Flexible Mixture Model for Collaborative Filtering

Luo Si LSI@CS.CMU.EDU
Rong Jin RONG@CS.CMU.EDU
School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15232 USA

Abstract

This paper presents a flexible mixture model (FMM) for collaborative filtering. FMM extends existing partitioning/clustering algorithms for collaborative filtering by clustering both users and items together simultaneously without assuming that each user and item should only belong to a single cluster. Furthermore, with the introduction of 'preference' nodes, the proposed framework is able to explicitly model how users rate items, which can vary dramatically, even among the users with similar tastes on items. Empirical study over two datasets of movie ratings has shown that our new algorithm outperforms five other collaborative filtering algorithms substantially.

1. Introduction

The rapid growth of the information on the Internet demands intelligent information agent 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: c*ollaborative filtering* (Breese, Heckerman & Kadie, 1998) and c*ontent-based recommending* (Basu & Hirsh, 1998). The difference between collaborative filtering and content-based recommending is that: collaborative filtering only utilizes the ratings of training users in order to predict ratings for test users while content-based recommendation systems rely on the contents of items for predictions. Therefore, collaborative filtering systems have advantages in the environments where the contents of items are not available due to privacy issues or where that contents are difficult for a computer to analyze. In this paper, we only focus on the collaborative filtering problems.

Most collaborative filtering methods fall into two categories: memory-based algorithms and model-based algorithms. Memory-based algorithms usually do not have a training phase. Instead, they simply store rating examples of users into a training database. In the predicting phase, the memory-based approaches first find users in the training database similar to the test user and then, predict the test user's ratings based on the corresponding ratings of these similar users. On the contrary, model-based algorithms build models that are able to explain the training examples well and predict the ratings of test users using the estimated models. Both of the memory-based algorithms and the model-based algorithms have their advantages and disadvantages. Memory-based algorithms have much less off-line computation costs while the model-based algorithms may have less on-line computation costs.

Though memory-based and model-based approaches differ from each other in many aspects, both of them assume that users with similar tastes should rate items similarly and therefore the idea of clustering is used in both approaches either explicitly or implicitly. For memory-based approaches, training users similar to the test user are grouped together and their ratings are combined to predict ratings for the test user. Meanwhile, model-based approaches cluster items and/or training users into classes explicitly and predict ratings of a test user by simply using the ratings of classes that fit in best with the test user and/or items to be rated. Thus, how to cluster users and items appropriately is a key issue in designing collaborative filtering systems, which can affect the scalability, robustness and performance.

While theoretically interesting, model-based approaches have achieved mixed results in previous studies (Breese et al., 1998; O'Connor & Herlocker, 2001). We suspect that this may be due to the inappropriate clustering algorithms used in their studies. More specifically, three issues of clustering algorithms are important for collaborative filtering: First, both users and items need to be clustered and more importantly, users and items are coupled with each other through the rating information. Therefore, a good clustering algorithm should be able to explicitly model both classes of users and items and be able to leverage their correlation. Secondly, many clustering techniques assume that each user or item belongs to a single class. However, since a user may have diverse interests and an item may have multiple aspects, it is desirable to allow both items and users to be in multiple classes simultaneously. In this paper, a flexible mixture model (FMM) is proposed in order to capture this idea. Thirdly, the assumption that users with similar tastes would have similar ratings may not necessarily be true because some users may tend to give a higher rating to all items than some others. In order to account for the variance in the rating patterns among the users with similar interests, we extend the flexible mixture model by introducing an additional new hidden 'preference' node. Such an extension allows us to infer the preference values underlying the surface ratings, which can then be used as (presumably more reliable) evidence for clustering. For the simplicity of computation, we compute the preference information for each user using 'decoupled models' (DM), and then apply the proposed flexible mixture model to cluster over the estimated preference values instead of the original rating values.

The rest of the paper is arranged as follows: Section 2 discusses previous work. The proposed flexible mixture model for collaborative filtering is presented in Section 3. The extended version of flexible mixture model with the decoupling of rating and preference is discussed in Section 4. Section 5 presents experiments results. Conclusions and future work are discussed in Section 6.

2. Existing Approaches

Let us first introduce the annotations that will be used for the rest of this paper. Let $X = \{x_1,......,x_M\}$ be a set of items, $Y = \{y_1,......,y_N\}$ be a set of users, and $\{1,...,R\}$ be a set of ratings. Let $\{(x_{(1)},y_{(1)},r_{(1)}),.....,(x_{(L)},y_{(L)},r_{(L)})\}$ be the ratings information in the training database, $X(y)$ be the set of items rated by user *y*, $R_y(x)$ be the rating of item *x* by user *y*, and $\bar{R}_y$ be the average rating by user *y*.

