# Transfer Learning for Collaborative Filtering via a Rating-Matrix Generative Model

**Bin Li** LIBIN@FUDAN.EDU.CN
School of Computer Science, Fudan University, Shanghai 200433, China

**Qiang Yang** QYANG@CSE.UST.HK
Dept. of Computer Science & Engineering, Hong Kong University of Science & Technology, Hong Kong, China

**Xiangyang Xue** XYXUE@FUDAN.EDU.CN
School of Computer Science, Fudan University, Shanghai 200433, China

## Abstract

Cross-domain collaborative filtering solves the sparsity problem by transferring rating knowledge across multiple domains. In this paper, we propose a rating-matrix generative model (RMGM) for effective cross-domain collaborative filtering. We first show that the relatedness across multiple rating matrices can be established by finding a shared implicit cluster-level rating matrix, which is next extended to a cluster-level rating model. Consequently, a rating matrix of any related task can be viewed as drawing a set of users and items from a user-item joint mixture model as well as drawing the corresponding ratings from the cluster-level rating model. The combination of these two models gives the RMGM, which can be used to fill the missing ratings for both existing and new users. A major advantage of RMGM is that it can share the knowledge by pooling the rating data from multiple tasks even when the users and items of these tasks do not overlap. We evaluate the RMGM empirically on three real-world collaborative filtering data sets to show that RMGM can outperform the individual models trained separately.

## 1. Introduction

Collaborative filtering (CF) in recommender systems aims at predicting an active user's ratings on a set of items based on a collection of like-minded users' rating records on the same set of items. Various CF methods have been proposed in the last decade. For example, memory-based methods (Resnick et al., 1994; Sarwar et al., 2001) find $K$-nearest neighbors based on some similarity measure. Model-based methods (Hofmann & Puzicha, 1999; Pennock et al., 2000; Si & Jin, 2003) learn prference/rating models for similar users (and items). Matrix factorization methods (Srebro & Jaakkola, 2003) find a low-rank approximation for the rating matrix. Most of these methods are based on the available ratings in the given rating matrix. Thus, the performance of these methods largely depends on the density of the given rating matrix.

However, in real-world recommender systems, users can rate a very limited number of items. Thus, the rating matrix is often extremely sparse. As a result, the available rating data that can be used for $K$-NN search, probabilistic modeling, or matrix factorization are radically insufficient. The sparsity problem has become a major bottleneck for most CF methods.

To alleviate the sparsity problem in collaborative filtering, one promising approach is to pool together the rating data from multiple rating matrices in related domains for knowledge transfer and sharing. In the real world, many web sites for recommending similar items, e.g., movies, books, and music, are closely related. On one hand, since many of these items are literary and entertainment works, they should share some common properties (e.g., genre and style). On the other hand, since these web services are geared towards the general population, users of these services, and the items interested by them, should share some properties as well. However, much of the shared knowledge across multiple related domains may be well hidden, and few

Appearing in *Proceedings of the 26th International Conference on Machine Learning*, Montreal, Canada, 2009. Copyright 2009 by the author(s)/owner(s).

studies have been done to uncover this knowledge.

In this paper, we solve the problem of learning a rating-matrix generative model from a set of rating matrices in multiple related recommender systems (domains) for collaborative filtering. Our aim is to alleviate the sparsity problem in individual rating matrices by discovering what is common among them. We first show that the relatedness across multiple rating matrices can be established by sharing an implicit cluster-level rating matrix. Then, we extend the shared cluster-level rating matrix to a more general cluster-level rating model, which defines a rating function in terms of the latent user- and item-cluster variables. Consequently, a rating matrix of any related task can be viewed as drawing a set of users and items from a user-item joint mixture model as well as drawing the corresponding ratings from the cluster-level rating model. The combination of these two models gives the rating-matrix generative model (RMGM). We also propose an algorithm for training the RMGM on the pooled rating data from multiple related rating matrices as well as an algorithm for predicting the missing ratings for new users in different tasks. Experimental comparison is carried out on the three real-world CF data sets. The results show that our proposed RMGM learned from multiple CF tasks can outperform the individual models trained separately.

The remainder of the paper is organized as follows. In Section 2, we first introduce the problem setting for cross-domain collaborative filtering and the notations used in this paper. In Section 3, we describe how to establish the relatedness across multiple rating matrices via a shared cluster-level rating matrix. The RMGM is presented in Section 4 as well as the training and prediction algorithms. Related work is introduced in Section 5. We experimentally validate the effectiveness of the RMGM for cross-domain collaborative filtering in Section 6 and conclude the paper in Section 7.

