a user/item to belong to multiple clusters. MAE results for the Joint Mixture Model, the Flexible Mixture Model, the Bayesian Clustering, and the Aspect Model for 'MovieRating' and 'EachMovie' are summarized in Tables 5 and 6 respectively.

Several interesting observations can be made from Tables 5 and 6:

1. Compared to the Joint Mixture Model (JMM), the Flexible Mixture Model (FMM) performs substantially better in most configurations except for the collection 'MovieRating' when the number of training users is only 20. This is because the FMM has more parameters to fit than the JMM and thus it fails to perform well when the number of training users is small. In the next experiment where smoothing methods are applied to the EM algorithm, we will see that the FMM is able to outperform the JMM substantially even for this single case. The only difference between these two models is that the FMM allows multiple classes for each user while the JMM does not. Thus, the fact that the FMM

Table 5 MAE results for 'MovieRating'

Training users size	Algorithms	5 items given	10 items given	20 items given
20	FMM	1.000	0.994	0.990
	JMM	0.990	**0.968**	**0.920**
	BC	1.10	1.09	1.08
	AM	**0.982**	0.976	0.958
100	FMM	**0.823**	**0.822**	**0.817**
	JMM	0.868	0.868	0.854
	BC	0.968	0.946	0.941
	AM	0.882	0.856	0.836
200	FMM	**0.804**	**0.801**	**0.799**
	JMM	0.840	0.837	0.831
	BC	0.949	0.942	0.912
	AM	0.891	0.850	0.818

'FMM' stands for the Flexible Mixture Model, 'JMM' stands for the Joint Mixture Model, 'BC' stands for Bayesian Clustering, and 'AM' stands for Aspect Model. A smaller value means a better performance.

outperforms the JMM indicates that allowing a user to be in multiple classes is important to collaborative filtering. The hypothesis is further confirmed by the fact that the aspect model performs better than the Bayesian Clustering algorithm for most configurations (except for the EachMovie dataset when the number of training users is 400).

2. Compared to the Bayesian Clustering and the Aspect Model, the Flexible Model and the Joint Mixture Model perform substantially better for most configurations except when the number of training users is small. Again, this is because both the FMM and the JMM are more sophisticated than the Bayesian Clustering and the Aspect Model and thus tend to overfit training data when the number of users is small. In the next experiment, we will see that with appropriate smoothing technique,both the FMM and the JMM perform well

Table 6 MAE results for 'EachMovie'

Training users size	Algorithms	5 items given	10 items given	20 items given
20	FMM	1.31	1.31	1.30
	JMM	1.38	1.37	1.36
	BC	1.46	1.45	1.44
	AM	**1.28**	**1.24**	**1.23**
200	FMM	**1.08**	**1.06**	**1.05**
	JMM	1.17	1.15	1.15
	BC	1.25	1.22	1.17
	AM	1.27	1.18	1.14
400	FMM	**1.06**	**1.05**	**1.04**
	JMM	1.10	1.09	1.09
	BC	1.17	1.15	1.14
	AM	1.28	1.19	1.16

'FMM' stands for the Flexible Mixture Model, 'JMM' stands for the Joint Mixture Model, 'BC' stands for Bayesian Clustering, and 'AM' stands for Aspect Model. A smaller value means a better performance.

Table 7 MAE for the Flexible Mixture Model (FMM) on the 'MovieRating' dataset using annealed EM algorithm (AEM) and maximum a posterior (MAP)

Training users size	Algorithms	5 items given	10 items given	20 items given
20	AEM	1.000	0.994	0.990
	MAP	**0.881**	**0.877**	**0.870**
100	AEM	0.823	0.822	0.817
	MAP	**0.821**	**0.820**	**0.813**
200	AEM	0.804	0.801	0.799
	MAP	**0.797**	**0.786**	**0.781**

even in the case of small training. Since both the FMM and JMM distinguish from the Aspect Model and the Bayesian Clustering in that separate clustering is applied to users and items, the results from Tables 5 and 6 indicate that modeling users and items separately is effective for collaborative filtering.