2.1 Memory-Based Algorithms

Two commonly used memory-based algorithms are Pearson Correlation Coefficient algorithm (PCC) (Resnick et al., 1994) and Vector Space Similarity (VS) (Breese, Heckerman & Kadie, 1998) algorithm. The main idea of these two algorithms is to calculate the similarities of the training users to the test user and the prediction of ratings is computed by performing a weighted average of deviations from the training users' mean. The difference between them is on how to compute the similarities between users, where a Pearson correlation coefficient is used for measuring the user similarity in the PCC algorithm and a cosine similarity is computed in the VS algorithm. More details can be found in (Resnick et al., 1994; Breese, Heckerman & Kadie, 1998).

2.2 Model-Based Algorithms

Three model-based algorithms are discussed here: the aspect model (AM) (Hofmann & Puzicha, 1999), two-sided clustering model (Hofmann & Puzicha, 1999) and the Personality Diagnosis model (PD) (Pennock et al., 2000).

Aspect Model (AM)

The aspect model (Hofmann & Puzicha, 1999) is a probabilistic latent space model, which models individual preferences as a convex combination of preference factors. The latent class variable $z \in Z = \{z_1, z_2,.....,z_K\}$ is associated with each observation pair of a user and an item. The aspect model assumes that users and items are independent from each other given the latent class variable. Thus, the probability for each observed pair (x,y) is calculated as follows:

$$P(x,y) = \sum_{z \in Z} P(z)P(x \mid z)P(y \mid z) \qquad (1)$$

where P(z) stands for class prior probability, P(x|z) and P(y|z) stand for class dependent distributions for items and users respectively. Essentially, the preference pattern of a user is modeled by a combination of typical preference patterns, which are represented by the distributions of P(z), P(x|z) and P(y|z).

Note that the aspect model only introduces one set of latent variables 'Z' for the purpose of clustering and there is no explicitly grouping of either users or items. In the proposed model, we intentionally introduce two sets of latent variables in order to model the clusters of users and the clusters of items separately.

Two of the choices (Hofmann & Puzicha, 1999) to incorporate the ratings '*r*' into the aspect model are expressed in Equation (2) and (3), respectively.

$$P(x_{(l)},y_{(l)},r_{(l)}) = \sum_{z \in Z} P(z)P(x_{(l)} \mid z)P(y_{(l)} \mid z)P(r_{(l)} \mid z) \qquad (2)$$

$$P(x_{(l)},y_{(l)},r_{(l)}) = \sum_{z \in Z} P(z)P(x_{(l)} \mid z)P(y_{(l)} \mid z)P(r_{(l)} \mid z, x_{(l)}) \qquad (3)$$

The corresponding graphical models for Equation (2) and (3) are shown in Figure 1 as model (a) and (b) respectively. According to the graphic models, the difference between these two methods is that, in the graphic model (a) (or Equation. (2)), the rating $r_{(l)}$ is conditioned only on the latent class variable 'Z', while the second model let the rating $r_{(l)}$ be conditioned on both the latent class variable 'Z' and the item $x_{(l)}$. The second model is a refined version of the first model, but the number of parameters is much larger than the first model with the same number of aspects.

Two-Sided Clustering Model

A two-sided clustering model is proposed for collaborative filtering in (Hofmann & Puzicha, 1999). This model assumes that each user should belong to exactly one group of users and the same is true for each item. Let $C = \{c_1,.....,c_I\}$ be the classes of users and $D = \{d_1,.....,d_J\}$ be the classes of items. Indicator

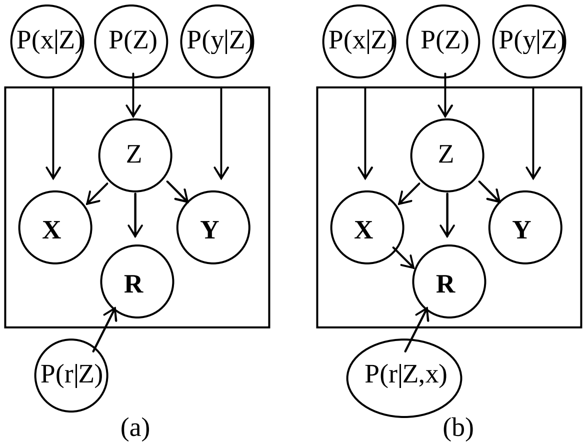

Figure 1. Graphical models for the two extensions of aspect model in order to capture rating values.

variables I_{iv} and $J_{ju} \in \{0,1\}$ indicates whether the *i*th user belongs to the *v*th user class and the *j*th item belongs to the *u*th item class respectively. Then, the joint probability P(x,y,r) is defined as:

$$P(x_{(l)}, y_{(l)}, r_{(l)}) = P(x_{(l)})P(y_{(l)})\sum_{v,u} I_{x_{(l)}v} J_{y_{(l)}u} C_{vu} \qquad (4)$$

C_{vu} is the cluster association parameter. In order to be consistent with the above assumption, a global normalization constraint has to be made as

$$\sum_{x,y} P(x)P(y)\sum_{v,u} I_{xv} J_{yu} C_{vu} = 1 \qquad (5)$$

(Hofmann & Puzicha, 1999) pointed out that this model has a different spirit from the aspect model and is less flexible in modeling the preferences relationship between the users and items, and we believe the key reason is its strong assumption. However, this model does try to model the clustering of users and items separately, which appears to be a better modeling approach. Since previous experiments have shown that the performance of aspect model is substantially better than the two-sided clustering model, we will not compare our model with the two-sided clustering model.

