



Transfer learning in heterogeneous collaborative filtering domains



Weike Pan, Qiang Yang*

Department of Computer Science and Engineering, Hong Kong University of Science and Technology, Clearwater Bay, Kowloon, Hong Kong

ARTICLE INFO

Article history:

Received 6 December 2010

Received in revised form 6 December 2012

Accepted 12 January 2013

Available online 11 February 2013

Keywords:

Transfer learning

Collaborative filtering

Missing ratings

ABSTRACT

A major challenge for collaborative filtering (CF) techniques in recommender systems is the data sparsity that is caused by missing and noisy ratings. This problem is even more serious for CF domains where the ratings are expressed numerically, e.g. as 5-star grades. We assume the 5-star ratings are unordered bins instead of ordinal relative preferences. We observe that, while we may lack the information in numerical ratings, we sometimes have additional auxiliary data in the form of binary ratings. This is especially true given that users can easily express themselves with their preferences expressed as likes or dislikes for items. In this paper, we explore how to use these binary auxiliary preference data to help reduce the impact of data sparsity for CF domains expressed in numerical ratings. We solve this problem by transferring the rating knowledge from some auxiliary data source in binary form (that is, likes or dislikes), to a target numerical rating matrix.

In particular, our solution is to model both the numerical ratings and ratings expressed as like or dislike in a principled way. We present a novel framework of *Transfer by Collective Factorization* (TCF), in which we construct a shared latent space collectively and learn the data-dependent effect separately. A major advantage of the TCF approach over the previous bilinear method of collective matrix factorization is that we are able to capture the data-dependent effect when sharing the data-independent knowledge. This allows us to increase the overall quality of knowledge transfer. We present extensive experimental results to demonstrate the effectiveness of TCF at various sparsity levels, and show improvements of our approach as compared to several state-of-the-art methods.

© 2013 Elsevier B.V. All rights reserved.

1. Introduction

Data sparsity is a major challenge in collaborative filtering [23,9,43]. Sparsity refers to the fact that some observed ratings, e.g. 5-star grades, in a *user-item* rating matrix are too few, such that overfitting can easily happen when we use a prediction model for missing values in the test data. However, we observe that, some auxiliary data of the form “like” or “dislike” may be more easily obtained, such as the favored/disfavored data in Moviepilot¹ and QiYi,² the dig/bury data in Tudou,³ the love/ban data in Last.fm,⁴ and the “Want to see”/“Not interested” data in Flixster.⁵ It is often more convenient for users to express such preferences instead of numerical ratings. The question we ask in this paper is: how do we take

* Corresponding author.

E-mail addresses: weikep@cse.ust.hk (W. Pan), qyang@cse.ust.hk (Q. Yang).

¹ <http://www.moviepilot.de>.

² <http://www.qiyi.com>.

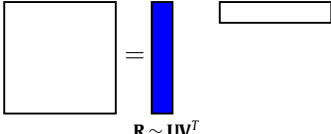
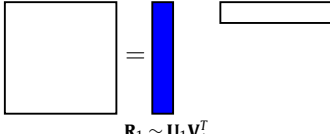
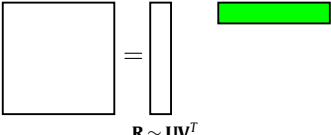
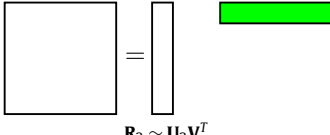
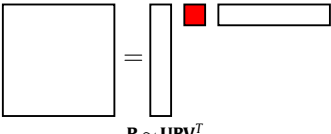
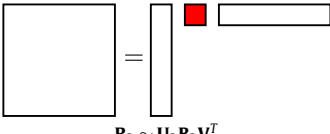
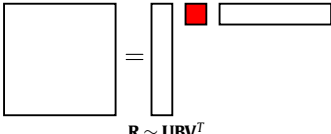
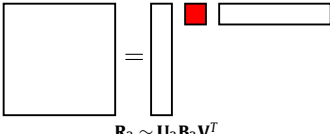
³ <http://www.tudou.com>.

⁴ <http://www.last.fm>.

⁵ <http://www.flixster.com>.

Table 1

Matrix illustration of some related work on transfer learning in collaborative filtering. Note that SoRec, CMF, CBT and RMGM can be applied in more general problem settings, e.g. more than two matrices, more than one types of alignments, etc.

Methods	Training data	Auxiliary data
SoRec (user side) [39]	 $\mathbf{R} \sim \mathbf{U}\mathbf{V}^T$ Knowledge sharing: $\mathbf{U} = \mathbf{U}_1$ Value domain: $(\mathbf{U}, \mathbf{V}), (\mathbf{U}_1, \mathbf{V}_1) \in D_{\mathbb{R}}$ $D_{\mathbb{R}} = \{(\mathbf{U}, \mathbf{V}) \mid \mathbf{U} \in \mathbb{R}^{n \times d}, \mathbf{V} \in \mathbb{R}^{m \times d}\}$	 $\mathbf{R}_1 \sim \mathbf{U}_1 \mathbf{V}_1^T$
CMF (item side) [52]	 $\mathbf{R} \sim \mathbf{U}\mathbf{V}^T$ Knowledge sharing: $\mathbf{V} = \mathbf{V}_2$ Value domain: $(\mathbf{U}, \mathbf{V}), (\mathbf{U}_2, \mathbf{V}_2) \in D_{\mathbb{R}}$ $D_{\mathbb{R}} = \{(\mathbf{U}, \mathbf{V}) \mid \mathbf{U} \in \mathbb{R}^{n \times d}, \mathbf{V} \in \mathbb{R}^{m \times d}\}$	 $\mathbf{R}_2 \sim \mathbf{U}_2 \mathbf{V}_2^T$
CBT (not aligned) [31]	 $\mathbf{R} \sim \mathbf{U}\mathbf{B}\mathbf{V}^T$ Knowledge sharing: $\mathbf{B} = \mathbf{B}_3$ Value domain: $(\mathbf{U}, \mathbf{V}), (\mathbf{U}_3, \mathbf{V}_3) \in D_{\{0,1\}}$ $D_{\{0,1\}} = \{(\mathbf{U}, \mathbf{V}) \mid \mathbf{U} \in \{0, 1\}^{n \times d}, \mathbf{U}_1 = \mathbf{1}, \mathbf{V} \in \{0, 1\}^{m \times d}, \mathbf{V}_1 = \mathbf{1}\}$	 $\mathbf{R}_3 \sim \mathbf{U}_3 \mathbf{B}_3 \mathbf{V}_3^T$
RMGM (not aligned) [32]	 $\mathbf{R} \sim \mathbf{U}\mathbf{B}\mathbf{V}^T$ Knowledge sharing: $\mathbf{B} = \mathbf{B}_3$ Value domain: $(\mathbf{U}, \mathbf{V}), (\mathbf{U}_3, \mathbf{V}_3) \in D_{\{0,1\}}$ $D_{\{0,1\}} = \{(\mathbf{U}, \mathbf{V}) \mid \mathbf{U} \in \{0, 1\}^{n \times d}, \mathbf{U}_1 = \mathbf{1}, \mathbf{V} \in \{0, 1\}^{m \times d}, \mathbf{V}_1 = \mathbf{1}\}$	 $\mathbf{R}_3 \sim \mathbf{U}_3 \mathbf{B}_3 \mathbf{V}_3^T$

advantage of auxiliary knowledge in the form of binary ratings to alleviate the sparsity problem in numerical ratings when we build a rating-prediction model?

To the best of our knowledge, no previous work answered the question of how to jointly model a target data of numerical ratings and an auxiliary data of binary ratings. There are some prior works on using both the numerical ratings and *implicit* data of “whether rated” [28,35] or “whether purchased” [57] to help boost the prediction performance. Among the previous works, Koren [28] uses implicit data of “rated” as offsets in a factorization model, and Liu et al. [35] adapt the collective matrix factorization (CMF) approach [52] to integrate the implicit data of “rated.” Zhang and Nie [57] convert the implicit data of simulated purchases to a *user-brand* matrix as a user-side meta data representing brand loyalty and a *user-item* matrix of “purchased.” However, none of these previous works consider how to use auxiliary data in the form of like and dislike type of binary ratings in collaborative filtering in a transfer learning framework.

Most existing transfer learning methods in collaborative filtering consider auxiliary data from several perspectives, including user-side transfer [39,11,58,38,55], item-side transfer [52], or knowledge-transfer using related but not aligned data [31, 32]. We illustrate the ideas of knowledge sharing from a matrix factorization view as shown in Table 1. We show four representative methods [39,52,31,32] in Table 1 and describe the details starting from a non-transfer learning method of probabilistic matrix factorization (PMF) [47].

Probabilistic matrix factorization The PMF [47] or latent factorization model (LFM) [4] seeks an appropriate low-rank approximation, $\mathbf{R} = \mathbf{U}\mathbf{V}^T$, for which any missing value can be predicted by $\hat{r}_{ui} = \mathbf{U}_u \cdot \mathbf{V}_i^T$, where $\mathbf{U} \in \mathbb{R}^{n \times d}$, $\mathbf{V} \in \mathbb{R}^{m \times d}$ are user-specific and item-specific latent feature matrices, respectively. The optimization problem of PMF is as follows [47,4],

$$\min_{\mathbf{U}, \mathbf{V}} \mathcal{E}_1(\mathbf{U}, \mathbf{V}) + \alpha \mathcal{R}(\mathbf{U}, \mathbf{V}) \quad (1)$$

where $\mathcal{E}_1(\mathbf{U}, \mathbf{V}) = \frac{1}{2} \sum_{u=1}^n \sum_{i=1}^m y_{ui} (r_{ui} - U_u \cdot V_i^T)^2 = \frac{1}{2} \|\mathbf{Y} \odot (\mathbf{R} - \mathbf{UV}^T)\|_F^2$ is the loss function, and $\mathcal{R}(\mathbf{U}, \mathbf{V}) = \frac{1}{2} (\sum_{u=1}^n \|U_u\|^2 + \sum_{i=1}^m \|V_i\|^2) = \frac{1}{2} (\|\mathbf{U}\|_F^2 + \|\mathbf{V}\|_F^2)$ is a regularization term used to avoid overfitting.

Social recommendation SoRec [39] is proposed to alternatively factorize the target rating matrix \mathbf{R} and a user-side social network matrix \mathbf{R}_1 with the constraint of sharing the same user-specific latent feature matrix (see $\mathbf{U} = \mathbf{U}_1$ in Table 1). The objective function is formalized as follows [39],

$$\min_{\mathbf{U}, \mathbf{V}_1, \mathbf{U}} \mathcal{E}_1(\mathbf{U}, \mathbf{V}) + \mathcal{E}_1(\mathbf{U}, \mathbf{V}_1) + \alpha \mathcal{R}(\mathbf{U}, \mathbf{V}, \mathbf{V}_1) \quad (2)$$

where $(\mathbf{U}, \mathbf{V}) \in D_{\mathbb{R}}$, and $\mathcal{R}(\mathbf{U}, \mathbf{V}, \mathbf{V}_1) = \frac{1}{2} (\|\mathbf{U}\|_F^2 + \|\mathbf{V}\|_F^2 + \|\mathbf{V}_1\|_F^2)$ is a regularization term on the latent variables.

Collective matrix factorization CMF [52] is proposed to alternatively factorize the target rating matrix \mathbf{R} and an item-side content matrix \mathbf{R}_2 with the constraint of sharing the same item-specific latent feature matrix (see $\mathbf{V} = \mathbf{V}_2$ in Table 1). This approach is similar to that in SoRec [39], but with different auxiliary data. The optimization problem of CMF is stated as follows [52],

$$\min_{\mathbf{U}, \mathbf{V}, \mathbf{U}_2} \mathcal{E}_1(\mathbf{U}, \mathbf{V}) + \mathcal{E}_1(\mathbf{U}_2, \mathbf{V}) + \alpha \mathcal{R}(\mathbf{U}, \mathbf{V}, \mathbf{U}_2) \quad (3)$$

where $(\mathbf{U}, \mathbf{V}) \in D_{\mathbb{R}}$, and $\mathcal{R}(\mathbf{U}, \mathbf{V}, \mathbf{U}_2) = \frac{1}{2} (\|\mathbf{U}\|_F^2 + \|\mathbf{V}\|_F^2 + \|\mathbf{U}_2\|_F^2)$ is again a regularization term used to avoid overfitting.

Codebook transfer The CBT [31] method consists of codebook construction and expansion steps. It achieves knowledge transfer with the assumption that both auxiliary and target data share a common cluster-level rating pattern (see $\mathbf{B} = \mathbf{B}_3$ in Table 1).