6.2. Experiments with smoothing methods

In Section 3, we discussed two different methods for smoothing the EM algorithms: the Annealed EM algorithm that avoids undesirable local optimum by slowly increasing variable 'b', and the maximum a posterior (MAP) approach that uses Dirichlet priors to regularize the mixture models. In our experiments, variable 'b' in the Annealing EM algorithm is increased from 0 to 1 at the pace of 0.1. The hyper parameters 'a', 'b', 'c', and 'd' in the MAP approach are set as follows:

$$a = \frac{\sum_y |X(y)|}{\gamma |Z_x|} \quad b = \frac{\sum_y |X(y)|}{\gamma \times M \times |Z_x|} \quad c = \frac{\sum_y |X(y)|}{\gamma \times N \times |Z_y|} \quad d = \frac{\sum_y |X(y)|}{\gamma \times R \times |Z_x| \times |Z_y|}$$

where $|X(y)|$ stands for the number of items rated by the user 'y'. Parameter γ (is determined by the cross validation approach. We randomly select 80% of training users as the training set and 20% of them as validation set. γ is ranged from 100 to 100000. The final value for γ used in our experiment is 10000.

Tables 7 and 8 summarize the results for the Flexible Mixture Model using two different smoothing methods. The results of applying smoothing methods to the Joint Mixture Model are presented in Tables 9 and 10.

Table 8 MAE for thve Flexible Mixture Model (FMM) on the 'EachMovie' dataset using Annealed EM algorithm (AEM) and maximum a posterior (MAP)

Training users size	Algorithms	5 items given	10 items given	20 items given
20	AEM	1.31	1.31	1.30
	MAP	**1.23**	**1.22**	**1.22**
200	AEM	**1.08**	1.06	1.05
	MAP	**1.08**	**1.05**	**1.04**
400	AEM	**1.06**	1.05	1.04
	MAP	**1.06**	**1.04**	**1.03**

Table 9 MAE for the Joint Mixture Model (JMM) on the 'MovieRating' dataset using Annealed EM algorithm (AEM) and maximum a posterior (MAP)

Training users size	Algorithms	5 items given	10 items given	20 items given
20	AEM	0.990	0.968	**0.920**
	MAP	**0.986**	**0.963**	**0.920**
100	AEM	0.868	0.868	**0.854**
	MAP	**0.864**	**0.863**	**0.854**
200	AEM	0.840	0.837	**0.831**
	MAP	**0.837**	**0.833**	**0.831**

Two observations can be drawn from Tables 7–10:

1. According to Tables 7–10, the MAP (i.e., maximum a posterior) approach outperforms (or as effective as) the Annealed EM algorithm for both the Joint Mixture Model and the Flexible Mixture Model in all configurations. In fact, compared to the results that do not use any smoothing algorithm in Tables 3 and 4, the Annealed EM algorithm only achieves the same performance as the original EM for all cases. Thus, our studies indicate that the MAP approach is a more effective smoothing method for collaborative filtering.
2. With a more careful examination of Tables 7 and 8, we see that the MAP approach is able to improve the performance of the FMM substantially when the number of training users is small (i.e., 20 for both 'MovieRating' and 'EachMovie'). The improvement becomes modest when the number of training user becomes large (i.e., 100 and 200 for 'MovieRating', and 200 and 400 for 'EachMovie'). This is consistent with the spirit of Bayesian statistics, in which a model prior is useful only when the amount of training data is small. When the amount of training data is sufficiently large, the effect of model prior will eventually diminish.
3. In the previous experiment, the aspect model is the winner in the case of small training data. With the help of appropriate smoothing, the FMM model is able to perform better than the aspect model in the case of small training data. This fact again indicates that the smoothing method is able to effectively alleviate the problem of sparse data.