Personality Diagnosis Model (PD)

In the personality diagnosis model (Pennock et al., 2000), the observed rating for the test user y^t on an item x is assumed to be drawn from an independent normal distribution with the mean as the true rating as $R_{y^t}^{True}(x)$:

$$P(R_{y^t}(x) \mid R_{y^t}^{True}(x)) \propto e^{-(R_{y^t}(x) - R_{y^t}^{True}(x))^2 / 2\sigma^2} \qquad (6)$$

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 user *y* in the training database can be written as:

$$P(R_{y^t} \mid R_y) \propto \prod_{x \in X(y^t)} e^{-(R_y(x) - R_{y^t}(x))^2 / 2\sigma^2} \qquad (7)$$

The likelihood for the test user y^t to rate an unseen item *x* as category *r* can be computed as:

$$P(R_{y^t}^{True}(x) = r) \propto \sum_y P(R_{y^t} \mid R_y) e^{-(R_y(x) - r)^2 / 2\sigma^2} \qquad (8)$$

The final predicted rating for item '*x*' by the test user will be the rating category '*r*' with the highest likelihood $P(R_{y^t}^{True}(x) = r)$. Empirical studies have shown that the PD method is able to outperform several other approaches for collaborative filtering (Pennock et al., 2000).

3. The Flexible Mixture Model (FMM)

In this section, we introduce the flexible mixture model (FMM) and show how it can be applied to the collaborative filtering task.

3.1 Model Description

The FMM for collaborative filtering is motivated by the following observations on the two-sided clustering model and the aspect model. Compared with the two-side clustering model, the aspect model has the flexibility of letting each user and item belong to multiple groups simultaneously while the two-sided clustering model restricts each user and item to be in exactly one cluster. This issue can be quite important for the collaborative filtering task because there may not be a set of underlying clusters for users and items that are exclusive from each other. Most likely, we will see overlapping clusters, which lead to multiple memberships for users and items. For example, the film "Tora! Tora! Tora!" may be deemed as a 'war movie' by a young man due to its intensive war scenes while a veteran may treat it as a 'historical film' because of the historical events described in the movie. Clearly, the non-exclusive nature between the category 'historical movie' and 'war movie' leads to the multiple membership for this movie. On the other hand, the two-sided clustering model is able to explicitly model the clusters of users and items, which appears to make sense. Based on these observations, we propose the flexible mixture model (FMM) for collaborative filtering, which tries to address the two issues, namely allowing each user and item to be in multiple clusters and modeling the clusters of users and items separately.

Let $C = \{c_1, ..., c_I\}$ be the classes of users and $D = \{d_1, ..., d_J\}$ be the classes of items; Latent variable Z_y indicates the class membership for user '*y*' and $P(Z_y)$ $(1 \le Z_y \le I)$ is a multinomial distribution on the user classes; Latent variable Z_x indicates the class membership for item '*x*' and $P(Z_x)$ $(1 \le Z_x \le J)$ is a multinomial distribution on the item classes; $P(Y \mid Z_y)$

$(1 \le Y \le N,\ 1 \le Z_y \le I)$ is a multinomial distribution describing the conditional probability of users Y given a specific user class Z_y; $P(X \mid Z_x)$ $(1 \le X \le M,\ 1 \le Z_x \le J)$ is a multinomial distribution describing the conditional probability of items X given a specific item class Z_x; $P(r \mid Z_x, Z_y)$ $(1 \le r \le R,\ 1 \le Z_y \le I,\ 1 \le Z_x \le J)$ is a multinomial distribution for the ratings 'r' given a specific user class Z_y and a specific item class Z_x.

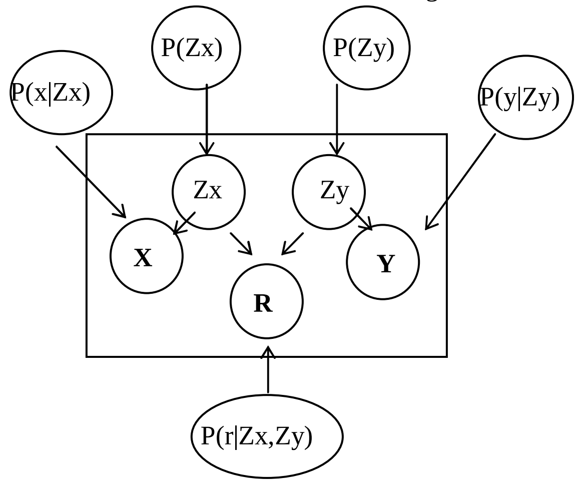

Figure 2. Graphical model representation for the flexible mixture model (FMM).