## 2. Problem Setting

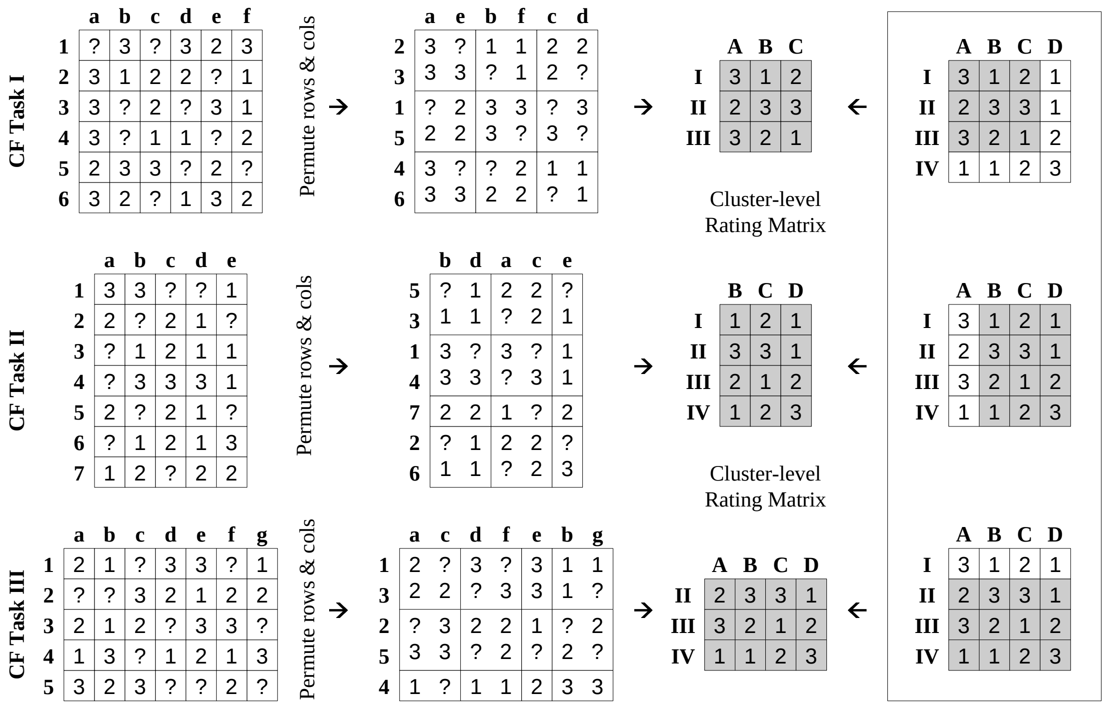
Suppose that we are given $Z$ rating matrices in related domains for collaborative filtering. In the $z$-th rating matrix, a set of users, $U_z = \{u_1^{(z)}, \ldots, u_{n_z}^{(z)}\} \subset \mathcal{U}$, make ratings on a set of items, $V_z = \{v_1^{(z)}, \ldots, v_{m_z}^{(z)}\} \subset \mathcal{V}$, where $n_z$ and $m_z$ denote the numbers of rows (users) and columns (items), respectively. The random variables $u$ and $v$ are assumed to be independent from each other. To consider the more difficult case, we assume that neither the user sets nor the item sets in the given rating matrices have intersections, i.e., $\bigcap_z U_z = \emptyset$ and $\bigcap_z V_z = \emptyset$ (in fact, there may exist intersections, but they are unobservable). The rating data in the $z$-th rating matrix is a set of triplets $D_z = \{(u_1^{(z)}, v_1^{(z)}, r_1^{(z)}), \ldots, (u_{s_z}^{(z)}, v_{s_z}^{(z)}, r_{s_z}^{(z)})\}$, where $s_z$ is the number of available ratings in the $z$-th rating matrix. The ratings in $\{D_1, \ldots, D_Z\}$ should be in the same rating scales $R$ (e.g., $1 - 5$).

For model-based CF methods, a preference/rating model, e.g., the aspect model (Hofmann & Puzicha, 1999), can be trained on $D_z$ for the $z$-th task. In our cross-domain collaborative filtering setting, we wish to train a rating-matrix generative model (RMGM) for all the given related tasks on the pooled rating data, namely, $\bigcup_z D_z$. Then, the $z$-th rating matrix can be viewed as drawing a set of users, $U_z$, and a set of items, $V_z$, from the learned RMGM. The missing values in the $z$-th rating matrix can be generated by the RMGM.

## 3. Cluster-Level Rating Matrix as Knowledge Sharing

To allow knowledge-sharing across multiple rating matrices, we first investigate how to establish the relatedness among the given tasks. A difficulty is that no explicit correspondence among the user sets or the item sets in the given rating matrices can be exploited. However, some collaborative filtering tasks are somewhat related in certain aspects. Take movie-rating and book-rating web sites for example. On one hand, movies and books have correspondence in genre. On the other hand, although the user sets are different from one another, they are the subsets sampled from the same population (this assumption only holds for popular web sites) and should reflect similar social aspects (Coyle & Smyth, 2008).

The above observation suggests that, although we can not find an explicit correspondence among individual users or items, we can establish a *cluster-level rating-pattern representation* as a "bridge" to connect all the related rating matrices. Figure 1 illustrates how the implicit relatedness among three artificially generated rating matrices is established via a cluster-level rating matrix. By permuting the rows and columns (which is equivalent to co-clustering) in each rating matrix, we can obtain three block rating matrices. Each block comprises a set of ratings provided by a user group on an item group. We can further reduce the block matrices to be the cluster-level rating matrices, in which each row corresponds to a user cluster and each column an item cluster. The entries in the cluster-level rating matrices are the average ratings of the corresponding user-item co-clusters. The resulting cluster-level rating matrices reveal that the three rating matrices implicitly share a common $4 \times 4$ cluster-level rating-pattern representation.



CF Task I

| | a | b | c | d | e | f |
|---|---|---|---|---|---|---|
| 1 | ? | 3 | ? | 3 | 2 | 3 |
| 2 | 3 | 1 | 2 | 2 | ? | 1 |
| 3 | 3 | ? | 2 | ? | 3 | 1 |
| 4 | 3 | ? | 1 | 1 | ? | 2 |
| 5 | 2 | 3 | 3 | ? | 2 | ? |
| 6 | 3 | 2 | ? | 1 | 3 | 2 |

| | a | e | b | f | c | d |
|---|---|---|---|---|---|---|
| 2 | 3 | ? | 1 | 1 | 2 | 2 |
| 3 | 3 | 3 | ? | 1 | 2 | ? |
| 1 | ? | 2 | 3 | 3 | ? | 3 |
| 5 | 2 | 2 | 3 | ? | 3 | ? |
| 4 | 3 | ? | ? | 2 | 1 | 1 |
| 6 | 3 | 3 | 2 | 2 | ? | 1 |

| | A | B | C |
|---|---|---|---|
| I | 3 | 1 | 2 |
| II | 2 | 3 | 3 |
| III | 3 | 2 | 1 |

| | A | B | C | D |
|---|---|---|---|---|
| I | 3 | 1 | 2 | 1 |
| II | 2 | 3 | 3 | 1 |
| III | 3 | 2 | 1 | 2 |
| IV | 1 | 1 | 2 | 3 |