1. Codebook construction. Assume that $(\mathbf{U}_3, \mathbf{V}_3) \in D_{\{0,1\}}$ are user-specific and item-specific membership indicator matrices of the auxiliary rating matrix \mathbf{R}_3 , which are obtained using co-clustering algorithms such as NMF [19]. The constructed codebook is represented as $\mathbf{B}_3 = [\mathbf{U}_3^T \mathbf{R}_3 \mathbf{V}_3] \oslash [\mathbf{U}_3^T (\mathbf{R}_3 > 0) \mathbf{V}_3]$ [31], where $[\mathbf{U}_3^T \mathbf{R}_3 \mathbf{V}_3]_{k\ell}$ denotes the summation of ratings by users in a user cluster k on items in an item cluster ℓ . $[\mathbf{U}_3^T (\mathbf{R}_3 > 0) \mathbf{V}_3]_{k\ell}$ denotes the number of ratings from users in a user cluster k on items in an item cluster ℓ , hence, the element-wise division \oslash resembles the idea of normalization, and $[\mathbf{B}_3]_{k\ell}$ is the average rating of users in a user cluster k on items in an item cluster ℓ .
2. Codebook expansion. The codebook expansion problem is formalized as follows [31],

$$\min_{\mathbf{U}, \mathbf{V}} \mathcal{E}_{\mathbf{B}}(\mathbf{U}, \mathbf{V}) \quad \text{s.t.} \quad (\mathbf{U}, \mathbf{V}) \in D_{\{0,1\}} \quad (4)$$

where $\mathcal{E}_{\mathbf{B}}(\mathbf{U}, \mathbf{V}) = \frac{1}{2} \|\mathbf{Y} \odot (\mathbf{R} - \mathbf{UBV}^T)\|_F^2$ is a \mathbf{B} -regularized square loss function, and $\mathbf{B} = \mathbf{B}_3$ is the codebook constructed from the auxiliary data \mathbf{R}_3 . In [31], an alternating greedy-search algorithm is proposed to solve the combinatorial optimization problem in Eq. (4), and the choices of $U_{uk} = 1$, $V_{i\ell} = 1$ are used to select the entry located at (k, ℓ) of \mathbf{B} via $[\mathbf{UBV}^T]_{ui} = U_u \cdot \mathbf{B} V_i^T$. Thus, the predicted rating $\hat{r}_{ui} = [\mathbf{UBV}^T]_{ui} = [\mathbf{B}]_{k\ell}$ is the average rating of users in the user cluster k on items in an item cluster ℓ of the auxiliary data.

Rating-matrix generative model RMGM [32] is derived and extended from the FMM generative model [50], and we rewrite it in a matrix factorization manner,

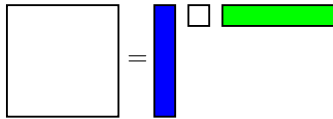
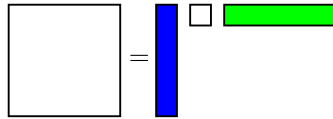
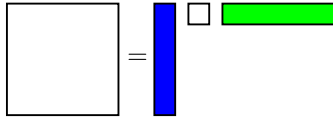
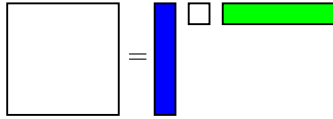
$$\min_{\mathbf{U}, \mathbf{V}, \mathbf{B}, \mathbf{U}_3, \mathbf{V}_3} \mathcal{E}_{\mathbf{B}}(\mathbf{U}, \mathbf{V}) + \mathcal{E}_{\mathbf{B}}(\mathbf{U}_3, \mathbf{V}_3) \quad \text{s.t.} \quad (\mathbf{U}, \mathbf{V}), (\mathbf{U}_3, \mathbf{V}_3) \in D_{\{0,1\}} \quad (5)$$

where $\mathcal{E}_{\mathbf{B}}(\mathbf{U}, \mathbf{V})$ is again a \mathbf{B} -regularized loss function, the same as given in Eq. (4). We can see that RMGM is different from CBT since it learns (\mathbf{U}, \mathbf{V}) and $(\mathbf{U}_3, \mathbf{V}_3)$ alternatively and relaxes the hard membership requirement as imposed by the indicator matrix, e.g. $\mathbf{U} \in \{0, 1\}^{n \times d}$. A soft indicator matrix is used in RMGM [32], e.g., $\mathbf{U} \in [0, 1]^{n \times d}$.

In this paper, we consider the situation where the auxiliary data is such that the following information are aligned: users and items of the target rating matrix and the auxiliary binary rating matrix. This assumption gives us precise information on the mapping between auxiliary and target data, which can lead to higher performance than not having this knowledge. We illustrate the idea of these assumptions using matrices in Table 2, where we can see that our problem setting and proposed solution are both novel and different from the previous ones as shown in Table 1. We will discuss these novelty in the sequel.

Our idea extends the ideas in our previous conference papers on this topic [43,42], on which we have extensively extended. Compared to these preliminary works, we have extended in the following aspects. First, we have included a new analysis of transfer learning methods in collaborative filtering from the perspective of matrix factorization in Section 1. Second, we have provided more detailed derivations of our equations in Section 3. Third, we have included more experimental results that are reported in Section 4. Finally, we have added more related works and associated discussions as given in Section 5.

Table 2
Matrix illustration of Transfer by Collective Factorization.

Variants of TCF	Training data	Auxiliary data
CMTF (frontal side)	 $\mathbf{R} \sim \mathbf{U}\mathbf{B}\mathbf{V}^T$	 $\tilde{\mathbf{R}} \sim \tilde{\mathbf{U}}\mathbf{B}\tilde{\mathbf{V}}^T$
	Knowledge sharing: $\mathbf{U} = \tilde{\mathbf{U}}, \mathbf{V} = \tilde{\mathbf{V}}$ Value domain: $(\mathbf{U}, \mathbf{V}), (\tilde{\mathbf{U}}, \tilde{\mathbf{V}}) \in D_{\mathbb{R}}$ $D_{\mathbb{R}} = \{(\mathbf{U}, \mathbf{V}) \mid \mathbf{U} \in \mathbb{R}^{n \times d}, \mathbf{V} \in \mathbb{R}^{m \times d}\}$	
CSVD (frontal side)	 $\mathbf{R} \sim \mathbf{U}\mathbf{B}\mathbf{V}^T$	 $\tilde{\mathbf{R}} \sim \tilde{\mathbf{U}}\mathbf{B}\tilde{\mathbf{V}}^T$
	Knowledge sharing: $\mathbf{U} = \tilde{\mathbf{U}}, \mathbf{V} = \tilde{\mathbf{V}}$ Value domain: $(\mathbf{U}, \mathbf{V}), (\tilde{\mathbf{U}}, \tilde{\mathbf{V}}) \in D_{\perp}$ $D_{\perp} = \{(\mathbf{U}, \mathbf{V}) \mid \mathbf{U} \in \mathbb{R}^{n \times d}, \mathbf{U}^T \mathbf{U} = \mathbf{I}, \mathbf{V} \in \mathbb{R}^{m \times d}, \mathbf{V}^T \mathbf{V} = \mathbf{I}\}$	

The organization of the paper is as follows. We give a formal definition of the problem in Section 2 and then describe our solution in detail in Section 3. We present experimental results on real-world data sets in Section 4, and discuss about some related work in Section 5. Finally, we give some concluding remarks and future works in Section 6.

2. Heterogeneous collaborative filtering problems

2.1. Problem definition

In the target data, we have a *user-item* numerical rating matrix $\mathbf{R} = [r_{ui}]_{n \times m} \in \{1, 2, 3, 4, 5, ?\}^{n \times m}$ with q observed ratings, where the question mark “?” denotes a missing value, which can be an unobserved value. Note that the observed rating values in \mathbf{R} are considered as unordered bins and are not limited to 5-star grades; instead, they can be any real numbers. We use an indicator matrix $\mathbf{Y} = [y_{ui}]_{n \times m} \in \{0, 1\}^{n \times m}$ to denote whether the entry (u, i) is observed ($y_{ui} = 1$) or not ($y_{ui} = 0$), and $\sum_{u,i} y_{ui} = q$. Similarly, in the auxiliary data, we have a *user-item* binary rating matrix $\tilde{\mathbf{R}} = [\tilde{r}_{ui}]_{n \times m} \in \{0, 1, ?\}^{n \times m}$ with \tilde{q} observations, where a value of one denotes the observed ‘like’ value, and zero denotes the observed ‘dislike’ value. The question mark denotes the missing value. Similar to the target data, we have a corresponding indicator matrix $\tilde{\mathbf{Y}} = [\tilde{y}_{ui}]_{n \times m} \in \{0, 1\}^{n \times m}$, and $\sum_{u,i} \tilde{y}_{ui} = \tilde{q}$. Note that there is a one-one mapping between the users and items of \mathbf{R} and $\tilde{\mathbf{R}}$. Our goal is to predict the missing values in \mathbf{R} by transferring the rating knowledge from $\tilde{\mathbf{R}}$. Note that the binary ratings here are different from the implicit data used in [28,35,57], which can be represented as $\{1, ?\}^{n \times m}$, since implicit data corresponds to positive observations only.

2.2. Challenges

Our problem setting is novel and challenging. In particular, we enumerate the following challenges for the problem setting (see Fig. 1).

1. *How to make use of the existing correspondences* among users and items from two domains, given that such relationships are important and can serve as a bridge across two domains. Some previous solutions were proposed without such correspondences [31,32], and are thus imprecise. Other works have used correspondence information as additional constraints on the user-specific or item-specific latent feature matrices [39,52].
2. *What to transfer and how to transfer*, as raised in [41]. Previous works that address this question include approaches that transfer the knowledge of latent features in an adaptive way [43] or in a collective way [39,52]. Some works in this direction include those that transfer cluster-level rating patterns [31] in an adaptive manner or in a collective manner [32].
3. *How to model the data-dependent effect* of numerical ratings and binary ratings when sharing the data-independent knowledge? This question is important since clearly the auxiliary and target data may have different distributions and quite different semantic meanings.

From Table 1, we can see that the solutions of [39,52,31,32] were proposed for different problem settings as compared to ours, as shown in Table 2 and Fig. 1. More specifically, for the aforementioned three challenges from our problem setting,

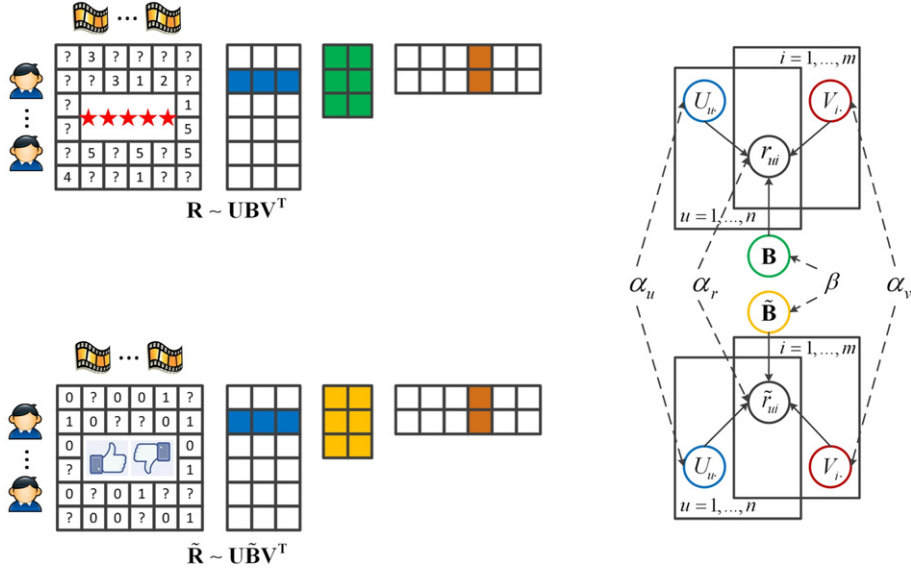


Fig. 1. Graphical model of *Transfer by Collective Factorization* for transfer learning in collaborative filtering. Note that we use the same set of user-specific latent feature vectors and the same set of item-specific latent feature vectors for both target data and auxiliary data.

the approaches of [39,52] cannot capture the data-dependent information, and the methods of [31,32] cannot make use of the existing correspondence information.

2.3. Overview of our solution

We propose a principled matrix-based transfer-learning framework referred to as *Transfer by Collective Factorization*, which jointly factorizes the data matrices in three parts: a user-specific latent feature matrix, an item-specific latent feature matrix, and two data-dependent inner matrices. Specifically, the main idea of our solution has two major steps. First, we factorize both the target numerical rating matrix, $\mathbf{R} \sim \mathbf{UBV}^T$, and the auxiliary binary rating matrix, $\tilde{\mathbf{R}} \sim \tilde{\mathbf{U}}\tilde{\mathbf{B}}\tilde{\mathbf{V}}^T$, with constraints of sharing user-specific latent feature matrix $\mathbf{U} = \tilde{\mathbf{U}}$ and item-specific latent feature matrix $\mathbf{V} = \tilde{\mathbf{V}}$ (see Table 2 for matrix illustration). Second, we learn the inner matrices \mathbf{B} and $\tilde{\mathbf{B}}$ separately in each domain to capture the domain-dependent information, since the semantic meaning and distributions of numerical ratings and binary ratings may be different. As an alternative interpretation, the inner matrices \mathbf{B} and $\tilde{\mathbf{B}}$ can be considered as data-dependent correlations between the rows of \mathbf{U} and columns of \mathbf{V}^T for target data and auxiliary data, respectively. Those two major steps are iterated to have richer interactions for knowledge sharing [13,54] until we reach convergence to a locally optimal state. The intuition of our approach is that same users and items in two domains are likely to have the same latent feature matrices, e.g. $\mathbf{U} = \tilde{\mathbf{U}}$ and $\mathbf{V} = \tilde{\mathbf{V}}$, while the domain differences, the data-dependent information, are left for the inner matrices, \mathbf{B} and $\tilde{\mathbf{B}}$.