Due to the success of the MAP method, it is used for the remaining experiments.

6.3. Experiments with the decoupled model (DM)

Compared to the other four models, the Decoupled Model is unique in that it explicitly addresses the distinction between preferences and ratings of users by modeling them separately.

Table 10 MAE for the Joint Mixture Model (JMM) on the 'EachMovie' dataset using Annealed EM algorithm (AEM) and maximum a posterior (MAP)

Training users size	Algorithms	5 items given	10 items given	20 items given
20	AEM	1.38	1.37	1.36
	MAP	**1.37**	**1.35**	**1.34**
200	AEM	**1.17**	**1.15**	1.15
	MAP	**1.17**	**1.15**	**1.14**
400	AEM	**1.10**	1.10	**1.09**
	MAP	**1.10**	**1.09**	**1.09**

Table 11 MAE for the Flexible Mixture Model (FMM) and the Decoupled Model (DM) on the 'MovieRating' dataset. A smaller value means a better performance

Training users size	Algorithms	5 items given	10 items given	20 items given
20	DM	**0.874**	**0.871**	**0.860**
	FMM	0.881	0.877	0.870
100	DM	**0.814**	**0.810**	**0.799**
	FMM	0.821	0.820	0.813
200	DM	**0.790**	**0.777**	**0.761**
	FMM	0.797	0.786	0.781

In this experiment we attempt to answer the question, i.e., *would modeling the distinction between the preferences and ratings help improve the performance*? The results for the Decoupled Model on 'MovieRating' and 'EachMovie' are listed in Tables 11 and 12 together with the results for the Flexible Mixture Model (copied from Tables 4 and 5) The Flexible Mixture Model is closely related to the Decoupled Model and differs from it only by the lack of modeling for rating patterns. By comparing the performance of these two models, we will be able to see if the introduction of separate class variables for preferences and ratings is effective for collaborative filtering.

According to Tables 11 and 12 the Decoupled Model outperforms the Flexible Mixture Model in all configurations. Although the difference in performance appears to be insignificant in some cases, it is interesting to note that when the number of given items increases, the gap between these two models also increases. One possible explanation is that when there are only a small number of given items, it is rather difficult to determine the type of rating patterns for the testing user. As the number of given items increases, this ambiguity will decrease quickly and therefore the advantage of the Decoupled Model over the Flexible Mixture Model becomes clearer. Indeed, it is a bit surprising that even with only five given items and a small number of training users, the Decoupled Model still improves the performance slightly as it has many more parameters to estimate than the Flexible Mixture Model. We suspect that the skewed distribution of ratings among items, i.e., a few items account for a large number of ratings, may have helped.

6.4. Comparison with other approaches for collaborative filtering

In this subsection, we compare all five mixture models to the memory-based approaches for collaborative filtering, including the Personal Diagnosis (PD), the Vector Similarity method

Table 12 MAE for the Flexible Mixture Model (FMM) and the Decoupled Model (DM) on the 'EachMovie' dataset. A smaller value means a better performance

Training users size	Algorithms	5 items given	10 items given	20 items given
20	DM	**1.20**	**1.18**	**1.17**
	FMM	1.23	1.22	1.22
200	DM	**1.07**	**1.04**	**1.03**
	FMM	1.08	1.05	1.04
400	DM	**1.05**	**1.03**	**1.02**
	FMM	1.06	1.04	1.03

(VS) and the Pearson Correlation Coefficient method (PCC). We first briefly introduce the three memory-based approaches and then present the empirical results.