CF Task II

| | a | b | c | d | e |
|---|---|---|---|---|---|
| 1 | 3 | 3 | ? | ? | 1 |
| 2 | 2 | ? | 2 | 1 | ? |
| 3 | ? | 1 | 2 | 1 | 1 |
| 4 | ? | 3 | 3 | 3 | 1 |
| 5 | 2 | ? | 2 | 1 | ? |
| 6 | ? | 1 | 2 | 1 | 3 |
| 7 | 1 | 2 | ? | 2 | 2 |

| | b | d | a | c | e |
|---|---|---|---|---|---|
| 5 | ? | 1 | 2 | 2 | ? |
| 3 | 1 | 1 | ? | 2 | 1 |
| 1 | 3 | ? | 3 | ? | 1 |
| 4 | 3 | 3 | ? | 3 | 1 |
| 7 | 2 | 2 | 1 | ? | 2 |
| 2 | ? | 1 | 2 | 2 | ? |
| 6 | 1 | 1 | ? | 2 | 3 |

| | B | C | D |
|---|---|---|---|
| I | 1 | 2 | 1 |
| II | 3 | 3 | 1 |
| III | 2 | 1 | 2 |
| IV | 1 | 2 | 3 |

| | A | B | C | D |
|---|---|---|---|---|
| I | 3 | 1 | 2 | 1 |
| II | 2 | 3 | 3 | 1 |
| III | 3 | 2 | 1 | 2 |
| IV | 1 | 1 | 2 | 3 |

CF Task III

| | a | b | c | d | e | f | g |
|---|---|---|---|---|---|---|---|
| 1 | 2 | 1 | ? | 3 | 3 | ? | 1 |
| 2 | ? | ? | 3 | 2 | 1 | 2 | 2 |
| 3 | 2 | 1 | 2 | ? | 3 | 3 | ? |
| 4 | 1 | 3 | ? | 1 | 2 | 1 | 3 |
| 5 | 3 | 2 | 3 | ? | ? | 2 | ? |

| | a | c | d | f | e | b | g |
|---|---|---|---|---|---|---|---|
| 1 | 2 | ? | 3 | ? | 3 | 1 | 1 |
| 3 | 2 | 2 | ? | 3 | 3 | 1 | ? |
| 2 | ? | 3 | 2 | 2 | 1 | ? | 2 |
| 5 | 3 | 3 | ? | 2 | ? | 2 | ? |
| 4 | 1 | ? | 1 | 1 | 2 | 3 | 3 |

| | A | B | C | D |
|---|---|---|---|---|
| II | 2 | 3 | 3 | 1 |
| III | 3 | 2 | 1 | 2 |
| IV | 1 | 1 | 2 | 3 |

| | A | B | C | D |
|---|---|---|---|---|
| I | 3 | 1 | 2 | 1 |
| II | 2 | 3 | 3 | 1 |
| III | 3 | 2 | 1 | 2 |
| IV | 1 | 1 | 2 | 3 |

*Figure 1.* Sharing cluster-level user-item rating patterns among three toy rating matrices in different domains. The missing values are denoted by '?'. After permuting the rows (users) and columns (items) in each rating matrix, it is revealed that the three rating matrices implicitly share a common $4 \times 4$ cluster-level rating matrix.

This toy example shows an ideal case in which the users and items in the same cluster behave exactly the same. In many real-world cases, since users may have multiple personalities and items may have multiple attributes, a user or an item can simultaneously belong to multiple clusters with different memberships. Thus, we need to introduce *softness* to clustering models. Suppose there are $K$ user clusters, $\{c_{\mathcal{U}}^{(1)}, \ldots, c_{\mathcal{U}}^{(K)}\}$, and $L$ item clusters, $\{c_{\mathcal{V}}^{(1)}, \ldots, c_{\mathcal{V}}^{(L)}\}$, in the shared cluster-level rating patterns. The membership of a user-item pair $(u, v)$ to a user-item co-cluster $(c_{\mathcal{U}}^{(k)}, c_{\mathcal{V}}^{(l)})$ is the joint posterior membership probability $P(c_{\mathcal{U}}^{(k)}, c_{\mathcal{V}}^{(l)}|u, v)$. Furthermore, a user-item co-cluster can also have multiple ratings with different probabilities $P(r|c_{\mathcal{U}}^{(k)}, c_{\mathcal{V}}^{(l)})$. Then, we can define the rating function $f_R(u, v)$ for a user $u$ on an item $v$ in terms of the two latent cluster variables $c_{\mathcal{U}}^{(k)}$ and $c_{\mathcal{V}}^{(l)}$

$$\begin{aligned} f_R(u,v) &= \sum_r rP(r|u,v) \\ &= \sum_r r \sum_{k,l} P(r|c_{\mathcal{U}}^{(k)}, c_{\mathcal{V}}^{(l)}) P(c_{\mathcal{U}}^{(k)}, c_{\mathcal{V}}^{(l)}|u,v) \\ &= \sum_r r \sum_{k,l} P(r|c_{\mathcal{U}}^{(k)}, c_{\mathcal{V}}^{(l)}) P(c_{\mathcal{U}}^{(k)}|u) P(c_{\mathcal{V}}^{(l)}|v), \quad (1) \end{aligned}$$

where (1) is obtained based on the assumption that random variables $u$ and $v$ are independent.

We can further rewrite (1) in the form of matrices

$$f_R(u,v) = \mathbf{p}_u^\top \mathbf{B} \mathbf{p}_v, \quad \mathbf{p}_u^\top \mathbf{1} = 1, \mathbf{p}_v^\top \mathbf{1} = 1, \qquad (2)$$