With above annotation, the joint generation probability P(x,y,r) for FMM can be written as:

$$P(x_{(l)}, y_{(l)}, r_{(l)}) = \sum_{Z_x, Z_y} P(Z_x) P(Z_y) P(x_{(l)} \mid Z_x) P(y_{(l)} \mid Z_y) P(r_{(l)} \mid Z_x, Z_y) \qquad (9)$$

The corresponding graphical model is shown in Figure 2. According to the model, the FMM differs from the aspect model in that it explicitly models the user classes and the items classes with two sets of latent variables $\{Z_y\}$ and $\{Z_x\}$. The FMM model is different from the two-sided clustering model by the fact that it does not have the global normalization restriction in Equation (5).

The graphical model most similar to the proposed FMM model is the product space mixture model (PSMM) **(**Hofmann & Puzicha, 1998), which was proposed for information retrieval. But the PSMM model only extends the aspect model by enforcing a decomposition of aspects that sum up to 1, while our FMM has two sets of latent variables 'Zx' and 'Zy', which normalized separately.

3.2 The Training Procedure

The Expectation and Maximization (EM) (Dempster & Rubin, 1977) algorithm is a well-known optimization algorithm, which alternates between two steps: In the expectation step, the joint posterior probabilities of the latent variables {*Zx*, *Zy*} are computed; in the maximization step, the model parameters are updated given the posterior probabilities estimated in the expectation step. More specifically, in the expectation step, the joint posterior probabilities are computed as:

$$P(z_x, z_y \mid x_{(l)}, y_{(l)}, r_{(l)}) = \frac{P(Z_x) P(Z_y) P(x_{(l)} \mid Z_x) P(y_{(l)} \mid Z_y) P(r_{(l)} \mid Z_x, Z_y)}{\sum_{Z_x, Z_y} P(Z_x) P(Z_y) P(x_{(l)} \mid Z_x) P(y_{(l)} \mid Z_y) P(r_{(l)} \mid Z_x, Z_y)} \qquad (10)$$

Then, the model parameters are updated in the maximization step as:

$$P(z_x) = \frac{\sum_l \sum_{z_y} P(z_x, z_y \mid x_{(l)}, y_{(l)}, r_{(l)})}{L} \qquad (11)$$

$$P(z_y) = \frac{\sum_l \sum_{z_x} P(z_x, z_y \mid x_{(l)}, y_{(l)}, r_{(l)})}{L} \qquad (12)$$

$$P(x \mid z_x) = \frac{\sum_{l: x_{(l)} = x} \sum_{z_y} P(z_x, z_y \mid x_{(l)}, y_{(l)}, r_{(l)})}{L \times P(z_x)} \qquad (13)$$

$$P(y \mid z_y) = \frac{\sum_{l: y_{(l)} = y} \sum_{z_x} P(z_x, z_y \mid x_{(l)}, y_{(l)}, r_{(l)})}{L \times P(z_y)} \qquad (14)$$

$$P(r \mid z_x, z_y) = \frac{\sum_{l: r_{(l)} = r} P(z_x, z_y \mid x_{(l)}, y_{(l)}, r_{(l)})}{\sum_l P(z_x, z_y \mid x_{(l)}, y_{(l)}, r_{(l)})} \qquad (15)$$

In order to avoid the unfavorable local maximum problems, we use a general form of the EM algorithm named annealed EM algorithm (AEM) **(**Hofmann & Puzicha, 1998), which is an EM algorithm with regularization. In this algorithm, the training database is divided into two parts: the training data and the held-out data. In the expectation step, a variable 'b' is introduced to control the training process as:

$$P(z_x, z_y \mid x_{(l)}, y_{(l)}, r_{(l)}) = \frac{(P(Z_x) P(Z_y) P(x_{(l)} \mid Z_x) P(y_{(l)} \mid Z_y) P(r_{(l)} \mid Z_x, Z_y))^b}{\sum_{Z_x, Z_y} (P(Z_x) P(Z_y) P(x_{(l)} \mid Z_x) P(y_{(l)} \mid Z_y) P(r_{(l)} \mid Z_x, Z_y))^b} \qquad (16)$$

When variable 'b' goes to positive infinity, the posterior probability becomes a delta function and the clustering process becomes the hard case (each user and item belong to a single class). When variable 'b' is set to 1, the AEM returns back to the original EM algorithm in Equation (10). Therefore, by varying the variable 'b', we can adjust the clustering process. In our procedure, b is initially set to 1. We perform EM algorithm with early stopping if the performance on hold-out data deteriorates. Then the variable b is decreased (b=0.9*b) and the EM is applied again until b is smaller than a lower bound (0.5). Finally a new model is trained over the whole training data (including the held-out dataset) with the current b value for several steps. A similar training procedure is applied to train the aspect models as described in Section 2.2.