6.4.1. Memory-based methods for collaborative filtering

Memory-based algorithms store the rating examples of training users and predict a test user's ratings based on the corresponding ratings of the users in the training database that are similar to the test user. Three commonly used methods will be compared in this experiment. They are:

- **Pearson Correlation Coefficient (PCC)**
  According to (Resnick et al., 1994), the Pearson Correlation Coefficient method predicts the rating of a test user y^t on item x as:
  $$\hat{R}_{y^t}(x) = \bar{R}_{y^t} + \frac{\sum_{y \in Y} w_{y^t,y}(R_y(x) - \bar{R}_y)}{\sum_{y \in Y} w_{y^t,y}}$$
  where the coefficient w_{y,y^t} is computed as
  $$w_{y^t,y} = \frac{\sum_{x \in X(y) \wedge \tilde{X}(y^t)} (R_y(x) - \bar{R}_y)(R_{y^t}(x) - \bar{R}_{y^t})}{\sqrt{\sum_{x \in X(y) \wedge \tilde{X}(y^t)} (R_y(x) - \bar{R}_y)^2} \sqrt{\sum_{x \in X(y) \wedge \tilde{X}(y^t)} (R_{y^t}(x) - \bar{R}_{y^t})^2}}$$

- **Vector Similarity (VS)**
  This method is very similar to the PCC method except that the correlation coefficient w_{y,y^t} is computed as:
  $$w_{y^t,y} = \frac{\sum_{x \in X(y)^{\tilde{X}}(y^t)} R_y(x) R_{y^t}(x)}{\sqrt{\sum_{x \in X(y)} R_y(x)^2} \sqrt{\sum_{x \in \tilde{X}(y^t)} R_{y^t}(x)^2}}$$

- **Personality Diagnosis (PD)**
  In the personality diagnosis model, the rating of test user y^t on item x is assumed to be drawn from an independent normal distribution with the mean as the true rating as $R_{y^t}^{True}(x)$:
  $$P\left(R_{y^t}(x) \middle| R_{y^t}^{True}(x)\right) \propto e^{-(R_{y^t}(x) - R_{y^t}^{True}(x))^2/2\sigma^2}$$
  where the standard deviation σ is set to constant 1 in our experiments. Then, the probability of generating the observed rating values of the test user by any training user y is written as:
  $$P(R_{y^t}|R_y) \propto \prod_{x \in X(y^t)} e^{-(R_y(x) - R_{y^t}(x))^2/2\sigma^2}$$
  Finally, the likelihood for test user y^t to rate an unseen item x as r is computed as:
  $$P(R_{y^t}(x) = r) \propto \sum_y P(R_{y^t}|R_y) e^{-(R_y(x) - r)^2/2\sigma^2}$$

The predicted rating for item 'x' by the test user will be the rating category r that has the largest likelihood $P(R_{y^t}(x) = r)$. Previous empirical studies have shown that the PD method performs better than several other approaches for collaborative filtering (Pennock et al., 2000).

6.4.2. Comparison results

The results for five mixture models and three memory-based approaches are summarized in Tables 13 and 14 Both the Decoupled Model and the Flexible Mixture Model are considerably

Table 13 MAE for eight different models on the 'MovieRating' dataset, including a Pearson Correlation Coefficient approach (PCC), a Vector Similarity approach (VS), a Personality Diagnosis approach (PD), a Aspect Model (AM), a Bayesian Clustering approach (BC), a Decoupled Model (DM), a Flexible Mixture Model (FMM) and a Joint Mixture Model (JMM). A smaller value means a better performance

Training users size	Algorithms	5 items given	10 items given	20 items given
20	PCC	0.912	**0.840**	**0.812**
	VS	0.912	**0.840**	**0.812**
	PD	0.888	0.882	0.875
	AM	0.982	0.976	0.958
	BC	1.10	1.09	1.08
	DM	**0.874**	0.871	0.860
	FMM	0.881	0.877	0.870
	JMM	0.986	0.963	0.920
100	PCC	0.881	0.832	0.809
	VS	0.859	0.834	0.823
	PD	0.839	0.826	0.818
	AM	0.882	0.856	0.836
	BC	0.968	0.946	0.941
	DM	**0.814**	**0.810**	**0.799**
	FMM	0.821	0.820	0.813
	JMM	0.864	0.863	0.854
200	PCC	0.878	0.828	0.801
	VS	0.862	0.950	0.854
	PD	0.835	0.816	0.806
	AM	0.891	0.850	0.818
	BC	0.949	0.942	0.912
	DM	**0.790**	**0.777**	**0.761**
	FMM	0.797	0.786	0.781
	JMM	0.837	0.833	0.831