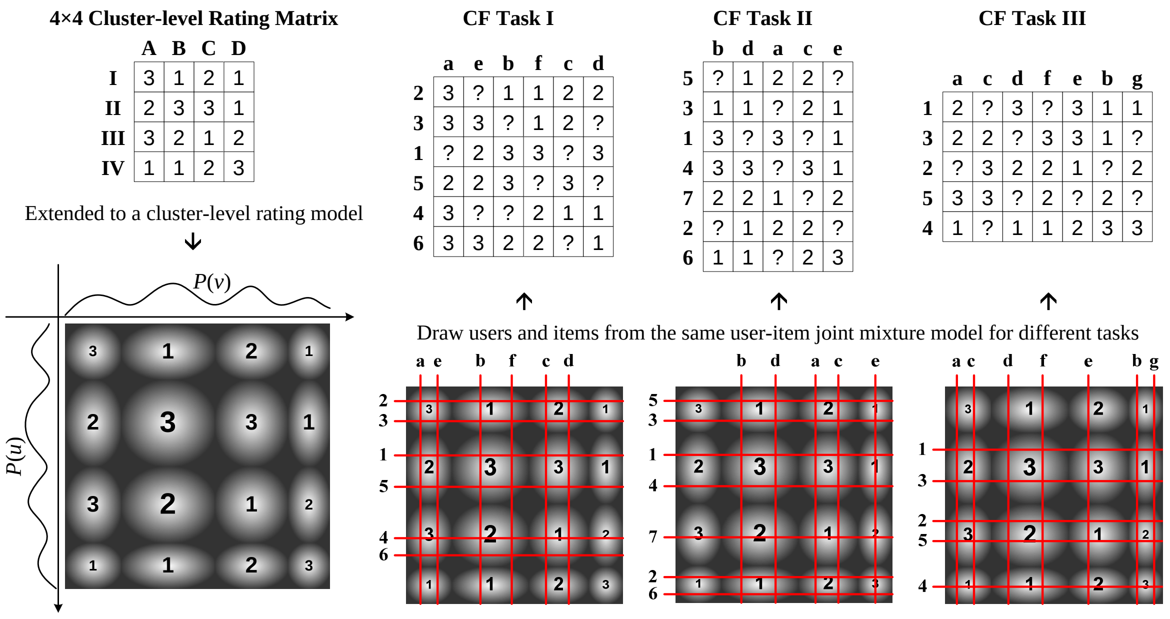
where $\mathbf{p}_u \in \mathbb{R}^K$ and $\mathbf{p}_v \in \mathbb{R}^L$ are the user- and item-cluster membership vectors ($[\mathbf{p}_u]_k = P(c_{\mathcal{U}}^{(k)}|u)$ and $[\mathbf{p}_v]_l = P(c_{\mathcal{V}}^{(l)}|v)$), and $\mathbf{B}$ is a $K \times L$ relaxed cluster-level rating matrix in which an entry can have multiple ratings with different probabilities

$$\mathbf{B}_{kl} = \sum_r rP(r|c_{\mathcal{U}}^{(k)}, c_{\mathcal{V}}^{(l)}). \qquad (3)$$

Eq. (2) implies that the relaxed cluster-level rating matrix $\mathbf{B}$ is a cluster-level rating model. In the next section, we focus on learning the user-item joint mixture model as well as the shared cluster-level rating model on the pooled rating data from multiple related tasks.

## 4. Rating-Matrix Generative Model

In order to extend the shared cluster-level rating matrix to a more general cluster-level rating model, we


4×4 Cluster-level Rating Matrix
CF Task I
CF Task II
CF Task III
Extended to a cluster-level rating model
P(v)
P(u)
Draw users and items from the same user-item joint mixture model for different tasks

4×4 Cluster-level Rating Matrix

| | A | B | C | D |
|---|---|---|---|---|
| I | 3 | 1 | 2 | 1 |
| II | 2 | 3 | 3 | 1 |
| III | 3 | 2 | 1 | 2 |
| IV | 1 | 1 | 2 | 3 |

CF Task I

| | a | e | b | f | c | d |
|---|---|---|---|---|---|---|
| 2 | 3 | ? | 1 | 1 | 2 | 2 |
| 3 | 3 | 3 | ? | 1 | 2 | ? |
| 1 | ? | 2 | 3 | 3 | ? | 3 |
| 5 | 2 | 2 | 3 | ? | 3 | ? |
| 4 | 3 | ? | ? | 2 | 1 | 1 |
| 6 | 3 | 3 | 2 | 2 | ? | 1 |

CF Task II

| | b | d | a | c | e |
|---|---|---|---|---|---|
| 5 | ? | 1 | 2 | 2 | ? |
| 3 | 1 | 1 | ? | 2 | 1 |
| 1 | 3 | ? | 3 | ? | 1 |
| 4 | 3 | 3 | ? | 3 | 1 |
| 7 | 2 | 2 | 1 | ? | 2 |
| 2 | ? | 1 | 2 | 2 | ? |
| 6 | 1 | 1 | ? | 2 | 3 |

CF Task III

| | a | c | d | f | e | b | g |
|---|---|---|---|---|---|---|---|
| 1 | 2 | ? | 3 | ? | 3 | 1 | 1 |
| 3 | 2 | 2 | ? | 3 | 3 | 1 | ? |
| 2 | ? | 3 | 2 | 2 | 1 | ? | 2 |
| 5 | 3 | 3 | ? | 2 | ? | 2 | ? |
| 4 | 1 | ? | 1 | 1 | 2 | 3 | 3 |

*Figure 2.* Each rating matrix can be viewed as drawing a set of users (horizontal straight lines) and items (vertical straight lines) from the same user-item joint mixture model (the joint probability of a user-item pair is indicted by gray-scales) as well as drawing the corresponding ratings (the crossing points of the horizontal and vertical lines) from a shared cluster-level rating model (the figures denote the ratings which are most likely to be obtained in those co-clusters).

should first define a user-item bivariate probability histogram over $\mathcal{U} \times \mathcal{V}$. Let $P_{\mathcal{U}}(u)$ and $P_{\mathcal{V}}(v)$ denote the marginal distributions for users and items, respectively. The user-item bivariate probability histogram is a $|\mathcal{U}| \times |\mathcal{V}|$ matrix, $\mathbf{H}$, which is defined as the user-item joint distribution

$$\mathbf{H}_{uv} = P(u, v) = P_{\mathcal{U}}(u) P_{\mathcal{V}}(v). \tag{4}$$

Thus, the user-item pairs for all the given tasks can be drawn from $\mathbf{H}$

$$\left(u_i^{(z)}, v_i^{(z)}\right) \sim \Pr(\mathbf{H}), \tag{5}$$

for $z = 1, \ldots, Z; i = 1, \ldots, s_z$.