3.3 The Prediction Procedure

The ultimate goal of the collaborative filtering is to predict ratings for the test user y^t on unseen items given a set of observed ratings of the test user y^t: $X(y^t)$ =

$\{(x^t_{(1)}, y^t, r^t_{(1)}),..,(x^t_{(N\_G)}, y^t, r^t_{(N\_G)})\}$ where N_G stands for the number of given ratings of the test user y^t.

A "fold-in" process can be used to make the prediction. The main idea of this process is to estimate the joint probability of the rating, item and the test user as $P(x, y^t, r)$ and to predict the rating with the expectation as $\hat{R}_{y^t}(x)$. The joint probability is calculated as:

$$P(x, y^t, r) = \sum_{Z_x, Z_y} P(Z_x)P(Z_y)P(x \mid Z_x)P(y^t \mid Z_y)P(r \mid Z_x, Z_y) \quad (17)$$

We have all the variables on the right hand side of Equation (17) from the training process except $P(y^t \mid Z_y)$, which can be computed by simply treating the test user y^t as another user in the training database and run the EM algorithm as described above with all the parameters fixed except $P(y^t \mid Z_y)$.

With the estimated joint probability $P(x, y^t, r)$, the prediction of rating on item '*x*' can be computed as:

$$\hat{R}_{y^t}(x) = \sum_r r \frac{P(x, y^t, r)}{\sum_{r'} P(x, y^t, r')} \quad (18)$$

4. The Combination of FMM and the Decoupled Models of Preferences and Ratings

The other key issue with the collaborative filtering is that users with similar or even identical taste (preference) on the items may give very different surface ratings. For example, two movie viewers A and B may have exactly the same taste on the films, which means they both like the same set of films and disfavor another same set of films. But the viewer A can be quite strict on his rating standard and may rate most of his favorite films only as rating '3' and rate all of his disfavored films with rating '1'. On the hand, the viewer B is a quite tolerating person and may rate most of his favorite films with highest rating '5' and even rate his disfavored films with rating '3'.

In order to account for the variance in the rating behavior among the users with similar interests, we extend the graphical model in Figure 2 by introducing a new latent node 'V', which accounts for the 'true' preference values ,and node 'Zr', which accounts for different rating behaviors. First, according to Figure 3, the user node 'Y' is determined jointly by nodes 'Zy' and 'Zr', namely users are distinguished from each other both by their interests encode in 'Zy' and by their rating patterns encoded in 'Zr'. Secondly, the preference node 'V' is determined jointly by the node 'Zx' and node 'Zy', e.g. P(v|Zx, Zy). Thirdly, the rating is generated by the preference node and the node 'Zr'. Therefore, it is not necessary that an item with a high rating be truly favored by the user due to the dependency of node 'Zx' and 'Zy'.

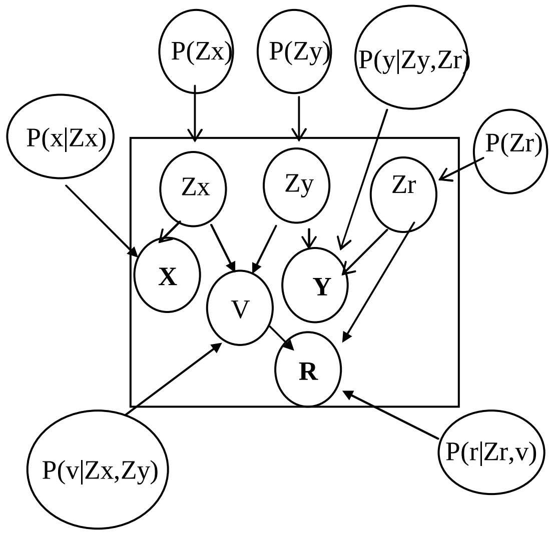

Figure 3. Graphical model representation of the extension of flexible mixture model to group rating patterns.

Finally, in the extended version of FMM model, two latent variables 'Zx' and 'Zy' are not coupled through the rating information 'R' as in Figure 2. Instead, they are connected directly through the preference node 'V' and the rating information 'R' can influence the correlation between nodes 'Zx' and 'Zy' indirectly through the preference node 'V'.

Let the preference variable 'V' be a continuous random variable ranging from 0 to 1(e.g. $0 \le v \le 1$). With the new graphic model, the joint probability can be calculated as:

$$P(x, y^t, r) = \sum_{Z_x, Z_y, Z_r} \int_v P(Z_x)P(Z_y)P(Z_r)P(x \mid Z_x)P(y^t \mid Z_y, Z_r)P(v \mid Z_x, Z_y)P(r \mid Z_r, v) \quad (19)$$

where $P(Z_r)$ and $P(r \mid Z_r, v)$ are multinomial distributions, and $P(v \mid Z_x, Z_y)$ is normal distribution.