In summary, our major contributions are:

1. We make full use of the correspondences among users and items, from a source and a target domains. We allow the aligned users and items to share the same user-specific latent feature matrix and item-specific latent feature matrix, respectively.
2. We construct a shared latent space to address the what to transfer question, via a matrix tri-factorization, or trilinear, method in a collective way to address the how to transfer question.
3. We model the data-dependent effect of binary ratings and numerical ratings by learning the inner matrices of trilinear method separately.

3. Transfer by collective factorization

3.1. Model formulation

We assume that a user u 's rating on an item i in the target data, r_{ui} , is generated from the user-specific latent feature vector $U_u \in \mathbb{R}^{1 \times d_u}$, item-specific latent feature vector $V_i \in \mathbb{R}^{1 \times d_v}$, and some data-dependent effect denoted as $\mathbf{B} \in \mathbb{R}^{d_u \times d_v}$. Note that this formulation is different from the PMF formulation [47], which only contains U_u and V_i . Similarly, our graphical model as shown in Fig. 1 is a significant extension of the graphical model of PMF [47], where U_u , $u = 1, \dots, n$, and V_i , $i = 1, \dots, m$, are shared to bridge two data, while \mathbf{B} , $\tilde{\mathbf{B}}$ are designed to capture the data-dependent effect. We fix $d = d_u = d_v$ for notation simplicity in the sequel. We define a conditional distribution as

$$p(r_{ui}|U_{u\cdot}, \mathbf{B}, V_{i\cdot}, \alpha_r) = \mathcal{N}(r_{ui}|U_{u\cdot}\mathbf{B}V_{i\cdot}^T, \alpha_r^{-1}),$$

where $\mathcal{N}(x|\mu, \alpha^{-1}) = \sqrt{\frac{\alpha}{2\pi}} \exp\left\{-\frac{\alpha(x-\mu)^2}{2}\right\}$ is the Gaussian distribution with mean μ and precision α . We further define the prior distributions over $U_{u\cdot}$, $V_{i\cdot}$ and \mathbf{B} as $p(U_{u\cdot}|\alpha_u) = \mathcal{N}(U_{u\cdot}|\mathbf{0}, \alpha_u^{-1}\mathbf{I})$, $p(V_{i\cdot}|\alpha_v) = \mathcal{N}(V_{i\cdot}|\mathbf{0}, \alpha_v^{-1}\mathbf{I})$, and $p(\mathbf{B}|\beta) = \mathcal{N}(\mathbf{B}|\mathbf{0}, (\beta/q)^{-1}\mathbf{I})$. We then have the log-posterior function over the latent variables $\mathbf{U} \in \mathbb{R}^{n \times d}$, $\mathbf{B} \in \mathbb{R}^{d \times d}$ and $\mathbf{V} \in \mathbb{R}^{m \times d}$ via Bayesian inference,

$$\begin{aligned} \log p(\mathbf{U}, \mathbf{B}, \mathbf{V}|\mathbf{R}, \alpha_r, \alpha_u, \alpha_v, \beta) \\ &= \log \prod_{u=1}^n \prod_{i=1}^m [p(r_{ui}|U_{u\cdot}, \mathbf{B}, V_{i\cdot}, \alpha_r) p(U_{u\cdot}|\alpha_u) p(V_{i\cdot}|\alpha_v) p(\mathbf{B}|\beta)]^{y_{ui}} \\ &= \log \prod_{u=1}^n \prod_{i=1}^m [\mathcal{N}(r_{ui}|U_{u\cdot}\mathbf{B}V_{i\cdot}^T, \alpha_r^{-1}) \mathcal{N}(U_{u\cdot}|\mathbf{0}, \alpha_u^{-1}\mathbf{I}) \mathcal{N}(V_{i\cdot}|\mathbf{0}, \alpha_v^{-1}\mathbf{I}) \mathcal{N}(\mathbf{B}|\mathbf{0}, (\beta/q)^{-1}\mathbf{I})]^{y_{ui}} \\ &= -\sum_{u=1}^n \sum_{i=1}^m y_{ui} \left[\frac{\alpha_r}{2} (r_{ui} - U_{u\cdot}\mathbf{B}V_{i\cdot}^T)^2 + \frac{\alpha_u}{2} \|U_{u\cdot}\|_F^2 + \frac{\alpha_v}{2} \|V_{i\cdot}\|_F^2 + \frac{\beta}{2q} \|\mathbf{B}\|_F^2 + C \right] \end{aligned}$$

where $C = \ln \sqrt{\frac{\alpha_r}{2\pi}} + \ln \sqrt{\frac{\alpha_u}{2\pi}} + \ln \sqrt{\frac{\alpha_v}{2\pi}} + \ln \sqrt{\frac{\beta}{2q\pi}}$ is a constant. Setting $\alpha_r = 1$, we have

$$-\sum_{u=1}^n \sum_{i=1}^m y_{ui} \left[\frac{1}{2} (r_{ui} - U_{u\cdot}\mathbf{B}V_{i\cdot}^T)^2 + \frac{\alpha_u}{2} \|U_{u\cdot}\|_F^2 + \frac{\alpha_v}{2} \|V_{i\cdot}\|_F^2 \right] - \frac{\beta}{2} \|\mathbf{B}\|_F^2.$$

Similarly, in the auxiliary data, we have a log-posterior function for the matrix tri-factorization, or trilinear, model, $\log p(\mathbf{U}, \tilde{\mathbf{B}}, \mathbf{V}|\tilde{\mathbf{R}}, \alpha_r, \alpha_u, \alpha_v, \beta)$. To jointly maximize these two log-posterior functions, we have

$$\begin{aligned} \max_{\mathbf{U}, \mathbf{V}, \mathbf{B}, \tilde{\mathbf{B}}} \log p(\mathbf{U}, \mathbf{B}, \mathbf{V}|\mathbf{R}, \alpha_r, \alpha_u, \alpha_v, \beta) + \lambda \log p(\mathbf{U}, \tilde{\mathbf{B}}, \mathbf{V}|\tilde{\mathbf{R}}, \alpha_r, \alpha_u, \alpha_v, \beta) \\ \text{s.t. } \mathbf{U}, \mathbf{V} \in \mathcal{D} \end{aligned}$$

where $\lambda > 0$ is a tradeoff parameter to balance the target and auxiliary data and \mathcal{D} is the value domain of the latent variables. \mathcal{D} can be $\mathcal{D}_{\mathbb{R}} = \{\mathbf{U} \in \mathbb{R}^{n \times d}, \mathbf{V} \in \mathbb{R}^{m \times d}\}$ or $\mathcal{D}_{\perp} = \mathcal{D}_{\mathbb{R}} \cap \{\mathbf{U}^T \mathbf{U} = \mathbf{I}, \mathbf{V}^T \mathbf{V} = \mathbf{I}\}$ to get the effect of finding latent topics [18,43] and noise reduction [6,27] in SVD. Thus we have two variants of TCF, CMTF (collective matrix tri-factorization) for $\mathcal{D}_{\mathbb{R}}$ and CSVD (collective SVD) for \mathcal{D}_{\perp} . Although 2DSVD or Tucker2 [20] can factorize a sequence of full matrices, it does not achieve the goal of missing-value prediction in sparse observation matrices, which is accomplished in our proposed approach.

Finally, we obtain the following equivalent minimization problem for TCF,

$$\begin{aligned} \min_{\mathbf{U}, \mathbf{V}, \mathbf{B}, \tilde{\mathbf{B}}} \sum_{u=1}^n \sum_{i=1}^m y_{ui} \left[\frac{1}{2} (r_{ui} - U_{u\cdot}\mathbf{B}V_{i\cdot}^T)^2 + \frac{\alpha_u}{2} \|U_{u\cdot}\|^2 + \frac{\alpha_v}{2} \|V_{i\cdot}\|^2 \right] \\ + \lambda \sum_{u=1}^n \sum_{i=1}^m \tilde{y}_{ui} \left[\frac{1}{2} (\tilde{r}_{ui} - U_{u\cdot}\tilde{\mathbf{B}}V_{i\cdot}^T)^2 + \frac{\alpha_u}{2} \|U_{u\cdot}\|^2 + \frac{\alpha_v}{2} \|V_{i\cdot}\|^2 \right] \\ + \frac{\beta}{2} \|\mathbf{B}\|_F^2 + \lambda \frac{\beta}{2} \|\tilde{\mathbf{B}}\|_F^2 \\ \text{s.t. } \mathbf{U}, \mathbf{V} \in \mathcal{D}. \end{aligned} \quad (6)$$

To solve the optimization problem in Eq. (6), we first collectively factorize two data matrices of \mathbf{R} and $\tilde{\mathbf{R}}$ to learn \mathbf{U} and \mathbf{V} . We then estimate \mathbf{B} and $\tilde{\mathbf{B}}$ separately. We transfer the knowledge of latent feature matrices, \mathbf{U} and \mathbf{V} via collective factorization of the rating matrices \mathbf{R} and $\tilde{\mathbf{R}}$. For this reason, we call our approach *Transfer by Collective Factorization*.

3.2. Learning the TCF

Learning \mathbf{U} and \mathbf{V} in CMTF Given \mathbf{B} and \mathbf{V} , we show that the user-specific latent feature matrix \mathbf{U} in Eq. (6) can be obtained analytically.

Theorem 1. Given \mathbf{B} and \mathbf{V} , we can obtain the user-specific latent feature matrix \mathbf{U} in a closed form.

Proof. Let $f_u = \sum_{i=1}^m y_{ui} [\frac{1}{2}(r_{ui} - U_u \mathbf{B} V_i^T)^2 + \frac{\alpha_u}{2} \|U_u\|^2 + \frac{\alpha_v}{2} \|V_i\|^2] + \frac{\beta}{2} \|\mathbf{B}\|_F^2 + \lambda \{\sum_{i=1}^m \tilde{y}_{ui} [\frac{1}{2}(\tilde{r}_{ui} - U_u \tilde{\mathbf{B}} V_i^T)^2 + \frac{\alpha_u}{2} \|U_u\|^2 + \frac{\alpha_v}{2} \|V_i\|^2] + \frac{\beta}{2} \|\tilde{\mathbf{B}}\|_F^2\}$, and we have

$$\begin{aligned} \frac{\partial f_u}{\partial U_u} &= \sum_{i=1}^m y_{ui} [(-r_{ui} + U_u \mathbf{B} V_i^T) V_i \mathbf{B}^T + \alpha_u U_u] \\ &\quad + \lambda \sum_{i=1}^m \tilde{y}_{ui} [(-\tilde{r}_{ui} + U_u \tilde{\mathbf{B}} V_i^T) V_i \tilde{\mathbf{B}}^T + \alpha_u U_u] \\ &= - \sum_{i=1}^m (y_{ui} r_{ui} V_i \mathbf{B}^T + \lambda \tilde{y}_{ui} \tilde{r}_{ui} V_i \tilde{\mathbf{B}}^T) \\ &\quad + \alpha_u U_u \sum_{i=1}^m (y_{ui} + \lambda \tilde{y}_{ui}) + U_u \sum_{i=1}^m (y_{ui} \mathbf{B} V_i^T V_i \mathbf{B}^T + \lambda \tilde{y}_{ui} \tilde{\mathbf{B}} V_i^T V_i \tilde{\mathbf{B}}^T). \end{aligned}$$

Setting $\frac{\partial f_u}{\partial U_u} = \mathbf{0}$, we have the update rule for each U_u ,

$$U_u = \mathbf{b}_u \mathbf{C}_u^{-1}, \quad (7)$$

where $\mathbf{C}_u = \sum_{i=1}^m (y_{ui} \mathbf{B} V_i^T V_i \mathbf{B}^T + \lambda \tilde{y}_{ui} \tilde{\mathbf{B}} V_i^T V_i \tilde{\mathbf{B}}^T) + \alpha_u \sum_{i=1}^m (y_{ui} + \lambda \tilde{y}_{ui}) \mathbf{I}$ and $\mathbf{b}_u = \sum_{i=1}^m (y_{ui} r_{ui} V_i \mathbf{B}^T + \lambda \tilde{y}_{ui} \tilde{r}_{ui} V_i \tilde{\mathbf{B}}^T)$.

We can see that U_u in Eq. (7) is independent of all other users' latent features given \mathbf{B} and \mathbf{V} , thus we can obtain the user-specific latent feature matrix \mathbf{U} analytically. \square

Similarly, given \mathbf{B} and \mathbf{U} , the latent feature vector V_i of each item i can be estimated in a closed form, and thus the whole item-specific latent feature matrix \mathbf{V} can be obtained analytically,

$$V_i = \mathbf{b}_i \mathbf{C}_i^{-1}, \quad (8)$$

where $\mathbf{C}_i = \sum_{u=1}^n (y_{ui} \mathbf{B}^T U_u U_u \mathbf{B} + \lambda \tilde{y}_{ui} \tilde{\mathbf{B}}^T U_u U_u \tilde{\mathbf{B}}) + \alpha_v \sum_{u=1}^n (y_{ui} + \lambda \tilde{y}_{ui}) \mathbf{I}$ and $\mathbf{b}_i = \sum_{u=1}^n (y_{ui} r_{ui} U_u \mathbf{B} + \lambda \tilde{y}_{ui} \tilde{r}_{ui} U_u \tilde{\mathbf{B}})$.