Based on the assumption that there are $K$ clusters in $\mathcal{U}$ and $L$ clusters in $\mathcal{V}$, we can model the user and item marginal distributions in the form of mixture models, in which each component corresponds to a latent user/item cluster

$$P_{\mathcal{U}}(u) = \sum_k P(c_{\mathcal{U}}^{(k)}) P(u|c_{\mathcal{U}}^{(k)}), \tag{6}$$

$$P_{\mathcal{V}}(v) = \sum_l P(c_{\mathcal{V}}^{(l)}) P(v|c_{\mathcal{V}}^{(l)}), \tag{7}$$

where $P(c_{\mathcal{U}}^{(k)})$ denotes the prior for the user cluster $c_{\mathcal{U}}^{(k)}$ and $P(u|c_{\mathcal{U}}^{(k)})$ the conditional probability of a user $u$ given the user cluster $c_{\mathcal{U}}^{(k)}$. The user-item bivariate probability histogram (4) can be rewritten as

$$\mathbf{H}_{uv} = \sum_{k,l} P(c_{\mathcal{U}}^{(k)}) P(c_{\mathcal{V}}^{(l)}) P(u|c_{\mathcal{U}}^{(k)}) P(v|c_{\mathcal{V}}^{(l)}). \tag{8}$$

Then, the users and items can be drawn respectively from the user and the item mixture models which are in terms of the two latent cluster variables

$$\left(u_i^{(z)}, v_i^{(z)}\right) \sim \sum_{k,l} P(c_{\mathcal{U}}^{(k)}) P(c_{\mathcal{V}}^{(l)}) P(u|c_{\mathcal{U}}^{(k)}) P(v|c_{\mathcal{V}}^{(l)}). \tag{9}$$

Eq. (9) defines the *user-item joint mixture model*. Furthermore, the ratings also can be drawn from the conditional distributions given the latent cluster variables

$$r_i^{(z)} \sim P(r|c_{\mathcal{U}}^{(k)}, c_{\mathcal{V}}^{(l)}). \tag{10}$$

Eq. (10) defines the *cluster-level rating model*.

Combining (9) and (10), we can obtain the *rating-matrix generative model* (RMGM), which can generate rating matrices. Figure 2 illustrates the rating-matrix generating process on the three toy rating matrices. The $4 \times 4$ cluster-level rating matrix from Figure 1 is extended to a cluster-level rating model. Each rating matrix can thus be viewed as drawing a set of users $U_z$ and items $V_z$ from the user-item joint mixture model as well as drawing the corresponding ratings for $(U_z, V_z)$ from the cluster-level rating model. Generally speaking, each rating matrix can be viewed as drawing $D_z$ from the RMGM.

The formulation of RMGM is similar to the flexible mixture model (FMM) (Si & Jin, 2003). The major difference is that RMGM can generate rating matrices for different CF tasks (recall that $\bigcap_z U_z = \emptyset$ and $\bigcap_z V_z = \emptyset$ and the sizes of rating matrices are also different from one another). RMGM can be viewed as extending FMM to a multi-task version such that the user- and item-cluster variables are shared by and learned from multiple tasks. Furthermore, since the RMGM is trained on the pooled rating data from multiple tasks, the training and prediction algorithms for RMGM are also different from those for FMM.

### 4.1. Training the RMGM

In this section, we introduce how to train an RMGM on the pooled rating data $\bigcup_z D_z$. We need to learn five sets of model parameters in (9) and (10), i.e., $P(c_{\mathcal{U}}^{(k)})$, $P(c_{\mathcal{V}}^{(l)})$, $P(u|c_{\mathcal{U}}^{(k)})$, $P(v|c_{\mathcal{V}}^{(l)})$, and $P(r|c_{\mathcal{U}}^{(k)}, c_{\mathcal{V}}^{(l)})$, for $k = 1, \ldots, K$; $l = 1, \ldots, L$; $u \in \bigcup_z U_z$; $v \in \bigcup_z V_z$; and $r \in R$.

We adopt the Expectation Maximization (EM) algorithm (Dempster et al., 1977) for RMGM training. In the E-step, the joint posterior probability of $(c_{\mathcal{U}}^{(k)}, c_{\mathcal{V}}^{(l)})$ given $(u_i^{(z)}, v_i^{(z)}, r_i^{(z)})$ can be computed using the five sets of model parameters

$$P(c_{\mathcal{U}}^{(k)}, c_{\mathcal{V}}^{(l)}|u_i^{(z)}, v_i^{(z)}, r_i^{(z)}) = \qquad (11)$$
$$\frac{P(u_i^{(z)}, c_{\mathcal{U}}^{(k)})P(v_i^{(z)}, c_{\mathcal{V}}^{(l)})P(r_i^{(z)}|c_{\mathcal{U}}^{(k)}, c_{\mathcal{V}}^{(l)})}{\sum_{p,q} P(u_i^{(z)}, c_{\mathcal{U}}^{(p)})P(v_i^{(z)}, c_{\mathcal{V}}^{(q)})P(r_i^{(z)}|c_{\mathcal{U}}^{(p)}, c_{\mathcal{V}}^{(q)})}$$

and $P(u_i^{(z)}, c_{\mathcal{U}}^{(k)}) = P(c_{\mathcal{U}}^{(k)})P(u_i^{(z)}|c_{\mathcal{U}}^{(k)})$, $P(v_i^{(z)}, c_{\mathcal{V}}^{(l)}) = P(c_{\mathcal{V}}^{(l)})P(v_i^{(z)}|c_{\mathcal{V}}^{(l)})$.

In the M-step, the five sets of model parameters for $Z$ given tasks are updated as follows (let $P(k, l|j^{(z)})$ as a shorthand for $P(c_{\mathcal{U}}^{(k)}, c_{\mathcal{V}}^{(l)}|u_j^{(z)}, v_j^{(z)}, r_j^{(z)})$ for simplicity)

$$P(c_{\mathcal{U}}^{(k)}) = \frac{\sum_z \sum_l \sum_j P(k, l|j^{(z)})}{\sum_z s_z} \qquad (12)$$

$$P(c_{\mathcal{V}}^{(l)}) = \frac{\sum_z \sum_k \sum_j P(k, l|j^{(z)})}{\sum_z s_z} \qquad (13)$$

$$P(u_i^{(z)}|c_{\mathcal{U}}^{(k)}) = \frac{\sum_l \sum_{j:u_j^{(z)}=u_i^{(z)}} P(k, l|j^{(z)})}{P(c_{\mathcal{U}}^{(k)}) \sum_z s_z} \qquad (14)$$

$$P(v_i^{(z)}|c_{\mathcal{V}}^{(l)}) = \frac{\sum_k \sum_{j:v_j^{(z)}=v_i^{(z)}} P(k, l|j^{(z)})}{P(c_{\mathcal{V}}^{(l)}) \sum_z s_z} \qquad (15)$$

$$P(r|c_{\mathcal{U}}^{(k)}, c_{\mathcal{V}}^{(l)}) = \frac{\sum_z \sum_{j:r_j^{(z)}=r} P(k, l|j^{(z)})}{\sum_z \sum_j P(k, l|j^{(z)})}. \qquad (16)$$