Unfortunately, due to the introduction of the preference node 'V' and the latent rating node 'Zr', the inference and prediction processes are rather complex and time-consuming. Considering that the collaborative filtering task usually demands high efficiency, instead of using the above complex graphical model directly, we explore a simple model that is able to compute the preference values for a user given a set of rated items of that user. With this simple model, we are able to obtain the preference value 'v' directly instead of having to infer it from the graphic model in Figure 3. Then, a FMM model similar to Figure 2 can be used for computing the ratings for the test user by simply replacing the rating node 'R' with the preference node 'V'.

4.1 The Decoupled Models of Preferences and Ratings

So we need to calculate the preference value on an item with specific rating 'r' for a user who gives a set of rated items. Two factors can influence this value: 1) the percentage of items that have been rated no more than '*r*'. The larger the number of items rated no more than '*r*', the more likely that the user prefers the item. 2) The percentage of items that have been rated as '*r*'. The larger the number of items rated as '*r*', the less likely that the item is preferred by the user. Based on this intuition, we let $P\_R_y(r)$ $(0 \le P\_R_y(r) \le 1 \;\; , r \in \{1,..,R\})$ be the likelihood function for user y on an item, which he rates with rating 'r'; the vector $C_y = \{c_y(1),....,c_y(R)\}$ stand for the rating count vector, which counts how many times the user y has rated items with specific ratings. $P\_R_y(r)$ is computed as:

$$P\_R_y(r) = P(Rating \le r \mid y) - P(Rating = r \mid y)/2$$
$$= \sum_{r' \le r} \frac{c_y(r')}{\sum_{r''=1}^{R} c_y(r'')} - \frac{c_y(r)}{2 * \sum_{r''=1}^{R} c_y(r'')} \qquad (20)$$

This procedure can be seen in another way as somehow normalizing the user ratings into user preference. When there are very few given ratings from the user, Equation (20) may become unreliable. A better solution is a smoothed version of the $P(Rating = r \mid y)$ function, which utilizes the rating patterns of similar users. The similarity coefficient between the user y and y' is defined as the probability of mistaking user y' given the rating patterns of user y:

$$w^R_{y,y'} = P(y' \mid C_y) = \frac{P(C_y \mid y')P(y')}{\sum_{y''} P(C_y \mid y'')P(y'')} = \frac{P(C_y \mid y')}{\sum_{y''} P(C_y \mid y'')} \qquad (21)$$

The last step is derived by assuming a uniform distribution on P(y). And $P(C_y \mid y')$ is computed as:

$$P(C_y \mid y') = \prod_{r=1}^{R} \left(\frac{c_{y'}(r)}{\sum_{r'=1}^{R} c_{y'}(r')} \right)^{C_y(r)} \qquad (22)$$

The smoothed version of $P(Rating = r \mid y)$ is as follows:

$$P^{'}(Rating = r \mid y)$$
$$= \frac{w^R_{y,y} P(Rating = r \mid y) + \sum_{y' \in Y} w^R_{y,y'} P(Rating = r \mid y')}{w^R_{y,y} + \sum_{y' \in Y} w^R_{y,y'}} \qquad (23)$$

Plugging Equation (23) into Equation (20), we can convert a rating '*r*' given by a user '*y*' into the likelihood of being preferred. Now, let us consider the opposite, namely how to convert an estimated likelihood of being preferred $V_y(x)$ into an appropriate rating '*r*'. We simply find the rank that leads to the preference probability $P\_R_y(r)$ closest to the estimated preference value $\hat{V}_y(x)$:

$$\hat{R}_y(x) = \arg\min_r \mid P\_R_y(r) - \hat{V}_y(x) \mid \qquad (24)$$

4.2 The Combination of the FMM and the DM

The basic idea of combining FMM with the decoupled models (DM) is to, first convert the ratings in the training database into their corresponding preference values using the DM model. Then similar to the graphic model in Figure 2, a FMM is built over preference values instead of the ratings (e.g. replacing the rating node 'R' with the preference node 'V' in Figure 2). More specifically, the joint probability P(x,y,v) for a user '*y*', an item '*x*', and a preference value '*v*', is computed as:

$$P(x_{(l)}, y_{(l)}, v_{(l)})$$
$$= \sum_{Z_x, Z_y} P(Z_x)P(Z_y)P(x_{(l)} \mid Z_x)P(y_{(l)} \mid Z_y)P(v_{(l)} \mid Z_x, Z_y) \qquad (25)$$

where $v_{(l)}$ is the preference value computed from Equation (20). Compared with Equation (10), Equation (25) contains term $P(v_{(l)} \mid Z_x, Z_y)$ instead of $P(r_{(l)} \mid Z_x, Z_y)$. For simplicity, we assume $P(v_{(l)} \mid Z_x, Z_y)$ to be a normal distribution, i.e.

$$P(v_{(l)} \mid Z_x, Z_y) = \frac{1}{\sqrt{2\pi}\sigma_{Zx,Zy}} \exp\left(-\frac{1}{2\sigma^2_{Zx,Zy}}(v_{(l)} - u_{Zx,Zy})^2\right) \qquad (26)$$

The updating equations for the means and standard deviations are derived by AEM algorithm in Section 3.2.