The closed-form update rule in Eq. (7) or Eq. (8) can be considered as a generalization of the alternating least square (ALS) approach in [4]. Note that Bell and Koren [4] consider bilinear model in a single matrix, which is different from our trilinear models of two matrices.

Learning \mathbf{U} and \mathbf{V} in CSVD Since the constraints \mathfrak{D}_\perp have similar effect of regularization, we remove the regularization terms in Eq. (6) and reach a simplified optimization problem,

$$\begin{aligned} \min_{\mathbf{U}, \mathbf{V}} & \frac{1}{2} \|\mathbf{Y} \odot (\mathbf{R} - \mathbf{U} \mathbf{B} \mathbf{V}^T)\|_F^2 + \frac{\lambda}{2} \|\tilde{\mathbf{Y}} \odot (\tilde{\mathbf{R}} - \mathbf{U} \tilde{\mathbf{B}} \mathbf{V}^T)\|_F^2 \\ \text{s.t.} & \quad \mathbf{U}^T \mathbf{U} = \mathbf{I}, \quad \mathbf{V}^T \mathbf{V} = \mathbf{I}. \end{aligned} \quad (9)$$

Let $f = \frac{1}{2} \|\mathbf{Y} \odot (\mathbf{R} - \mathbf{U} \mathbf{B} \mathbf{V}^T)\|_F^2 + \frac{\lambda}{2} \|\tilde{\mathbf{Y}} \odot (\tilde{\mathbf{R}} - \mathbf{U} \tilde{\mathbf{B}} \mathbf{V}^T)\|_F^2$. We have the gradients on \mathbf{U} as follows,

$$\frac{\partial f}{\partial \mathbf{U}} = (\mathbf{Y} \odot (\mathbf{U} \mathbf{B} \mathbf{V}^T - \mathbf{R})) \mathbf{V} \mathbf{B}^T + \lambda (\tilde{\mathbf{Y}} \odot (\mathbf{U} \tilde{\mathbf{B}} \mathbf{V}^T - \tilde{\mathbf{R}})) \mathbf{V} \tilde{\mathbf{B}}^T.$$

Then, the variable \mathbf{U} can be learned via a gradient descent algorithm on the Grassmann manifold [21,10,27],

$$\mathbf{U} \leftarrow \mathbf{U} - \gamma (\mathbf{I} - \mathbf{U} \mathbf{U}^T) \frac{\partial f}{\partial \mathbf{U}} = \mathbf{U} - \gamma \nabla \mathbf{U}. \quad (10)$$

We now show that γ can be obtained analytically in the following theorem.

Theorem 2. The step size γ in Eq. (10) can be obtained analytically.

Proof. Plugging in the update rule in Eq. (10) into the objective function in Eq. (9), we have

$$\begin{aligned} g(\gamma) &= \frac{1}{2} \|\mathbf{Y} \odot [\mathbf{R} - (\mathbf{U} - \gamma \nabla \mathbf{U}) \mathbf{B} \mathbf{V}^T]\|_F^2 \\ &\quad + \frac{\lambda}{2} \|\tilde{\mathbf{Y}} \odot [\tilde{\mathbf{R}} - (\mathbf{U} - \gamma \nabla \mathbf{U}) \tilde{\mathbf{B}} \mathbf{V}^T]\|_F^2 \\ &= \frac{1}{2} \|\mathbf{Y} \odot (\mathbf{R} - \mathbf{U} \mathbf{B} \mathbf{V}^T) + \gamma \mathbf{Y} \odot (\nabla \mathbf{U} \mathbf{B} \mathbf{V}^T)\|_F^2 \\ &\quad + \frac{\lambda}{2} \|\tilde{\mathbf{Y}} \odot (\tilde{\mathbf{R}} - \mathbf{U} \tilde{\mathbf{B}} \mathbf{V}^T) + \gamma \tilde{\mathbf{Y}} \odot (\nabla \mathbf{U} \tilde{\mathbf{B}} \mathbf{V}^T)\|_F^2. \end{aligned}$$

Denoting $t_1 = \mathbf{Y} \odot (\mathbf{R} - \mathbf{UBV}^T)$, $\tilde{t}_1 = \tilde{\mathbf{Y}} \odot (\tilde{\mathbf{R}} - \mathbf{UBV}^T)$, $t_2 = \mathbf{Y} \odot (\nabla \mathbf{UBV}^T)$, $\tilde{t}_2 = \tilde{\mathbf{Y}} \odot (\nabla \mathbf{UBV}^T)$, we have $g(\gamma) = \frac{1}{2} \|t_1 + \gamma t_2\|_F^2 + \frac{\lambda}{2} \|\tilde{t}_1 + \gamma \tilde{t}_2\|_F^2$, and the gradient

$$\frac{\partial g(\gamma)}{\partial \gamma} = \text{tr}(t_1^T t_2) + \gamma \text{tr}(t_2^T t_2) + \lambda [\text{tr}(\tilde{t}_1^T \tilde{t}_2) + \gamma \text{tr}(\tilde{t}_2^T \tilde{t}_2)],$$

from which we obtain $\gamma = \frac{-\text{tr}(t_1^T t_2) - \lambda \text{tr}(\tilde{t}_1^T \tilde{t}_2)}{\text{tr}(t_2^T t_2) + \lambda \text{tr}(\tilde{t}_2^T \tilde{t}_2)}$ via setting $\frac{\partial g(\gamma)}{\partial \gamma} = 0$. \square

Similarly, we have the update rule for the item-specific latent feature matrix \mathbf{V} ,

$$\mathbf{V} \leftarrow \mathbf{V} - \gamma \nabla \mathbf{V} \quad (11)$$

where $\nabla \mathbf{V} = (\mathbf{I} - \mathbf{VV}^T) \frac{\partial f}{\partial \mathbf{V}}$, and $\frac{\partial f}{\partial \mathbf{V}} = (\mathbf{Y} \odot (\mathbf{UBV}^T - \mathbf{R}))\mathbf{UB} + \lambda (\tilde{\mathbf{Y}} \odot (\mathbf{UBV}^T - \tilde{\mathbf{R}}))\mathbf{UB}$.

Note that the previous works of [10,27] use the gradient descent approach also on a Grassmann manifold. But, they study a single-matrix factorization problem and adopt a different learning algorithm on the Grassmann manifold for searching the step size γ .

Learning \mathbf{B} and $\tilde{\mathbf{B}}$ Given \mathbf{U}, \mathbf{V} , we can estimate \mathbf{B} and $\tilde{\mathbf{B}}$ separately in each data, e.g. for the target data. Let $\mathcal{F}(\mathbf{R} \sim \mathbf{UBV}^T) = \sum_{u=1}^n \sum_{i=1}^m y_{ui} [\frac{1}{2} (r_{ui} - U_u \cdot \mathbf{BV}_i^T)^2 + \frac{\alpha_u}{2} \|U_u\|^2 + \frac{\alpha_v}{2} \|V_i\|^2] + \frac{\beta}{2} \|\mathbf{B}\|_F^2$, we have

$$\begin{aligned} \mathcal{F}(\mathbf{R} \sim \mathbf{UBV}^T) &\propto \sum_{u=1}^n \sum_{i=1}^m y_{ui} \left[\frac{1}{2} (r_{ui} - U_u \cdot \mathbf{BV}_i^T)^2 \right] + \frac{\beta}{2} \|\mathbf{B}\|_F^2 \\ &= \frac{1}{2} \|\mathbf{Y} \odot (\mathbf{R} - \mathbf{UBV}^T)\|_F^2 + \frac{\beta}{2} \|\mathbf{B}\|_F^2. \end{aligned}$$

Thus, we obtain the following equivalent minimization problem,

$$\min_{\mathbf{B}} \frac{1}{2} \|\mathbf{Y} \odot (\mathbf{R} - \mathbf{UBV}^T)\|_F^2 + \frac{\beta}{2} \|\mathbf{B}\|_F^2 \quad (12)$$

where the data-dependent parameter \mathbf{B} can be estimated exactly the same as that of estimating \mathbf{w} in a corresponding least square SVM problem, where $\mathbf{w} = \text{vec}(\mathbf{B}) = [B_{\cdot 1}^T \dots B_{\cdot d}^T]^T \in \mathbb{R}^{d^2 \times 1}$ is a large vector that is concatenated from columns of matrix \mathbf{B} . The instances can be constructed as $\{\mathbf{x}_{ui}, r_{ui}\}$ with $y_{ui} = 1$, where $\mathbf{x}_{ui} = \text{vec}(U_u^T V_i) \in \mathbb{R}^{d^2 \times 1}$. Hence, we obtain the following least-square SVM problem,

$$\min_{\mathbf{w}} \frac{1}{2} \|\mathbf{r} - \mathbf{X}\mathbf{w}\|_F^2 + \frac{\beta}{2} \|\mathbf{w}\|_F^2 \quad (13)$$

where $\mathbf{X} = [\dots \mathbf{x}_{ui} \dots]^T \in \mathbb{R}^{p \times d^2}$ (with $y_{ui} = 1$) is the data matrix, and $\mathbf{r} \in \{1, 2, 3, 4, 5\}^{p \times 1}$ is the corresponding observed ratings from \mathbf{R} . Setting $\nabla \mathbf{w} = -\mathbf{X}^T (\mathbf{r} - \mathbf{X}\mathbf{w}) + \beta \mathbf{w} = \mathbf{0}$, we have

$$\mathbf{w} = (\mathbf{X}^T \mathbf{X} + \beta \mathbf{I})^{-1} \mathbf{X}^T \mathbf{r}. \quad (14)$$

Note that \mathbf{B} or \mathbf{w} can be considered as a linear compact operator [1] and solved efficiently using various existing off-the-shelf tools.

Finally, we can solve the optimization problem in Eq. (6) by alternatively estimating $\mathbf{B}, \tilde{\mathbf{B}}, \mathbf{U}$ and \mathbf{V} . The complete algorithm is given in Fig. 2. Note that we can scale the target matrix \mathbf{R} with $r_{ui} = \frac{r_{ui}-1}{4}$, $y_{ui} = 1$, $u = 1, \dots, n$, $i = 1, \dots, m$, in order to remove the value range difference of two data sources. We adopt random initialization for \mathbf{U}, \mathbf{V} in CMTF and SVD results [17] of $\tilde{\mathbf{R}}$ for that in CSVD.

3.3. Analysis

Each sub-step of updating $\mathbf{B}, \tilde{\mathbf{B}}, \mathbf{U}$ and \mathbf{V} in Fig. 2 will monotonically decrease the objective function in Eq. (6), and hence ensure the convergence to a local minimum. We use a validation data set to determine the convergence condition and tune the parameters (see Section 4.3). The time complexity of TCF and other baseline methods (see Section 4) are obtained as follows: (i) AF: $O(q)$, (ii) PMF [47]: $O(Kqd^2 + K \max(n, m)d^3)$, (iii) cPMF [47]: $O(Kq\tilde{c}d^2 + K \max(n, m)d^3)$, (iv) SVD [48]: $O(nm)$ since it fills the missing ratings with average values, (v) PCC [45]: $O(n^2)$, (vi) OptSpace [27]: $O(Kqd^3 + Kd^6)$, (vii) CMF [52]: $O(K \max(q, \tilde{q})d^2 + K \max(n, m)d^3)$, and (viii) TCF: $O(K \max(q, \tilde{q})d^3 + Kd^6)$, where K is the number of iterations to convergence, q, \tilde{q} ($q, \tilde{q} > n, m$) is the number of non-missing entries in the matrix \mathbf{R} and $\tilde{\mathbf{R}}$, respectively, \tilde{c} is the average number of raters of an item in $\tilde{\mathbf{R}}$, and d is the number of latent features.

Note that the TCF algorithm can be sped up via a stochastic sampling (or stochastic gradient descent) algorithm or distributed computing. More specifically, the step for estimating \mathbf{B} or $\tilde{\mathbf{B}}$ in both CMTF and CSVD is equivalent to that

Input: The target *user-item* numerical rating matrix \mathbf{R} , the auxiliary *user-item* binary rating matrix $\tilde{\mathbf{R}}$, the target *user-item* indicator matrix \mathbf{Y} , the auxiliary *user-item* indicator matrix $\tilde{\mathbf{Y}}$.

Output: The shared user-specific latent feature matrix \mathbf{U} , the shared item-specific latent feature matrix \mathbf{V} , the inner matrix to model the target data-dependent information \mathbf{B} , the inner matrix to model the auxiliary data-dependent information $\tilde{\mathbf{B}}$.