better in most configurations than the other methods for collaborative filtering, including the three mixture models and three model-based approaches for most cases. The only exception is when the number of training user is 20, in which the memory-based models perform substantially better than the model-based approaches. The overall success of the Decoupled Model and the Flexible Mixture Model suggests that, compared to the memory-based approaches, graphical models are not only advantageous in principle, but also empirically superior due to their capabilities of capturing the distinction between the intrinsic preferences and rating patterns in a principled way.

The fact that memory-based approaches perform better in the case of small training data is because the number of parameters used by the model-based approaches is larger than the size of training data. When there are only 20 training users, the number of rated items is less than 3,000 (1700 for the 'MovieRating' dataset and 2500 for 'EachMovie' dataset), but the number of parameters is actually over 20,000 for all the models (over 20,000 for 'MovieRating' dataset and 30,000 for 'EachMovie' dataset.). Therefore, when there are only 20 training users, the amount of training data is insufficient for creating a reliable and effective model for collaborative filtering.

Table 14 MAE for eight different models on the 'EachMovie' dataset, including a Pearson Correlation Coefficient approach (PCC), a Vector Similarity approach (VS), a Personality Diagnosis approach (PD), a Aspect Model (AM), a Bayesian Clustering approach (BC), a Decoupled Model (DM), a Flexible Mixture Model (FMM) and a Joint Mixture Model (JMM). A smaller value means a better performance

Training users size	Algorithms	5 items given	10 items given	20 items given
20	PCC	1.26	1.19	1.18
	VS	1.24	1.19	**1.17**
	PD	1.25	1.24	1.23
	AM	1.28	1.24	1.23
	BC	1.46	1.45	1.44
	DM	**1.20**	**1.18**	**1.17**
	FMM	1.23	1.22	1.22
	JMM	1.37	.135	1.34
200	PCC	1.22	1.16	1.13
	VS	1.25	1.24	1.26
	PD	1.19	1.16	1.15
	AM	1.27	1.18	1.14
	BC	1.25	1.22	1.17
	DM	**1.07**	**1.04**	**1.03**
	FMM	1.08	1.05	1.04
	JMM	1.17	1.15	1.14
400	PCC	1.22	1.16	1.13
	VS	1.32	1.33	1.37
	PD	1.18	1.16	1.15
	AM	1.28	1.19	1.16
	BC	1.17	1.15	1.14
	DM	**1.05**	**1.03**	**1.02**
	FMM	1.06	1.04	1.03
	JMM	1.10	1.09	1.09

This analysis indicates that the performance of model-based approaches usually depends strongly on the availability of training data. When the amount of training data is small, it is better to use memory-based approaches for collaborative filtering.

7. Conclusions and future work

In this paper, we conduct a systematic study of a large subset of graphicals models – mixture models – for collaborative filtering. In general, there are three components that need to be modeled carefully: the users, the items and the ratings. We proposed three desirable properties that a reasonable graphical model for collaborative filtering should satisfy: (1) separate clustering of users and items; (2) flexibility for a user/item to be in multiple clusters; (3) decoupling of users' preferences and rating patterns.

We thoroughly analyzed five different mixture models, including the Bayesian Clustering (BC), the Aspect Model (AM), the Flexible Mixture Model (FMM), the Joint Mixture Model (JMM) and the Decoupled Model (DM) based on the three proposed properties, and found that (1) The DM is the only model that satisfies all the three properties, and all others fail to decouple user preferences and rating patterns; (2) The JMM and FMM models allow separate

clustering of users and items, whereas the BC and AM do not; and (3) Compared with JMM, the FMM further allows a user to be in multiple clusters.