With this modified FMM model using preference value, we will be able to compute the estimated preference value '*v*' for an item *x* given the user y, e.g. $\hat{V}_y(x)$. The final rating is computed by converting the estimated preference value $\hat{V}_y(x)$ into rating $\hat{R}_y(x)$ using Equation (24).

5. Experiments

In this section, we will present experiment results in order to address two issues. 1) Is the FMM more effective than other collaborative filtering algorithms? In the experiment, we will compare the proposed FMM model to other popular algorithms. 2) Can the FMM be further improved by combining it with the decoupled models (DM) as discussed in Section 4? In the experiment, we will compare the performance of FMM with and without the decoupled models (DM).

Two datasets of movie ratings are used in our experiments. The first one is the 'MovieRating' testbed[1]. The second testbed comes from the 'EachMovie'[2], where a subset of 2000 users with more than 40 ratings was extracted. The details of these two datasets are listed in Table 1. To compare different algorithms in a large spectrum, we tried several different configurations. For the MovieRating testbed, we set the first 100 or 200 users to be the training users. For the EachMovie testbed, the first 200 or 400 users were used. Furthermore, 5, 10 or 20

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

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

Table 1. Characteristics of the MovieRating Testbed and the EachMovie Testbed.

	MovieRating	EachMovie
Number of Users	500	2000
Number of Items	1000	1682
Average Number of Rated Items Per User	87.7	129.6
Number of Ratings	5	6

Table 2. Experiment Results (MAE) on the MovieRaing Testbed. A smaller value means a better performance.

Training Users Size	Algorithms	5 Items Given	10 Items Given	20 Items Given
100	PCC	0.881	0.832	0.809
	VS	0.859	0.834	0.823
	PD	0.839	0.826	0.818
	AM_a *(5)*	0.882	0.856	0.836
	AM_b *(2)*	0.869	0.857	0.850
	FMM	**0.829**	**0.822**	**0.807**
200	PCC	0.878	0.828	0.801
	VS	0.862	0.950	0.854
	PD	0.835	0.816	0.806
	AM_a *(5)*	0.891	0.850	0.818
	AM_b *(4)*	0.837	0.833	0.825
	FMM	**0.800**	**0.787**	**0.768**

Table 3. Experiment Results (MAE) on the EachMovie Testbed. A smaller value means a better performance.

Training Users Size	Algorithms	5 Items Given	10 Items Given	20 Items Given
200	PCC	1.22	1.16	1.13
	VS	1.25	1.24	1.26
	PD	1.19	1.16	1.15
	AM_a *(20)*	1.27	1.18	1.14
	AM_b *(10)*	1.18	1.17	1.16
	FMM	**1.07**	**1.04**	**1.02**
400	PCC	1.22	1.16	1.13
	VS	1.32	1.33	1.37
	PD	1.18	1.16	1.15
	AM_a *(20)*	1.28	1.19	1.16
	AM_b *(10)*	1.15	1.14	1.13
	FMM	**1.05**	**1.03**	**1.01**

items were provided as exposed items for a test user on both these two testbeds. As we believe that it is had for collaborative filtering system to collect huge amount of training data before it can provide recommendation service to the customers (so it is more important to evaluate the system performance with a limit number of training users), a relatively small number of training users were used in our experiments. But other experiments with more training users (300 and 400 for MovieRating, 800 and 1000 EachMovie) were conducted. As the limit of space, the results are not reported here but they are consistent with the results reported (the proposed FMM model got the best performance in all cases).

The evaluation metric used in our experiments was the commonly used mean absolute error (MAE), which is the average absolute deviation of the predicted ratings from the actual ratings on items the test users have voted.

$$MAE = \frac{1}{L_{Test}} \sum_{l} | r_{(l)} - \hat{R}_{y^t}(x_{(l)}) | \qquad (27)$$

where L_{Test} is the number of the test ratings.

5.1 Experiment Results

The first set of experiments is shown in Table 2 and Table 3. In addition to the proposed FMM model, two memory-based and three model-based approaches are evaluated. They are: the Pearson Correlation Coefficient method (PCC), the Vector Similarity method (VS), the aspect model using extension Equation (2) (AM_a), the aspect model with extension Equation (3) (AM_b), and the Personality Diagnosis model (PD). The two-sided clustering model is not included because previous studies have shown that its performance is substantially inferior to the aspect models. The number of user classes and the number of item classes in the FMM were set to 10 and 20 for users and items separately without much tuning. (Varying the number of classes from 5*10 to 20*40 gives us similar results to those reported in Table 2 and 3).