Step 1. Scale ratings in \mathbf{R} ($r_{ui} = \frac{r_{ui}-1}{4}$, $y_{ui} = 1$, $u = 1, \dots, n$, $i = 1, \dots, m$).

Step 2. Initialize \mathbf{U}, \mathbf{V} : randomly initialize \mathbf{U} and \mathbf{V} for CMTF; initialize \mathbf{U} and \mathbf{V} in CSVD using the SVD [17] results of $\tilde{\mathbf{R}}$.

Step 3. Estimate \mathbf{B} and $\tilde{\mathbf{B}}$ as shown in Eq. (14).

Step 4. Update $\mathbf{U}, \mathbf{V}, \mathbf{B}, \tilde{\mathbf{B}}$.

repeat

repeat

 Step 4.1.1. Fix \mathbf{B} and \mathbf{V} , update \mathbf{U} in CMTF as shown in Eq. (7) or CSVD as shown in Eq. (10).

 Step 4.1.2. Fix \mathbf{B} and \mathbf{U} , update \mathbf{V} in CMTF as shown in Eq. (8) or CSVD as shown in Eq. (11).

until Convergence

Step 4.2. Fix \mathbf{U} and \mathbf{V} , update \mathbf{B} and $\tilde{\mathbf{B}}$ as shown in Eq. (14).

until Convergence

Fig. 2. The algorithm of Transfer by Collective Factorization.

of least square SVM, thus various existing off-the-shelf tools can be used, e.g. we can use the stochastic sampling (or stochastic gradient descent) method [8] and distributed algorithms [14]. Second, the step for estimating \mathbf{U}, \mathbf{V} in CMTF can be distributed the same as that of PMF and CMF. For example, once \mathbf{B} and \mathbf{V} are given, each user u 's latent feature vector \mathbf{U}_u is independent of that of other users, which fits the MPI (message passing interface) framework well.

4. Experimental results

Our experiments are designed to verify the following hypotheses. We believe that transfer learning is effective in addressing the data sparsity problem in collaborative filtering, although the smoothing methods are very competitive baselines for the task of missing-value prediction in a sparse rating matrix. In particular,

- (a) we believe that the proposed transfer learning methods, CMTF and CSVD, perform better than baseline algorithms;
- (b) we believe that the transfer learning method CMTF-link is better than the non-transfer learning methods of PMF [47], SVD [48] and OptSpace [27];
- (c) we believe that the transfer learning method CMTF is better than CMF-link, since the inner matrices \mathbf{B} and $\tilde{\mathbf{B}}$ in CMTF are used to capture data-dependent information;
- (d) we believe that the transfer learning method CSVD is better than CMTF, since the orthonormal constraints in CSVD can selectively transfer the most useful knowledge via noise reduction.

We verify each of the above four hypotheses in Section 4.3.

4.1. Data sets and evaluation metrics

We evaluate the proposed method using two movie rating data sets, Moviepilot and Netflix,⁶ and compare to some state-of-the-art baseline algorithms.

Subset of Moviepilot data The Moviepilot rating data contains more than 4.5×10^6 ratings with values in $[0, 100]$, which are given by more than 1.0×10^5 users on around 2.5×10^4 movies [46]. The data set used in the experiments is constructed as follows,

1. we first randomly extract a 2000×2000 dense rating matrix \mathfrak{R} from the Moviepilot data. We then normalize the ratings by $\frac{r_{ui}}{25} + 1$, and the new rating range is $[1, 5]$;
2. we randomly split \mathfrak{R} into training and test sets, T_R, T_E , with 50% ratings, respectively. $T_R, T_E \subset \{(u, i, r_{ui}) \in \mathbb{N} \times \mathbb{N} \times [1, 5] \mid 1 \leq u \leq n, 1 \leq i \leq m\}$. T_E is kept unchanged, while different (average) number of observed ratings for each user, 4, 8, 12, 16, are randomly sampled from T_R for training, with different sparsity ($\sum_{u,i} y_{ui}/n/m$) levels of 0.2%, 0.4%, 0.6% and 0.8% correspondingly;
3. we randomly pick 40 observed ratings on average from T_R for each user to construct the auxiliary data matrix $\tilde{\mathbf{R}}$. To simulate heterogeneous auxiliary and target data, we adopt a pre-processing approach [51] on $\tilde{\mathbf{R}}$, by relabeling ratings with value $r_{ui} \leq 3$ in $\tilde{\mathbf{R}}$ as 0 (dislike), and then ratings with value $r_{ui} > 3$ as 1 (like). The overlap between $\tilde{\mathbf{R}}$ and \mathbf{R} ($\sum_{u,i} y_{ui} \tilde{y}_{ui}/n/m$) is 0.026%, 0.062%, 0.096% and 0.13% correspondingly.

⁶ <http://www.netflix.com>.

Table 3Description of subset of Moviepilot data ($n = m = 2000$) and subset of Netflix data ($n = m = 5000$).

Data set		Form	Sparsity
Moviepilot (subset)	target (training)	$\{1, 5\} \cup \{?\}$	< 1%
	target (test)	$\{1, 5\} \cup \{?\}$	11.4%
	auxiliary	$\{0, 1, ?\}$	2%
Netflix (subset)	target (training)	$\{1, 2, 3, 4, 5, ?\}$	< 1%
	target (test)	$\{1, 2, 3, 4, 5, ?\}$	11.3%
	auxiliary	$\{0, 1, ?\}$	2%

Subset of Netflix data The Netflix rating data contains more than 10^8 ratings with values in $\{1, 2, 3, 4, 5\}$, which are given by more than 4.8×10^5 users on around 1.8×10^4 movies. The data set used in the experiments is constructed as follows,

1. we use the target data in our previous work [43], which is a dense 5000×5000 rating matrix \mathfrak{R} from the Netflix data; more specifically, in [43], we first identify 5000 movies appearing both in MovieLens⁷ and Netflix via the movie title, and then select 10000 most frequent users and another 5000 most popular items from Netflix, and the 5000 items used in this paper are the movies appearing both in MovieLens and Netflix and the 5000 users used in this paper are the most frequent 5000 users;
2. we randomly split \mathfrak{R} into training and test sets, T_R, T_E , with 50% ratings, respectively. T_E is kept unchanged, while different (average) number of observed ratings for each user, 10, 20, 30, 40, are randomly sampled from T_R for training, with different sparsity levels of 0.2%, 0.4%, 0.6% and 0.8% correspondingly;
3. we randomly pick 100 observed ratings on average from T_R for each user to construct the auxiliary data matrix $\tilde{\mathbf{R}}$. To simulate heterogeneous auxiliary and target data, we adopt the pre-processing approach [51] on $\tilde{\mathbf{R}}$, by relabeling 1, 2, 3 ratings in $\tilde{\mathbf{R}}$ as 0 (dislike), and then 4, 5 ratings as 1 (like). The overlap between $\tilde{\mathbf{R}}$ and \mathbf{R} ($\sum_{u,i} y_{ui} \tilde{y}_{ui} / n/m$) is 0.035%, 0.071%, 0.11% and 0.14% correspondingly.

The final data sets⁸ are summarized in Table 3.

Evaluation metrics We adopt the evaluation metrics of Mean Absolute Error (MAE) and Root Mean Square Error (RMSE),

$$MAE = \sum_{(u,i,r_{ui}) \in T_E} |r_{ui} - \hat{r}_{ui}| / |T_E|,$$

$$RMSE = \sqrt{\sum_{(u,i,r_{ui}) \in T_E} (r_{ui} - \hat{r}_{ui})^2 / |T_E|}$$

where r_{ui} and \hat{r}_{ui} are the true and predicted ratings, respectively, and $|T_E|$ is the number of test ratings. In all experiments, we run 3 random trials when generating the required number of observed ratings from T_R , and averaged results are reported.

4.2. Baselines and parameter settings

We compare our TCF method with five non-transfer learning methods: the average filling method (AF), Pearson correlation coefficient (PCC) [45], PMF [47], SVD [48], OptSpace [27], as well as two learning methods using auxiliary data: CMF [52] with logistic link function (CMF-link) and constrained PMF (cPMF) [47].

We study the following six average filling (AF) methods,

$$\begin{aligned} \hat{r}_{ui} &= \bar{r}_{u\cdot}, \\ \hat{r}_{ui} &= \bar{r}_{\cdot i}, \\ \hat{r}_{ui} &= (\bar{r}_{u\cdot} + \bar{r}_{\cdot i}) / 2, \\ \hat{r}_{ui} &= b_{u\cdot} + \bar{r}_{\cdot i}, \\ \hat{r}_{ui} &= \bar{r}_{u\cdot} + b_{\cdot i}, \\ \hat{r}_{ui} &= \bar{r} + b_{u\cdot} + b_{\cdot i} \end{aligned}$$

⁷ <http://www.grouplens.org/node/73>.

⁸ The data and code can be downloaded at <http://www.cse.ust.hk/~weikep/TCF-data-code.zip>.

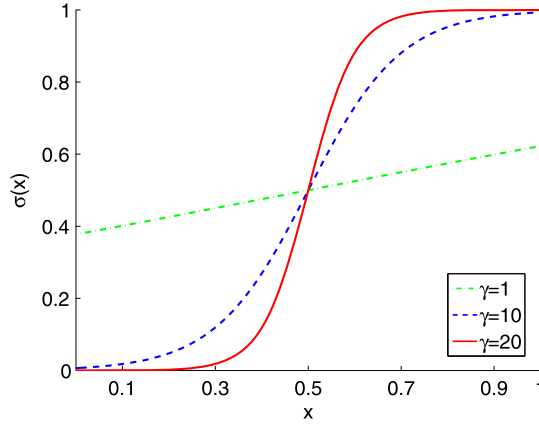


Fig. 3. Logistic link function $\sigma(x) = \frac{1}{1+\exp[-\gamma(x-0.5)]}$.

where $\bar{r}_u = \sum_i y_{ui} r_{ui} / \sum_i y_{ui}$ is the average rating of user u , $\bar{r}_i = \sum_u y_{ui} r_{ui} / \sum_u y_{ui}$ is the average rating of item i , $b_u = \sum_i y_{ui} (r_{ui} - \bar{r}_i) / \sum_i y_{ui}$ is the bias of user u , $b_i = \sum_u y_{ui} (r_{ui} - \bar{r}_u) / \sum_u y_{ui}$ is the bias of item i , and $\bar{r} = \sum_{u,i} y_{ui} r_{ui} / \sum_{u,i} y_{ui}$ is the global average rating. We use $\hat{r}_{ui} = \bar{r} + b_u + b_i$ as it performs best in our experiments. In order to compare with the commonly used average filling methods, we also report the results of $\hat{r}_{ui} = \bar{r}_u$ and $\hat{r}_{ui} = \bar{r}_i$.

For SVD [48], we adopt the approach of 5-star numerical rating predictions, which are reported as the best one in [48]. Specifically, we convert the original rating matrix \mathbf{R} to $\check{\mathbf{R}}$ as follows [48],

$$r_{ui} \rightarrow \check{r}_{ui} = \begin{cases} r_{ui} - \bar{r}_u, & \text{if } y_{ui} = 1 \text{ (rated)}, \\ \bar{r}_i - \bar{r}_u, & \text{if } y_{ui} = 0 \text{ (not rated)} \end{cases}$$

where \bar{r}_u is the user u 's average rating and \bar{r}_i is the item i 's average rating, the same as that used in the aforementioned average filling methods; and then we apply SVD [5,48] on the matrix $\check{\mathbf{R}}$, $\check{\mathbf{R}} = \mathbf{U}\Sigma\mathbf{V}^T$; and finally, the rating of user u on item i can be predicted as follows [48],

$$\hat{r}_{ui} = \bar{r}_u + \mathbf{U}_u \Sigma \mathbf{V}_i^T$$

where the average rating \bar{r}_u is added to the prediction rule.

For PCC, since the data matrices are sparse, we use the whole set of neighboring users in the prediction rule. For PMF, cPMF, SVD, OptSpace, CMF-link and TCF, we fix the latent feature number $d = 10$. For PMF, different tradeoff parameters of $\alpha_u = \alpha_v \in \{0.01, 0.1, 1\}$ are tried; for cPMF, different tradeoff parameters of $\alpha_u = \alpha_v = \alpha_w \in \{0.01, 0.1, 1\}$ are tried; for CMF-link, different tradeoff parameters $\alpha_u = \alpha_v \in \{0.01, 0.1, 1\}$, $\lambda \in \{0.01, 0.1, 1\}$ are tried; for CMTF, β is fixed as 1, and different tradeoff parameters $\alpha_u = \alpha_v \in \{0.01, 0.1, 1\}$, $\lambda \in \{0.01, 0.1, 1\}$ are tried; for CSVD, different tradeoff parameters $\lambda \in \{0.01, 0.1, 1\}$ are tried.