In Eqs. (12–16), all the parameters in terms of the two latent cluster variables are computed using the pooled rating data $\bigcup_z D_z$. By alternating E-step and M-step, an RMGM which is fit onto a set of related CF tasks can be obtained. In particular, the user-item joint mixture model defined in (9) and the shared cluster-level rating model defined in (10) can be learned. A rating triplet $(u_i^{(z)}, v_i^{(z)}, r_i^{(z)})$ from any task can thus be viewed as drawing from the RMGM.

### 4.2. RMGM-Based Prediction

After training the RMGM, according to (1), the missing values in the $K$ given rating matrices can be generated by

$$\begin{aligned} f_R(u_i^{(z)}, v_i^{(z)}) \quad &= \quad \sum_r r \sum_{k,l} P(r|c_{\mathcal{U}}^{(k)}, c_{\mathcal{V}}^{(l)}) \\ &\qquad P(c_{\mathcal{U}}^{(k)}|u_i^{(z)})P(c_{\mathcal{V}}^{(l)}|v_i^{(z)}), \end{aligned} \qquad (17)$$

where $P(c_{\mathcal{U}}^{(k)}|u_i^{(z)})$ and $P(c_{\mathcal{V}}^{(l)}|v_i^{(z)})$ can be computed using the learned parameters based on the Bayes rule.

To predict the ratings on $V_z$ for a new user $u^{(z)}$ in the $z$-th task, we can solve a quadratic optimization problem to compute the user-cluster membership, $\mathbf{p}_{u^{(z)}} \in \mathbb{R}^K$, for $u^{(z)}$ based on the given ratings $\mathbf{r}_{u^{(z)}} \in \{R, 0\}^{m_z}$ (the unobserved ratings are set to 0)

$$\begin{aligned} \min_{\mathbf{p}_{u^{(z)}}} \quad & \left\| [\mathbf{B}\mathbf{P}_{V_z}]^\top \mathbf{p}_{u^{(z)}} - \mathbf{r}_{u^{(z)}} \right\|_{\mathbf{W}_{u^{(z)}}}^2 \\ \text{s.t.} \quad & \mathbf{p}_{u^{(z)}}^\top \mathbf{1} = 1. \end{aligned} \qquad (18)$$

In Eq. (18), $\mathbf{P}_{V_z}$ is an $L \times m_z$ item-cluster membership matrix, where $[\mathbf{P}_{V_z}]_{li} = P(c_{\mathcal{V}}^{(l)}|v_i^{(z)})$; $\mathbf{W}_{u^{(z)}}$ is an $m_z \times m_z$ diagonal matrix, where $[\mathbf{W}_{u^{(z)}}]_{ii} = 1$ if $[\mathbf{r}_{u^{(z)}}]_i$ is given and $[\mathbf{W}_{u^{(z)}}]_{ii} = 0$ otherwise. Here $\|\mathbf{x}\|_{\mathbf{W}}$ denotes a weighted $l_2$-norm, $\sqrt{\mathbf{x}^\top \mathbf{W} \mathbf{x}}$. The quadratic optimization problem (18) is very simple and can be solved by any quadratic solver. After obtaining the optimal user-cluster membership $\hat{\mathbf{p}}_{u^{(z)}}$ for $u^{(z)}$, the ratings of $u^{(z)}$ on $v_i^{(z)}$ can be predicted by

$$f_R(u^{(z)}, v_i^{(z)}) = \hat{\mathbf{p}}_{u^{(z)}}^\top \mathbf{B} \mathbf{p}_{v_i^{(z)}}, \qquad (19)$$

where $\mathbf{p}_{v_i^{(z)}}$ is the $i$-th column in $\mathbf{P}_{V_z}$. Similarly, based on the learned parameters, we can also predict the ratings of all the existing users in the $z$-th task on a new item. Due to space limitation, we skip the details.

### 4.3. Implementation Details

**Initialization**: Since the optimization problem for RMGM training is non-convex, the initialization for the five sets of model parameters is crucial for searching a better local maximum. We first select the densest rating matrix from the given tasks, and simultaneously cluster the rows (users) and columns (items) in that matrix using orthogonal nonnegative matrix tri-factorization (Ding et al., 2006) (other co-clustering methods are also applicable). Based on the co-clustering results, we can coarsely estimate $P(c_{\mathcal{U}}^{(k)})$, $P(c_{\mathcal{V}}^{(l)})$, and $P(r|c_{\mathcal{U}}^{(k)}, c_{\mathcal{V}}^{(l)})$. We use random values for initializing $P(u_i^{(z)}|c_{\mathcal{U}}^{(k)})$ and $P(v_i^{(z)}|c_{\mathcal{V}}^{(l)})$. Note that the five sets of initialized parameters should be respectively normalized: $\sum_k P(c_{\mathcal{U}}^{(k)}) = 1$, $\sum_l P(c_{\mathcal{V}}^{(l)}) = 1$, $\sum_r P(r|c_{\mathcal{U}}^{(k)}, c_{\mathcal{V}}^{(l)}) = 1$, $\sum_z \sum_i P(u_i^{(z)}|c_{\mathcal{U}}^{(k)}) = 1$, and $\sum_z \sum_i P(v_i^{(z)}|c_{\mathcal{V}}^{(l)}) = 1$.

**Regularization**: In order to avoid unfavorable local maxima, we also impose regularization on the EM algorithm (Hofmann & Puzicha, 1998). We adopt the same strategy used in (Si & Jin, 2003) and we skip this part for space limitation.