We study the empirical impact of such analytical difference on real datasets. Experiments over two datasets of movie ratings under several different configurations show that in general, the fulfillment of the proposed properties tends to be positively correlated with the model's performance. In particular, the Decoupled Model, which satisfies all three properties, outperforms the other mixture models as well as most memory-based approaches for collaborative filtering. Experiments also show that the Flexible Mixture Model is consistently better than the Joint Mixture Model by the MAE measure, which indicates that it is beneficial to allow a user to be in multiple classes. Meanwhile, the success of the FMM over the Bayesian Clustering algorithm and the Aspect Model indicates that it is important to have separate clustering of users and items for collaborative filtering.

We also empirically study two smoothing methods, the Annealed EM algorithm (AEM) and the Maximum A Posterior (MAP), and found that smoothing is important for improving the performance of collaborative filtering systems, particularly when the number of training users is small. Empirical results show that the MAP is a more effective method for collaborative filtering.

In summary, our study shows that graphical models are powerful tools for modeling collaborative filtering, but careful design of the model is necessary to achieve good performance.

There are several interesting directions for extending this work. First, given the success of decoupling user preferences from rating patterns, it would be very interesting to explore other ways of modeling preferences as done in some related work (Ha & Haddawy, 1998; Freund et al., 1998;Cohen et al., 1999). One potentially promising direction is to treat the rating problem as a ranking problem, and apply the existing ranking algorithms, such as Prank and RankBoost, to collaborative filtering. In the future, we plan to study how to incorporate the ranking algorithm into the graphical models. Second, we also believe that the decoupling problem that we addressed may represent a more general need of modeling "noise" in similar problems such as gene microarray data analysis in bioinformatics. We plan to explore a more general probabilistic framework for all these similar problems.

Apendix A: The EM algorithm for the joint mixture model using maximum a posterior (MAP) approach

We studied the MAP approach for mixture models in Section 4.2. The idea is to introduce the model priors that express the preference of parameters given no training data. The resulting parameters will not only maximize the likelihood of training data but also satisfy the prior preference. The E-step for the Joint Mixture Model using MAP approach is same as the original one that is already stated in Eqs. (12) and (13). The updating equations in M-step are changed to the following expressions:

$$P(z_y \mid \{x, R_y(x)\}_{x\in X(y)}, y) = \frac{c + P(z_y \mid y)\prod_{x\in X(y)} P(x, R_y(x)|z_y)}{\sum_{z_y}\left\{c + P(z_y \mid y)\prod_{x\in X(y)} P(x, R_y(x)|z_y)\right\}} \tag{14′}$$

$$P(z_x) = \frac{a + \sum_{z_y} \sum_y \sum_{x \in X(y)} P(z_x \mid x, R_y(x), y, z_y) P(z_y \mid y)}{\sum_{z_x} \{a + \sum_{z_y} \sum_y \sum_{x \in X(y)} P(z_x \mid x, R_y(x), y, z_y) P(z_y \mid y)\}} \quad (15')$$

$$P(x \mid z_x) = \frac{b + \sum_{z_y} \sum_y \sum_{x' \in X(y)} P(z_x \mid x', R_y(x'), y, z_y) P(z_y \mid y) \delta(x = x')}{Mb + \sum_{z_y} \sum_y \sum_{x' \in X(y)} P(z_x \mid x', R_y(x'), y, z_y) P(z_y \mid y)} \quad (16')$$

$$P(r \mid z_x, z_y) = \frac{d + \sum_y \sum_{x \in X(y)} P(z_x \mid x, R_y(x), y, z_y) P(z_y \mid y) \delta(R_y(x) = r)}{Rd + \sum_y \sum_{x \in X(y)} P(z_x \mid x, R_y(x), y, z_y) P(z_y \mid y)} \quad (17')$$

Compared to the EM algorithm for the JMM in Eqs. (14)–(17), hyper parameters a, b, c, and d in the above equations behave like pseudo counts. In addition to the 'counts' that are collected from training data, all probabilities are also affected by the pseudo counts that come from hyper parameters. When the number of training examples is small, the pseudo counts will dominate over the estimation and thus the distribution tends to be uniform. On the other hand, when the amount of training data is large, the effect of pseudo counts will be ignored and the results obtained from the maximum a posterior approach will be similar to the maximum likelihood approach.