To alleviate the data heterogeneity of $\{0, 1\}$ and $\frac{\{1,2,3,4,5\}-1}{4}$ or $\frac{\{1,2,3,4,5\}-1}{4}$, a logistic link function $\sigma(\mathbf{U}_u \mathbf{V}_i^T)$ is embedded in the auxiliary data matrix factorization of CMF,

$$\begin{aligned} \min_{\mathbf{U}, \mathbf{V}} \sum_{u=1}^N \sum_{i=1}^M y_{ui} & \left[\frac{1}{2} (r_{ui} - \mathbf{U}_u \mathbf{V}_i^T)^2 + \frac{\alpha_u}{2} \|\mathbf{U}_u\|^2 + \frac{\alpha_v}{2} \|\mathbf{V}_i\|^2 \right] \\ & + \lambda \sum_{u=1}^N \sum_{i=1}^M \check{y}_{ui} \left[\frac{1}{2} (\check{r}_{ui} - \sigma(\mathbf{U}_u \mathbf{V}_i^T))^2 + \frac{\alpha_u}{2} \|\mathbf{U}_u\|^2 + \frac{\alpha_v}{2} \|\mathbf{V}_i\|^2 \right] \end{aligned}$$

where $\sigma(x) = \frac{1}{1+\exp[-\gamma(x-0.5)]}$ (see Fig. 3) and different parameters $\gamma \in \{1, 10, 20\}$ are tried.

For cPMF [47], we integrate the auxiliary data as follows,

$$\begin{aligned} \min_{\mathbf{U}, \mathbf{V}, \mathbf{W}} \sum_{u=1}^n \sum_{i=1}^m y_{ui} & \left[\frac{1}{2} \left(r_{ui} - \left(\mathbf{U}_u + \sum_{j=1}^m \check{y}_{uj} \mathbf{W}_j / \sum_{j=1}^m \check{y}_{uj} \right) \mathbf{V}_i^T \right)^2 \right. \\ & \left. + \frac{\alpha_u}{2} \|\mathbf{U}_u\|^2 + \frac{\alpha_v}{2} \|\mathbf{V}_i\|^2 + \frac{\alpha_w}{2} \sum_{j=1}^m \|\mathbf{W}_j\|^2 \right] \end{aligned}$$

where $\mathbf{U} \in \mathbb{R}^{n \times d}$ is a user-specific latent feature matrix, $\mathbf{V} \in \mathbb{R}^{m \times d}$ is an item-specific latent feature matrix, and $\mathbf{W} \in \mathbb{R}^{m \times d}$ is called the latent similarity constraint matrix [47]. Once we have learned the model parameters, we can predict the rating

Table 4

Prediction performance on the subset of Moviepilot data (see Table 3) of AF for $\hat{r}_{ui} = \bar{r} + b_{u\cdot} + b_{\cdot i}$, AF (user) for $\hat{r}_{ui} = \bar{r}_{u\cdot}$, AF (item) for $\hat{r}_{ui} = \bar{r}_{\cdot i}$, PCC [45], SVD [48], PMF [47], cPMF for constrained PMF [47], OptSpace [27], CMF-link for CMF [52] with logistic link function, and two variants of *Transfer by Collective Factorization*, TCF (CMTF) and TCF (CSVD). Numbers in boldface (e.g. **0.7087**) and in Italic (e.g. *0.7415*) are the best and second best results among all methods, respectively.

Metrics	Methods	Sparsity			
		0.2%	0.4%	0.6%	0.8%
		(tr. 3, val. 1)	(tr. 7, val. 1)	(tr. 11, val. 1)	(tr. 15, val. 1)
MAE	AF	0.7942±0.0047	0.7259±0.0022	0.6956±0.0017	0.6798±0.0010
	AF (user)	0.8269±0.0081	0.7819±0.0041	0.7643±0.0018	0.7559±0.0011
	AF (item)	0.8126±0.0035	0.7721±0.0014	0.7541±0.0011	0.7449±0.0002
	PCC	0.7956±0.0237	0.7785±0.0102	0.7215±0.0211	0.6766±0.0095
	PMF	0.8118±0.0014	0.7794±0.0009	0.7602±0.0009	0.7513±0.0005
	cPMF	0.8368±0.0012	0.7681±0.0011	0.7526±0.0013	0.7462±0.0007
	SVD	0.8262±0.0081	0.7796±0.0039	0.7603±0.0017	0.7505±0.0013
	OptSpace	1.3465±0.0352	0.7971±0.0031	0.7541±0.0039	0.7260±0.0024
	CMF-link	0.9956±0.0149	0.7632±0.0005	0.7121±0.0007	0.6905±0.0007
	TCF (CMTF)	<i>0.7415±0.0018</i>	0.7021±0.0020	0.6871±0.0013	0.6776±0.0006
	TCF (CSVD)	0.7087±0.0035	0.6860±0.0023	0.6743±0.0048	0.6612±0.0028
RMSE	AF	1.0391±0.0071	0.9558±0.002	0.9177±0.0017	0.8977±0.0002
	AF (user)	1.0867±0.0120	1.0206±0.0054	0.9929±0.0025	0.9802±0.0015
	AF (item)	1.0615±0.0053	1.0073±0.0012	0.9836±0.0009	0.9722±0.0003
	PCC	1.0395±0.0358	1.0217±0.0091	0.9582±0.0261	0.9005±0.0125
	PMF	1.0330±0.0012	1.0123±0.0013	0.9832±0.0009	0.9706±0.0003
	cPMF	1.0906±0.0016	0.9900±0.0004	0.9679±0.0013	0.9599±0.0004
	SVD	1.0869±0.0121	1.0210±0.0053	0.9936±0.0024	0.9813±0.0014
	OptSpace	1.7189±0.0314	1.0611±0.0062	0.9952±0.0024	0.9543±0.0042
	CMF-link	1.3024±0.0170	1.0066±0.0036	0.9366±0.0007	0.9072±0.0009
	TCF (CMTF)	<i>0.9449±0.0018</i>	<i>0.9109±0.0013</i>	0.8967±0.0011	0.8875±0.0003
	TCF (CSVD)	0.9298±0.0038	0.9039±0.0018	0.8898±0.0052	0.8744±0.0033

of user u on item i as $\hat{r}_{ui} = (U_{u\cdot} + \sum_{j=1}^m \tilde{y}_{uj} W_{j\cdot} / \sum_{j=1}^m \tilde{y}_{uj}) V_{\cdot i}^T$. In our experiments of cPMF, we use auxiliary data of “like” since it produces better results than using both “like” and “dislike.”

4.3. Summary of the experimental results

We randomly sample n ratings (one rating per user on average) from the training data \mathbf{R} and use them as the validation set to determine the tradeoff parameters (α_u , α_v , α_w , β , λ) and the number of iterations to convergence for PMF, cPMF, OptSpace, CMF-link and TCF. For AF, PCC and SVD, both the training set and validation set are combined as one set of training data. The results on test data (unavailable during training) are reported in Tables 4 and 5. We can make the following observations:

1. For the smoothing method of average filling (AF), we can see that the best variant, $\hat{r}_{ui} = \bar{r}_{u\cdot} + b_{u\cdot} + b_{\cdot i}$, is very competitive for sparse rating data, while the commonly used average filling methods of $\hat{r}_{ui} = \bar{r}_{u\cdot}$ and $\hat{r}_{ui} = \bar{r}_{\cdot i}$ are much worse. There are two reasons for the advantages of AF. First, average filling is a very strong baseline, especially on small and dense subsets of the Netflix and Moviepilot data. Second, PMF and cPMF show their advantages when the *user-item* rating matrix is large, e.g. the whole data set used in the Netflix competition, and can be improved if we tune the parameters in finer granularity.
2. For matrix factorization methods with orthonormal constraint including SVD and OptSpace, we can see that SVD is better than OptSpace when the sparsity is lower (e.g. $\leq 0.6\%$ for Moviepilot and $\leq 0.4\%$ for Netflix), while OptSpace beats SVD when the rating matrix becomes denser, which can be explained by the different strategies adopted by SVD and OptSpace for missing ratings. SVD fills the missing ratings with average values, which may help for an extremely sparse rating matrix, but will hurt the performance when the rating matrix becomes denser.
3. For the sparsity problem in collaborative filtering, transfer learning is a very attractive technique:
 - (a) The proposed transfer learning methods of CMTF and CSVD perform significantly better than all other baselines at all sparsity levels.
 - (b) For the transfer learning method of CMF-link, we can see that it is significantly better than the non-transfer learning methods of PMF, SVD and OptSpace at almost all sparsity levels (except the extremely sparse case of 0.2% on Moviepilot), but is still worse than AF, which can be explained by the heterogeneity of the auxiliary binary rating data and target numerical rating data, and the usefulness of smoothing (AF) for sparse data. For PMF and cPMF, we can see that cPMF with auxiliary data is better than PMF in most cases.
 - (c) For the transfer learning methods of CMTF and CMF-link, we can see that CMTF performs better than CMF-link in all cases, which shows the advantages of modeling the data-dependent effect using inner matrices \mathbf{B} and $\tilde{\mathbf{B}}$ in CMTF.

Table 5

Prediction performance on the subset of Netflix data (see Table 3) of AF for $\hat{r}_{ui} = \bar{r} + b_u + b_i$, AF (user) for $\hat{r}_{ui} = \bar{r}_u$, AF (item) for $\hat{r}_{ui} = \bar{r}_i$, PCC [45], SVD [48], PMF [47], cPMF for constrained PMF [47], OptSpace [27], CMF-link for CMF [52] with logistic link function, and two variants of *Transfer by Collective Factorization*, TCF (CMTF) and TCF (CSVD). Numbers in boldface (e.g. **0.7405**) and in Italic (e.g. *0.7589*) are the best and second best results among all methods, respectively.

Metrics	Methods	Sparsity			
		0.2%	0.4%	0.6%	0.8%
		(tr. 9, val. 1)	(tr. 19, val. 1)	(tr. 29, val. 1)	(tr. 39, val. 1)
MAE	AF	0.7765±0.0006	0.7429±0.0006	0.7308±0.0005	0.7246±0.0003
	AF (user)	0.8060±0.0021	0.7865±0.0010	0.7798±0.0009	0.7767±0.0003
	AF (item)	0.8535±0.0007	0.8372±0.0005	0.8304±0.0002	0.8270±0.0001
	PCC	0.8233±0.0228	0.7888±0.0418	0.7714±0.0664	0.7788±0.0516
	PMF	0.8879±0.0008	0.8467±0.0006	0.8087±0.0188	0.7642±0.0003
	cPMF	0.8491±0.0181	0.8147±0.0006	0.8122±0.0005	0.7864±0.0057
	SVD	0.8055±0.0021	0.7846±0.0010	0.7757±0.0009	0.7711±0.0002
	OptSpace	0.8276±0.0004	0.7812±0.0040	0.7572±0.0027	0.7418±0.0038
	CMF-link	0.7994±0.0017	0.7508±0.0008	0.7365±0.0004	0.7295±0.0003
	TCF (CMTF)	0.7589±0.0175	0.7195±0.0055	0.7031±0.0005	0.6962±0.0009
	TCF (CSVD)	0.7405 ±0.0007	0.7080 ±0.0002	0.6948 ±0.0007	0.6877 ±0.0007
RMSE	AF	0.9855±0.0004	0.9427±0.0007	0.9277±0.0006	0.9200±0.0002
	AF (user)	1.0208±0.0015	0.9921±0.0012	0.9834±0.0004	0.9791±0.0002
	AF (item)	1.0708±0.0011	1.0477±0.0005	1.0386±0.0004	1.0339±0.0001
	PCC	1.0462±0.0326	1.0041±0.0518	0.9841±0.0848	0.9934±0.0662
	PMF	1.0779±0.0001	1.0473±0.0004	1.0205±0.0112	0.9691±0.0007
	cPMF	1.0606±0.0199	1.0125±0.0007	1.0066±0.0006	0.9930±0.0044
	SVD	1.0202±0.0014	0.9906±0.0012	0.9798±0.0005	0.9741±0.0004
	OptSpace	1.0676±0.0020	1.0089±0.0024	0.9750±0.0010	0.9543±0.0037
	CMF-link	1.0204±0.0013	0.9552±0.0009	0.9369±0.0004	0.9277±0.0004
	TCF (CMTF)	0.9653±0.0198	0.9171±0.0063	0.8971±0.0005	0.8884±0.0007
	TCF (CSVD)	0.9502 ±0.0005	0.9074 ±0.0004	0.8903 ±0.0006	0.8809 ±0.0005

- (d) For the two variants of TCF, we can see that the transfer learning method CSVD further improves the performance over CMTF in all cases, which shows the effect of noise reduction from the orthonormal constraints, $\mathbf{U}^T \mathbf{U} = \mathbf{I}$ and $\mathbf{V}^T \mathbf{V} = \mathbf{I}$.

To further study the effectiveness of selective transfer via noise reduction in TCF, we compare the performance of CMTF and CSVD at different sparsity levels with different auxiliary data of sparsity 1%, 2% and 3% on the subset Netflix data. The results are shown in Fig. 4. We can see that CSVD performs better than CMTF in all cases, which again shows the advantage of CSVD in transferring the most useful knowledge.

There is a very fundamental question in transfer learning [41], namely *when to transfer*, which is related to *negative transfer* [40]. For our problem setting (see Fig. 1), negative transfer [40] may happen when the density of auxiliary binary ratings is lower than that of target numerical ratings, or the semantic meaning of auxiliary binary ratings are completely different from that of target numerical ratings. However, in our work, we assume that the auxiliary binary ratings are denser than the target numerical ratings, and both ratings are related though there are some differences. Thus, under our assumption, negative transfer is not likely to happen. In fact, negative transfer is not observed in our empirical studies.