**Model Selection**: We need to set the numbers of user and item clusters, $K$ and $L$, to start with. The cluster-level rating model $\mathbf{B}$ should be not only expressive enough to encode and compress various cluster-level user-item rating patterns but also compact enough to avoid over-fitting. In the empirical tests, we observed that the performance is rather stable when $K$ and $L$ are in the range of $[20, 50]$. Thus, we simply set $K = 20$ and $L = 20$ in our experiments.

## 5. Related Works

The proposed cross-domain collaborative filtering belongs to multi-task learning. The earliest studies on multi-task learning should be (Caruana, 1997; Baxter, 2000), which learn multiple tasks by sharing a hidden layer in neural network. In our proposed RMGM method, each given rating matrix in the related domains can be generated by drawing a set of users and items as well as the corresponding ratings from the RMGM. In other words, each user/item in the given rating matrix is a linear combination of the prototypes for user/item clusters (see Eq. (19)). The shared cluster-level rating model $\mathbf{B}$ is a *two-sided feature representation* for both users and items. This knowledge sharing fashion is similar to the feature-representation based multi-task/transfer learning, such as (Jebara, 2004; Argyriou et al., 2007; Raina et al., 2007). They intend to find a common feature representation (usually a low-dimensional subspace) that is beneficial for the related tasks. A major difference from our work is that these methods learn a one-sided feature representation (in row space) while our method learns a two-sided feature representation (in both row and column spaces). Owing to such two-sided feature representation, RMGM can share the knowledge across multiple tabular data sets from different domains.

Since RMGM is a mixture model, our method is also related to various model-based CF methods. The most similar one is the flexible mixture model (FMM) (Si & Jin, 2003) which simultaneously models users and items into mixture models in terms of two latent cluster variables. However, as pointed out in Section 4, our RMGM is different from FMM in both training and prediction algorithms; moreover, the major difference is that RMGM is able to generate rating matrices in different domains. Several methods also aim at simultaneously clustering users and items for modeling rating patterns, such as the two-sided clustering model (Hofmann & Puzicha, 1999) and the co-clustering-based model (George & Merugu, 2005).

## 6. Experiments

In this section, we investigate whether the CF performance can be improved by applying RMGM to extracting the shared knowledge from multiple rating matrices in related domains. We compare our RMGM-based cross-domain collaborative filtering method to two baseline single-task methods. One is the well known memory-based method, Pearson correlation coefficients (PCC) (Resnick et al., 1994), and we search 20-nearest neighbors in our experiments. The other is the flexible mixture model (FMM) (Si & Jin, 2003), which can be viewed as a single-task version of RMGM. Since (Si & Jin, 2003) claims that FMM performs better than some well-known state-of-the-art model-based methods, we only compare our method to FMM. We aim to validate that sharing useful information by learning a common rating model for multiple related CF tasks can obtain better performance than learning individual models for these tasks separately.

### 6.1. Data Sets

The following three real-world CF data sets are used for performance evaluation. Our method will learn a shared model (RMGM) on the union of the rating data from these data sets, and the learned model is applicable for either task.

**MovieLens**[1]: A movie rating data set comprising 100,000 ratings (scales $1 - 5$) provided by 943 users on 1682 movies. We randomly select 500 users with more than 20 ratings and 1000 movies for experiments (rating ratio 4.33%).

[1]http://www.grouplens.org/node/73

**EachMovie**[2]: A movie rating data set comprising 2.8 million ratings (scales $1-6$) provided by 72,916 users on 1628 movies. We randomly select 500 users with more than 20 ratings and 1000 movies for experiments (rating ratio 3.28%). For a rating scale consistency with other tasks, we replace 6 with 5 in the rating matrix to make the rating scales from 1 to 5.

**Book-Crossing**[3]: A book rating data set comprising more than 1.1 million ratings (scales $1-10$) provided by 278,858 users on 271,379 books. We randomly select 500 users and 1000 books with more than 16 ratings for experiments (rating ratio 2.78%). We also normalize the rating scales from 1 to 5.

## 6.2. Evaluation Protocol

We evaluate the performance of the compared methods under different configurations. The first 100, 200, and 300 users in the three rating matrices (each data set forms a $500 \times 1000$ rating matrix) are used for training, respectively, and the last 200 users for testing. For each test user, three different sizes of the observed ratings (Given5, Given10, Given15) are provided for training and the remaining ratings are used for evaluation. Note that in our experiments, the given observed rating indices are randomly selected 10 times, so that the reported results in Table 1 are the average results over 10 splits.

The evaluation metric we adopt is mean absolute error (MAE): $(\sum_{i \in T} |r_i - \tilde{r}_i|)/|T|$, where $T$ denotes the set of test ratings, $r_i$ is the ground truth and $\tilde{r}_i$ is the predicted rating. A smaller value of MAE means a better performance.

## 6.3. Results

The comparison results on the three data sets are reported in Table 1. One can see that our method clearly outperforms the two baseline methods under all the testing configurations on all the three data sets. FMM performs slightly better than PCC, which implies that the model-based methods can benefit from sharing knowledge within user and item clusters. RMGM performs even better than FMM, which implies that clustering users and items across multiple related tasks can aggregate even more useful knowledge than clustering users and items in individual tasks. The overall experimental results have validated that the proposed RMGM indeed can gain additional useful knowledge by pooling the rating data from multiple related CF tasks to make these tasks benefit from one another.