References

Breese JS, Heckerman D and Kadie C (1998) Empirical analysis of predictive algorithms for collaborative filtering. In the Proceeding of the Fourteenth Conference on Uncertainty in Artificial Intelligence

Cohen W, Shapire R and Singer Y (1998) Learning to order things. In: Advances in Neural Processing Systems 10. MIT Press, Denver, CO, 1997

Connor M and Herlocker J (2001) Clustering items for collaborative filtering. In the Proceedings of SIGIR-2001 Workshop on Recommender Systems, New Orleans, LA

Dempster AP, Laird NM and Rubin DB (1977) Maximum likelihood from incomplete data via the EM algorithm. Journal of the Royal Statistical Society, B39:1–38

Fisher D, Hildrum K, Hong J, Newman M and Vuduc R (2000) SWAMI: A framework for collaborative filtering algorithm development and evaluation. In the Proceedings of the 23rd Annual International Conference on Research and Development in Information Retrieval (SIGIR)

Freund Y, Iyer R, Shapire R and Singer Y (1998) An efficient boosting algorithm for combining preferences. In Proceedings of ICML 1998

Ha V and Haddawy P (1998) Toward case-based preference elicitation: Similarity measures on preference structures. In: Proceedings of UAI 1998

Herlocker JL, Konstan JA, Brochers A and Riedl J (1999) An algorithm framework for performing collaborative filtering. In: Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)

Hofmann T and Puzicha J (1999) Latent class models for collaborative filtering. In: Proceedings of International Joint Conference on Artificial Intelligence 1999

Hofmann T and Puzicha J (1998) Statistical models for co-occurrence data (Technical report). Artificial Intelligence Laboratory Memo 1625, M.I.T

Hofmann T (2003) Gaussian latent semantic models for collaborative filtering. In: Proceedings of the 26th Annual International ACM SIGIR Conference

Jin R, Si L and Zhai CX (2003) Preference-based graphical models for collaborative filtering. In: Proceedings of the Nineteenth Conference on Uncertainty in Artificial Intelligence (UAI)

Melville P, Mooney RJ and Nagarajan R (2002) Content-boosted collaborative filtering for improved recommendations. In the Proceedings of the Eighteenth National Conference on Artificial Intelligence (AAAI)

Pennock DM, Horvitz E, Lawrence S and Giles CL (2000) Collaborative filtering by personality diagnosis: A hybrid memory- and model-based approach. In the Proceeding of the Sixteenth Conference on Uncertainty in Artificial Intelligence

Popescul A Ungar LH Pennock DM and Lawrence S (2001) Probabilistic models for unified collaborative and content-based recommendation in sparse-data environments. In: Proceeding of the Seventeenth Conference on Uncertainty in Artificial Intelligence

Resnick P, Iacovou N, Suchak M, Bergstrom P and Riedl J (1994) Grouplens: An open architecture for collaborative filtering of netnews. In Proceeding of the ACM 1994 Conference on Computer Supported Cooperative Work

Ross DA and Zemel RS (2002) Multiple-cause vector quantization. In NIPS-15: Advances in Neural Information Processing Systems 15

Si L and Jin R (2003) Product space mixture model for collaborative filtering. In: Proceedings of the Twentieth International Conference on Machine Learning (ICML)

Ueda N and Nakano R (1998) Deterministic annealing EM algorithm. Neural Networks, 11(2):271–282