5. Related works

SVD Low-rank singular value decomposition (SVD) or principal component analysis (PCA) [5,25] is widely used in information retrieval and data mining to find latent topics [18] and to reduce noise [6]. These solutions have also been applied in collaborative filtering [24,22,7,44,48,53,29,10,27]. Among them, some works apply non-iterative SVD or PCA on a full matrix after some preprocessing to remove the missing values [24,22,7,44,48], while other works [53,29] use iterative SVD on a full matrix in an expectation-maximization (EM) procedure. Still other works [10,27] take the missing ratings as unknown and directly optimize the objective function over the observed ratings only. Our strategy is similar to that of [10,27], since we also take missing ratings as unknown. We use two representative methods of SVD [48] and OptSpace [27] as our baselines in the experiments.

The differences of our approach and those previously published SVD-based methods can be identified from two aspects. First, we take missing ratings as unknown, while most previous works pre-process the rating matrix to obtain a full matrix on which PCA or SVD is applied. Second, we make use of some auxiliary data besides the target rating data via transfer learning techniques, while the aforementioned works only have a target rating matrix.

PMF PMF [47] is a recently proposed method for missing-value prediction in a single matrix, which can be reduced from TCF in Eq. (6) when $\mathcal{D} = \mathcal{D}_{\mathbb{R}}$, $\lambda = 0$, $\beta = 0$ and $\mathbf{B} = \mathbf{I}$. The RSTE model [38] generalizes PMF and factorizes a single rating

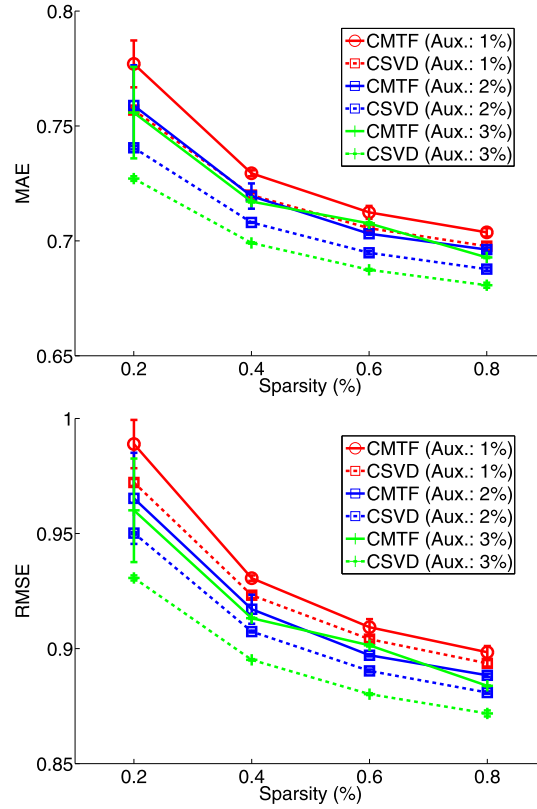


Fig. 4. Prediction performance of TCF (CMTF, CSVD) on Netflix at different sparsity levels with different auxiliary data.

matrix with a regularization term from the user-side social data, which is different from our two-matrix factorization model. The PLRM model [57] generalizes the PMF model to incorporate numerical ratings, *implicit* purchasing data, meta data and social network information, but does not consider the *explicit* auxiliary data of both like and dislike. Mathematically, the PLRM model only considering numerical ratings and *implicit feedback* can be considered as a special case of our TCF framework, CMTF for $\mathcal{D} = \mathcal{D}_R$, but the learning algorithm is still different since CMTF has closed-form solutions for all steps.

CMF CMF [52] is proposed for jointly factorizing two matrices with the constraints of sharing item-specific latent features, and SoRec [39] is proposed for sharing user-specific latent features. CMF and SoRec can be reduced from TCF in Eq. (6) when $\mathcal{D} = \mathcal{D}_R$, $\beta = 0$, $\mathbf{B} = \tilde{\mathbf{B}} = \mathbf{I}$, and only requiring one-side latent feature matrix to be the same, e.g. user side of $\mathbf{R} \sim \mathbf{U}\mathbf{V}^T$, $\tilde{\mathbf{R}} \sim \mathbf{U}\tilde{\mathbf{V}}^T$, or item side of $\mathbf{R} \sim \mathbf{U}\mathbf{V}^T$, $\tilde{\mathbf{R}} \sim \tilde{\mathbf{U}}\mathbf{V}^T$. However, in our problem setting as shown in Fig. 1, both users and items are aligned. To alleviate the data heterogeneity in CMF or SoRec, we embed a logistic link function in the auxiliary data matrix factorization in our experiments.

There are at least three differences between TCF and CMF. First, TCF is a trilinear method, $\mathbf{R} = \mathbf{U}\mathbf{B}\mathbf{V}^T$, $\tilde{\mathbf{R}} = \mathbf{U}\tilde{\mathbf{B}}\mathbf{V}^T$, where the inner matrices \mathbf{B} and $\tilde{\mathbf{B}}$ are designed to capture the domain-dependent information, while CMF is a bilinear method and cannot be applied to our studied problem (see Fig. 1). Second, we introduce orthonormal constraints in one variant of TCF, CSVD, which is empirically proved to be more effective on noise reduction, while CMF does not have such constraints and effect. Finally, the learning algorithms of TCF (CSVD), TCF (CMTF) and CMF are different.

DPMF Dependent probabilistic matrix factorization (DPMF) [2] is a multi-task version of PMF based on Gaussian processes, which is proposed for incorporating *homogeneous*, but not *heterogeneous*, side information via sharing the inner covariance matrices of user-specific and item-specific latent features. The slice sampling algorithm used in DPMF may be too time consuming for some medium sized problems, e.g. the problems studied in the experiments.

CST Coordinate system transfer (CST) [43] is a recently proposed transfer learning method in collaborative filtering to transfer the coordinate system from two auxiliary CF matrices to a target one in an adaptive way. CST performs quite well when the coordinate system is constructed when the auxiliary data is dense, and when the target data is not very sparse [43]. However, when the auxiliary and target data are not so dense, constructing the shared latent feature matrices in a collective way as used in TCF may perform better, since the collective behavior brings in richer interactions when bridging two data sources [13,54].

Table 6

Summary of related work of transfer learning in collaborative filtering.

	Knowledge (what to transfer)	Algorithm style (how to transfer)	
		Adaptive	Collective
PMF [47] family	Covariance Latent features	CST [43]	DPMF [2] SoRec [39], CMF [52], TCF
NMF [30] family	Codebook Latent features	CBT [31]	RMGM [32] WNMCTF [56]

Parallel to the PMF family of CMF and DPMF, there is a corresponding NMF [30] family with non-negative constraints:

1. Trilinear method of WNMCTF [56] is proposed to factorize three matrices of *user-item*, *item-content* and *user-demographics*, and
2. codebook sharing methods of CBT [31] and RMGM [32] can be considered as adaptive and collective extensions of [50,19]. RMGM-OT [33] is a follow-up work of RMGM [32], which studies the effect of user preferences over time by sharing the cluster-level rating patterns across temporal domains. This work focused on homogeneous user feedbacks of 5-star grades instead of heterogeneous user feedbacks.

Models in the NMF family usually have better interpretability, e.g. the learned latent feature matrices \mathbf{U} and \mathbf{V} in CBT [31] and RMGM [32] can be considered as memberships of the corresponding users and items, while the top ranking models [28] in collaborative filtering are from the PMF family. We summarize the above related work in Table 6, in the perspective of whether having non-negative constraints on the latent variables, and what & how to transfer in transfer learning [41].

Clustering on relational data Long et al. [36,37] study a clustering problem on a full matrix without missing values, which is different from our problem setting for missing rating prediction, while the idea of sharing common subspace or latent feature matrices is similar to ours. Cohn et al. [15] study document clustering using content information and auxiliary information of *document-document* link information, while the two matrices of *term-document* and *document-document* are both full without missing values. Banerjee et al. [3] study clustering of relational data without missing values or the missing entries are imputed with zeros, while our approach takes missing values as unknown and aims for missing rating prediction.

Logistic loss function in matrix factorization There are some matrix factorization methods using logistic loss functions for binary rating data [16,26,49]. There are two reasons why we do not use such loss functions. First, using different loss functions, e.g. the logistic loss function in binary PCA [16,26,49], is a vertical research direction to our focus of developing transfer learning solutions, and we will study this issue in our future work. Second, it is difficult to justify using logistic loss function [16,26,49] in the factorization of the auxiliary binary rating matrix and square loss function in the target numerical rating matrix, since the objective functions are then totally different, and thus the meanings and scales of the user-specific latent feature matrix \mathbf{U} in two domains are not comparable (similar for \mathbf{V}), which may cause the difficulty of knowledge sharing.

We illustrate the two loss functions bellow,

$$- [r_{ui} \log \hat{r}_{ui} + (1 - r_{ui}) \log(1 - \hat{r}_{ui})] \quad \text{vs.} \quad (r_{ui} - U_u \cdot V_i^T)^2$$

where $r_{ui} \in \{0, 1\}$ is the true binary rating, $\hat{r}_{ui} = \sigma(U_u \cdot V_i^T) \in [0, 1]$ is the predicted rating, and $\sigma(\theta) = \frac{1}{1 + \exp(-\theta)}$ is the sigmoid function (or logistic link function).

Furthermore, to address the heterogeneities of numerical ratings and binary ratings, we have scaled the 5-star numerical ratings to the range of $[0, 1]$ and then introduced a sigmoid link function (or logistic link function) instead of logistic loss function as follows (see Section 4),

$$(r_{ui} - \hat{r}_{ui})^2 \quad \text{vs.} \quad (r_{ui} - U_u \cdot V_i^T)^2$$

where $\hat{r}_{ui} = \sigma(U_u \cdot V_i^T) \in [0, 1]$ is the predicted rating.

To sum up, the differences between our proposed transfer learning solution and other works include the following. First, we focus on missing rating prediction instead of clustering [36]. Second, we study auxiliary data of user feedbacks instead of content information [52]. Third, we leverage auxiliary data from frontal side instead of user side [11] or item side [52]. Fourth, we take missing ratings as unknown instead of negative feedbacks of zeros [3] in order to optimize the objective function specifically on the observed ratings only. Fifth, we introduce orthonormal constraints instead of non-negative constraints [56] to resemble the effect of noise reduction. Sixth, we design a collective algorithm instead of an adaptive algorithm for richer interactions between the auxiliary domain and the target domain [13,54]. Seventh, we transfer knowledge of latent features among all aligned users and items instead of sharing only compressed knowledge of cluster-level

rating patterns [31,32] or covariance matrix [2]. Finally, we extend a trilinear base model instead of a bilinear model [52] to capture both domain-independent knowledge and domain-dependent effect. In summary, the first three points illustrate the novelty of our proposed problem setting, and the next six points show the novelty of our designed algorithm.

6. Conclusions and future work

In this paper, we investigate how to address the *sparsity* problem in collaborative filtering via a transfer learning solution. Specifically, we present a novel transfer learning framework of *Transfer by Collective Factorization*, to transfer knowledge from auxiliary data of explicit binary ratings (like and dislike), which alleviates the data sparsity problem in the target numerical ratings. Note that we assume the 5-star ratings are unordered bins instead of ordinal relative preferences. Our method constructs the shared latent space \mathbf{U}, \mathbf{V} in a collective manner, captures the data-dependent effect via learning inner matrices $\mathbf{B}, \tilde{\mathbf{B}}$ separately, and selectively transfers the most useful knowledge via noise reduction by introducing orthonormal constraints. The novelty of our algorithm includes generalizing transfer learning methods in collaborative filtering in a principled way. Experimental results show that TCF performs significantly better than several state-of-the-art baseline algorithms at various sparsity levels.

The problem setting of TCF (Fig. 1) for *heterogeneous explicit user feedbacks* is novel and widely applicable in many applications beyond the *user-item* representation in recommender systems. Examples include *query-document* in information retrieval, *author-word* in academic publications, *user-community* in social network services [59], *location-activity* in ubiquitous computing [60], and even *drug-protein* in biomedicine, etc.

For our future work, we will study and extend the transfer learning framework in additional areas and to include more theoretical analysis and larger-scale experiments. In particular, we will address the “pure” cold-start recommendation problem for users without any rating, sparse learning and matrix completion [27], partial correspondence between users and items [34], distributed implementation on the MPI framework, adaptive transfer learning [12] in collaborative filtering, more complex user feedbacks of different rating distributions, and different loss functions [16,26], etc.

Acknowledgements

We thank the support of RGC-NSFC Joint Research Grant N_HKUST624/09 and Hong Kong RGC Grant 621211. We also thank the anonymous reviewers for their detailed and helpful comments.