According to Table 2 and 3, the proposed new FMM performs better than all the other algorithms on all different configurations in terms of the MAE. Furthermore, consistent with (Pennock et al., 2000), the PD method achieves the second best performance, and is generally better than the other four methods (except AM_b on Each Movie with 400 training users). The number of aspects in the AM_a and AM_b were turned for the best performance (shown in italic in Table 2 and 3). As for the same number of aspects, AM_b has much more parameters than AM_a, thus it can be seen that the optimal number of aspects in AM_b model is smaller than AM_a. Since the proposed FMM model is similar to the aspect model except for the explicitly modeling of user and item clusters, we attribute the good performance of FMM to its ability of modeling the classes of users and items separately. Furthermore, the difference between the proposed model and the two-sided clustering model suggests that it is beneficial not to assume that each user (item) belongs to a single class.

The second set of experiments is shown in Table 4 and Table 5, where we compare the performance of FMM with and without the decoupled models (DM). It is clear that the combination of FMM model with DM model outperforms the basic FMM in all configurations. Therefore, it is important to cluster users with similar preference patterns instead of rating patterns. It is interesting to further explore efficient inference and prediction algorithms for the graphical model in Figure 3, which is able to simultaneously group users with similar preference patterns, rating patterns and items with similar characteristics and therefore may result in even more improvement in the prediction accuracy.

Table 4. Experiment Results (MAE) on the MovieRaing Testbed for FMM and FMM plus Decoupled models (DM). A smaller value means a better performance.

Training Users Size	Algorithms	5 Items Given	10 Items Given	20 Items Given
100	FMM	0.829	0.822	0.807
	FMM+DM	**0.791**	**0774**	**0.751**
200	FMM	0.800	0.787	0.768
	FMM+DM	**0.770**	**0.753**	**0.730**

Table 5. Experiment Results (MAE) on the EachMovie Testbed for FMM and FMM plus Decoupled models. A smaller value means a better performance.

Training Users Size	Algorithms	5 Items Given	10 Items Given	20 Items Given
200	FMM	1.07	1.04	1.02
	FMM+DM	**1.06**	**1.02**	**1.00**
400	FMM	1.05	1.03	1.01
	FMM+DM	**1.04**	**1.01**	**0.99**

6. Conclusion and Future Work

Partition or clustering techniques have been studied intensively in the previous work for collaborative filtering. In this work, we proposed a formal graphical model for collaborative filtering, named flexible mixture model (FMM). The new model tries to address three issues in collaborative filtering: 1) explicitly modeling both classes of users and items by taking into account their correlations; 2) allowing each user and item to belong to multiple clusters simultaneously; 3) clustering users with similar preference patterns instead of rating patterns. Experiments on two common testbeds with several different configurations indicated that the proposed model is able to outperform five other algorithms for collaborative filtering task substantially.

The combination method of the flexible mixture model and the decoupled models is rather preliminary. As future work, we plan to explore better efficient approximation algorithms for inference and prediction with the complex graphical model, which can simultaneously group users with similar preference patterns, rating patterns and items with similar characteristics. Finally, the proposed FMM model is only applied and evaluated on the problem of predicting item rating for new users; we hope to extend this model for other tasks such as recommending new items to known users in the further work.

Acknowledgements

We thank ChengXiang Zhai for his helpful discussion, Paul Ogilvie and Jamie Callan for their useful comments and help to improve the English in this paper.

References

Basu, C., & Hirsh, H. (1998). Recommendation as classification: Using social and content-based information in recommendation. *In the Proceedings of Fifteenth National Conference on Artificial Intelligence.*

Breese J. S., Heckerman D., Kadie C. (1998). Empirical Analysis of Predictive Algorthms for Collaborative Filtering. *In the Proceeding of the Fourteenth Conference on Uncertainty in Artificial Intelligence.*

O'Connor, M. & Herlocker, Jon. (2001). Clustering Items for Collaborative Filtering. *In the Proceedings of SIGIR-2001 Workshop on Recommender Systems, New Orleans, LA.*

Dempster, A. P., Laird, N. M., & Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society,* B39: 1-38.

Hofmann, T., & Puzicha, J. (1999). Latent Class Models for Collaborative Filtering. *In the Proceedings of International Joint Conference on Artificial Intelligence.*

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

Pennock, D. M., Horvitz, E., Lawrence, S., & Giles, C. L. (2000). Collaborative Filtering by Personality Diagosis: A Hybrid Memory- and Model-Based Approach. *In the Proceeding of the Sixteenth Conference on Uncertainty in Artificial Intelligence.*

Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., & Riedl, J. (1994). Grouplens: An Open Architecture for Collaborative Filtering of Netnews. *In Proceeding of the ACM 1994 Conference on Computer Supported Cooperative Work.*