[2]http://www.cs.cmu.edu/~lebanon/IR-lab.htm

[3]http://www.informatik.uni-freiburg.de/~cziegler/BX/

*Table 1.* MAE Comparison on MovieLens (ML), Each-Movie (EM), and Book-Crossing (BX).

| Train | Method | Given5 | Given10 | Given15 |
|---|---|---|---|---|
| ML100 | PCC | 0.930 | 0.908 | 0.895 |
| | FMM | 0.908 | 0.868 | 0.846 |
| | RMGM | *0.868* | *0.822* | *0.808* |
| ML200 | PCC | 0.934 | 0.899 | 0.888 |
| | FMM | 0.890 | 0.863 | 0.847 |
| | RMGM | *0.859* | *0.821* | *0.806* |
| ML300 | PCC | 0.935 | 0.896 | 0.888 |
| | FMM | 0.885 | 0.868 | 0.846 |
| | RMGM | *0.857* | *0.820* | *0.804* |
| EM100 | PCC | 0.996 | 0.952 | 0.936 |
| | FMM | 0.969 | 0.937 | 0.924 |
| | RMGM | *0.942* | *0.908* | *0.895* |
| EM200 | PCC | 0.983 | 0.943 | 0.930 |
| | FMM | 0.955 | 0.933 | 0.923 |
| | RMGM | *0.934* | *0.905* | *0.890* |
| EM300 | PCC | 0.976 | 0.937 | 0.933 |
| | FMM | 0.952 | 0.930 | 0.924 |
| | RMGM | *0.934* | *0.906* | *0.890* |
| BX100 | PCC | 0.617 | 0.599 | 0.600 |
| | FMM | 0.619 | 0.592 | 0.583 |
| | RMGM | *0.612* | *0.583* | *0.573* |
| BX200 | PCC | 0.621 | 0.612 | 0.620 |
| | FMM | 0.617 | 0.602 | 0.596 |
| | RMGM | *0.615* | *0.591* | *0.583* |
| BX300 | PCC | 0.621 | 0.619 | 0.630 |
| | FMM | 0.615 | 0.604 | 0.596 |
| | RMGM | *0.612* | *0.590* | *0.581* |

## 6.4. Discussion

Although the proposed method can clearly outperform the other compared methods on all the three data sets, we can see that there still exists some room for further performance improvements. A crucial problem lies in the inherent problem of the data sets, i.e., the users and items in the rating matrices may not always be able to be grouped into high quality clusters. We observe that the average ratings of the three data sets are far larger than the medians (given the median being 3, the average ratings are 3.64, 3.95, and 4.22 for the three data sets, respectively). This may be caused by the fact that the items with the most ratings are usually the most popular ones. In other words, users are willing to rate their favorite items and to recommend them to others, but have little interest to rate the items they dislike. Given that no clear user and item groups can be discovered for these cases, it is hard to learn a good cluster-level rating model.

## 7. Conclusion

In this paper, we proposed a novel cross-domain collaborative filtering method based on the rating-matrix generative model (RMGM) for recommender systems. RMGM can share useful knowledge across multiple rating matrices in related domains to alleviate the sparsity problems in individual tasks. The knowledge is shared in the form of a latent cluster-level rating model, which is trained on the pooled rating data from multiple related rating matrices. Each rating matrix can thus be viewed as drawing a set of users and items from the user-item joint mixture model as well as drawing the corresponding ratings from the cluster-level rating model. The experimental results have validated that the proposed RMGM indeed can gain additional useful knowledge by pooling the rating data from multiple related tasks to make these tasks benefit from one another.

In our future work, we will 1) investigate how to statistically quantify the "relatedness" between rating matrices in different domains, and 2) consider an asymmetric problem setting where knowledge can be transferred from a dense auxiliary rating matrix in one domain to a sparse target one in another domain.

## Acknowledgments

Bin Li and Qiang Yang are supported by Hong Kong CERG Grant 621307; Bin Li and Xiangyang Xue are supported in part by Shanghai Leading Academic Discipline Project (No. B114) and NSF of China (No. 60873178).

## References

Argyriou, A., Evgeniou, T., & Pontil, M. (2007). Multi-task feature learning. *Advances in Neural Information Processing Systems 19* (pp. 41–48).

Baxter, J. (2000). A model of inductive bias learning. *J. of Artificial Intelligence Research*, *12*, 149–198.

Caruana, R. A. (1997). Multitask learning. *Machine Learning*, *28*, 41–75.

Coyle, M., & Smyth, B. (2008). Web search shared: Social aspects of a collaborative, community-based search network. *Proc. of the Fifth Int'l Conf. on Adaptive Hypermedia and Adaptive Web-Based Systems* (pp. 103–112).

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

Ding, C., Li, T., Peng, W., & Park, H. (2006). Orthogonal nonnegative matrix tri-factorizations for clustering. *Proc. of the 12th ACM SIGKDD Int'l Conf.* (pp. 126–135).

George, T., & Merugu, S. (2005). A scalable collaborative filtering framework based on co-clustering. *Proc. of the Fifth IEEE Int'l Conf. on Data Mining* (pp. 625–628).

Hofmann, T., & Puzicha, J. (1998). *Statistical models for co-occurrence data* (Technical Report AIM-1625). Artifical Intelligence Laboratory, MIT.

Hofmann, T., & Puzicha, J. (1999). Latent class models for collaborative filtering. *Proc. of the 16th Int'l Joint Conf. on Artificial Intelligence* (pp. 688–693).

Jebara, T. (2004). Multi-task feature and kernel selection for SVMs. *Proc. of the 21st Int'l Conf. on Machine Learning* (pp. 329–336).

Pennock, D. M., Horvitz, E., Lawrence, S., & Giles, C. L. (2000). Collaborative filtering by personality diagnosis: A hybrid memory- and model-based approach. *Proc. of the 16th Conf. on Uncertainty in Artificial Intelligence* (pp. 473–480).

Raina, R., Battle, A., Lee, H., Packer, B., & Ng, A. Y. (2007). Self-taught learning: Transfer learning from unlabeled data. *Proc. of the Int'l Conf. on Machine Learning* (pp. 759–766).

Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., & Riedl, J. (1994). GroupLens: An open architecture for collaborative filtering of netnews. *Proc. of the ACM Conf. on Computer Supported Cooperative Work* (pp. 175–186).

Sarwar, B., Karypis, G., Konstan, J., & Riedl, J. (2001). Item-based collaborative filtering recommendation algorithms. *Proc. of the 10th Int'l World Wide Web Conf.* (pp. 285–295).

Si, L., & Jin, R. (2003). Flexible mixture model for collaborative filtering. *Proc. of the 20th Int'l Conf. on Machine Learning* (pp. 704–711).

Srebro, N., & Jaakkola, T. (2003). Weighted low-rank approximations. *Proc. of the 20th Int'l Conf. on Machine Learning* (pp. 720–727).