References

- [1] Jacob Abernethy, Francis Bach, Theodoros Evgeniou, Jean-Philippe Vert, A new approach to collaborative filtering: Operator estimation with spectral regularization, *J. Mach. Learn. Res.* 10 (June 2009) 803–826.
- [2] Ryan P. Adams, George E. Dahl, Iain Murray, Incorporating side information into probabilistic matrix factorization using Gaussian processes, in: *Uncertainty in Artificial Intelligence (UAI)*, 2010, pp. 1–9.
- [3] Arindam Banerjee, Sugato Basu, Srulana Merugu, Multi-way clustering on relation graphs, in: *SIAM International Conference on Data Mining (SDM)*, 2007.
- [4] Robert M. Bell, Yehuda Koren, Scalable collaborative filtering with jointly derived neighborhood interpolation weights, in: *Proceedings of the 2007 Seventh IEEE International Conference on Data Mining*, IEEE Computer Society, Washington, DC, USA, 2007, pp. 43–52.
- [5] Michael W. Berry, Svdpack: A fortran-77 software library for the sparse singular value decomposition, Technical report, Knoxville, TN, USA, 1992.
- [6] Michael W. Berry, Susan T. Dumais, Gavin W. O'Brien, Using linear algebra for intelligent information retrieval, *SIAM Rev.* 37 (December 1995) 573–595.
- [7] Daniel Billsus, Michael J. Pazzani, Learning collaborative information filters, in: *Proceedings of the Fifteenth International Conference on Machine Learning, ICML'98*, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1998, pp. 46–54.
- [8] Léon Bottou, Large-scale machine learning with stochastic gradient descent, in: Yves Lechevallier, Gilbert Saporta (Eds.), *Proceedings of the 19th International Conference on Computational Statistics (COMPSTAT'2010)*, Springer, Paris, France, August 2010, pp. 177–187.
- [9] John S. Breese, David Heckerman, Carl Myers Kadie, Empirical analysis of predictive algorithms for collaborative filtering, Technical report, MSR-TR-98-12, 1998.
- [10] Nicoletta Del Buono, Tiziano Politi, A continuous technique for the weighted low-rank approximation problem, in: *International Conference on Computational Science and Applications (ICCSA)*, 2004, pp. 988–997.
- [11] Bin Cao, Nathan Nan Liu, Qiang Yang, Transfer learning for collective link prediction in multiple heterogenous domains, in: *International Conference on Machine Learning (ICML)*, 2010, pp. 159–166.
- [12] Bin Cao, Sinno Jialin Pan, Yu Zhang, Dit-Yan Yeung, Qiang Yang, Adaptive transfer learning, in: *Twenty-Fourth Conference on Artificial Intelligence (AAAI)*, 2010.
- [13] Rich Caruana, Multitask learning, *Mach. Learn.* 28 (July 1997) 41–75.
- [14] Edward Y. Chang, Hongjie Bai, Kaihua Zhu, Hao Wang, Jian Li, Zhihuan Qiu, PSVM: Parallel support vector machines with incomplete Cholesky factorization, in: *Scaling up Machine Learning: Parallel and Distributed Approaches*, Cambridge Univ. Press, 2011.
- [15] David Cohn, Deepak Verma, Karl Pfleger, Recursive attribute factoring, in: *Neural Information Processing Systems (NIPS)*, 2006, pp. 297–304.
- [16] Michael Collins, S. Dasgupta, Robert E. Schapire, A generalization of principal components analysis to the exponential family, in: *Neural Information Processing Systems (NIPS)*, 2001, pp. 617–624.
- [17] Paolo Cremonesi, Yehuda Koren, Roberto Turrin, Performance of recommender algorithms on top-n recommendation tasks, in: *Proceedings of the Fourth ACM Conference on Recommender Systems, RecSys'10*, ACM, New York, NY, USA, 2010, pp. 39–46.
- [18] Scott C. Deerwester, Susan T. Dumais, Thomas K. Landauer, George W. Furnas, Richard A. Harshman, Indexing by latent semantic analysis, *J. Am. Soc. Inf. Sci.* 41 (6) (1990) 391–407.
- [19] Chris Ding, Tao Li, Wei Peng, Haesun Park, Orthogonal nonnegative matrix tri-factorizations for clustering, in: *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD'06*, ACM, New York, NY, USA, 2006, pp. 126–135.
- [20] Chris H.Q. Ding, Jieping Ye, 2-dimensional singular value decomposition for 2d maps and images, in: *SIAM International Conference on Data Mining (SDM)*, 2005, pp. 32–43.

- [21] Alan Edelman, Tomás A. Arias, Steven T. Smith, The geometry of algorithms with orthogonality constraints, *SIAM J. Matrix Anal. Appl.* 20 (2) (1999) 303–353.
- [22] Danyel Fisher, Kris Hildrum, Jason Hong, Mark Newman, Megan Thomas, Rich Vuduc, Swami (poster session): A framework for collaborative filtering algorithm development and evaluation, in: *Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR'00*, ACM, New York, NY, USA, 2000, pp. 366–368.
- [23] David Goldberg, David Nichols, Brian M. Oki, Douglas Terry, Using collaborative filtering to weave an information tapestry, *Commun. ACM* 35 (December 1992) 61–70.
- [24] Ken Goldberg, Theresa Roeder, Dhruv Gupta, Chris Perkins, Eigentaste: A constant time collaborative filtering algorithm, *Inf. Retr.* 4 (July 2001) 133–151.
- [25] Gene H. Golub, Charles F. Van Loan, *Matrix Computations*, 3rd ed., Johns Hopkins University Press, Baltimore, MD, USA, 1996.
- [26] Geoffrey J. Gordon, Generalized² linear² models, in: *Neural Information Processing Systems (NIPS)*, 2002, pp. 577–584.
- [27] Raghunandan H. Keshavan, Andrea Montanari, Sewoong Oh, Matrix completion from noisy entries, *J. Mach. Learn. Res.* 11 (August 2010) 2057–2078.
- [28] Yehuda Koren, Factor in the neighbors: Scalable and accurate collaborative filtering, *ACM Trans. Knowl. Discov. Data* 4 (1) (January 2010) 1–24.
- [29] Miklós Kurucz, András A. Benczúr, Balázs Torma, Methods for large scale svd with missing values, in: *KDDCup 2007*, 2007.
- [30] Daniel D. Lee, H. Sebastian Seung, Algorithms for non-negative matrix factorization, in: *Neural Information Processing Systems (NIPS)*, 2001, pp. 556–562.
- [31] Bin Li, Qiang Yang, Xiangyang Xue, Can movies and books collaborate? Cross-domain collaborative filtering for sparsity reduction, in: *International Joint Conferences on Artificial Intelligence (IJCAI)*, 2009, pp. 2052–2057.
- [32] Bin Li, Qiang Yang, Xiangyang Xue, Transfer learning for collaborative filtering via a rating-matrix generative model, in: *International Conference on Machine Learning (ICML)*, 2009, pp. 617–624.
- [33] Bin Li, Xingquan Zhu, Ruijiang Li, Chengqi Zhang, Xiangyang Xue, Xindong Wu, Cross-domain collaborative filtering over time, in: *IJCAI*, 2011, pp. 2293–2298.
- [34] Tao Li, Vikas Sindhwani, Chris H.Q. Ding, Yi Zhang, Bridging domains with words: Opinion analysis with matrix tri-factorizations, in: *SIAM International Conference on Data Mining (SDM)*, 2010, pp. 293–302.
- [35] Nathan N. Liu, Evan W. Xiang, Min Zhao, Qiang Yang, Unifying explicit and implicit feedback for collaborative filtering, in: *Proceedings of the 19th ACM International Conference on Information and Knowledge Management, CIKM'10*, ACM, New York, NY, USA, 2010, pp. 1445–1448.
- [36] Bo Long, Zhongfei (Mark) Zhang, Xiaoyun Wú, Philip S. Yu, Spectral clustering for multi-type relational data, in: *Proceedings of the 23rd International Conference on Machine Learning, ICML'06*, ACM, New York, NY, USA, 2006, pp. 585–592.
- [37] Bo Long, Zhongfei Mark Zhang, Philip S. Yu, A probabilistic framework for relational clustering, in: *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD'07*, ACM, New York, NY, USA, 2007, pp. 470–479.
- [38] Hao Ma, Irwin King, Michael R. Lyu, Learning to recommend with explicit and implicit social relations, *ACM Trans. Intell. Syst. Technol.* 2 (3) (May 2011) 1–19.
- [39] Hao Ma, Haixuan Yang, Michael R. Lyu, Irwin King, Sorec: Social recommendation using probabilistic matrix factorization, in: *ACM Conference on Information and Knowledge Management (CIKM)*, 2008, pp. 931–940.
- [40] Leslie Pack Kaelbling, Michael T. Rosenstein, Zvika Marx, To transfer or not to transfer, in: *Neural Information Processing Systems (NIPS)*, 2005.
- [41] Sinno Jialin Pan, Qiang Yang, A survey on transfer learning, *IEEE Trans. Knowl. Data Eng.* 22 (10) (2010) 1345–1359.
- [42] Weike Pan, Nathan N. Liu, Evan W. Xiang, Qiang Yang, Transfer learning to predict missing ratings via heterogeneous user feedbacks, in: *International Joint Conferences on Artificial Intelligence (IJCAI)*, 2011, pp. 2318–2323.
- [43] Weike Pan, Evan W. Xiang, Nathan N. Liu, Qiang Yang, Transfer learning in collaborative filtering for sparsity reduction, in: *Twenty-Fourth Conference on Artificial Intelligence (AAAI)*, 2010, pp. 230–235.
- [44] Michael H. Pryor, The effects of singular value decomposition on collaborative filtering, Technical report, Hanover, NH, USA, 1998.
- [45] Paul Resnick, Neophytos Iacovou, Mitesh Suchak, Peter Bergstrom, John Riedl, GroupLens: An open architecture for collaborative filtering of netnews, in: *Computer Supported Cooperative Work (CSCW)*, 1994, pp. 175–186.
- [46] Alan Said, Shlomo Berkovsky, Ernesto W. De Luca, Putting things in context: Challenge on context-aware movie recommendation, in: *Proceedings of the Workshop on Context-Aware Movie Recommendation, CAMRa'10*, ACM, New York, NY, USA, 2010, pp. 2–6.
- [47] Ruslan Salakhutdinov, Andriy Mnih, Probabilistic matrix factorization, in: *Neural Information Processing Systems (NIPS)*, 2008, pp. 1257–1264.
- [48] Badrul M. Sarwar, George Karypis, Joseph A. Konstan, John T. Riedl, Application of dimensionality reduction in recommender system – A case study, in: *ACM WEBKDD WORKSHOP*, 2000.
- [49] Andrew I. Schein, Lawrence K. Saul, Lyle H. Ungar, A generalized linear model for principal component analysis of binary data, in: *Proceedings of the 9th International Workshop on Artificial Intelligence and Statistics*, 2003.
- [50] Luo Si, Rong Jin, Flexible mixture model for collaborative filtering, in: *International Conference on Machine Learning (ICML)*, 2003, pp. 704–711.
- [51] Vikas Sindhwani, S.S. Bucak, J. Hu, A. Mojsilovic, A family of non-negative matrix factorizations for one-class collaborative filtering, in: *RIA Workshop of ACM Conference on Recommender Systems*, 2009.
- [52] Ajit P. Singh, Geoffrey J. Gordon, Relational learning via collective matrix factorization, in: *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD'08*, ACM, New York, NY, USA, 2008, pp. 650–658.
- [53] Nathan Srebro, Tommi Jaakkola, Weighted low-rank approximations, in: *International Conference on Machine Learning (ICML)*, 2003, pp. 720–727.
- [54] Charles Sutton, Andrew McCallum, Composition of conditional random fields for transfer learning, in: *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing, HLT'05*, Association for Computational Linguistics, Stroudsburg, PA, USA, 2005, pp. 748–754.
- [55] Vishvas Vasuki, Nagarajan Natarajan, Zhengdong Lu, Berkant Savas, Inderjit Dhillon, Scalable affiliation recommendation using auxiliary networks, *ACM Trans. Intell. Syst. Technol.* 3 (1) (October 2011) 1–20.
- [56] Jiho Yoo, Seungjin Choi, Weighted nonnegative matrix co-tri-factorization for collaborative prediction, in: *Proceedings of the 1st Asian Conference on Machine Learning: Advances in Machine Learning, ACML'09*, Springer-Verlag, Berlin, Heidelberg, 2009, pp. 396–411.
- [57] Yi Zhang, Jiazhong Nie, Probabilistic latent relational model for integrating heterogeneous information for recommendation, Technical report, School of Engineering, UCSC, 2010.
- [58] Yu Zhang, Bin Cao, Dit-Yan Yeung, Multi-domain collaborative filtering, in: *Uncertainty in Artificial Intelligence (UAI)*, 2010, pp. 725–732.
- [59] Shiwang Zhao, Michelle X. Zhou, Xiatian Zhang, Quan Yuan, Wentao Zheng, Rongyao Fu, Who is doing what and when: Social map-based recommendation for content-centric social web sites, *ACM Trans. Intell. Syst. Technol.* 3 (1) (October 2011) 1–23.
- [60] Yu Zheng, Xing Xie, Learning travel recommendations from user-generated gps traces, *ACM Trans. Intell. Syst. Technol.* 2 (1) (January 2011) 1–29.