Transfer Learning to Predict Missing Ratings via Heterogeneous User Feedbacks

Weike Pan, Nathan N. Liu, Evan W. Xiang, Qiang Yang
Department of Computer Science and Engineering
Hong Kong University of Science and Technology, Hong Kong
{weikep, nliu, wxiang, qyang}@cse.ust.hk

Abstract

Data sparsity due to missing ratings is a major challenge for collaborative filtering (CF) techniques in recommender systems. This is especially true for CF domains where the ratings are expressed numerically. We observe that, while we may lack the information in numerical ratings, we may have more data in the form of binary ratings. This is especially true when users can easily express themselves with their likes and dislikes for certain items. In this paper, we explore how to use the binary preference data expressed in the form of like/dislike to help reduce the impact of data sparsity of more expressive numerical ratings. We do this by transferring the rating knowledge from some auxiliary data source in binary form (that is, likes or dislikes), to a target numerical rating matrix. Our solution is to model both numerical ratings and like/dislike in a principled way, using a novel framework of *Transfer by Collective Factorization* (TCF). In particular, we construct the shared latent space collectively and learn the data-dependent effect separately. A major advantage of the TCF approach over previous collective matrix factorization (or bi-factorization) methods is that we are able to capture the data-dependent effect when sharing the data-independent knowledge, so as to increase the overall quality of knowledge transfer. Experimental results demonstrate the effectiveness of TCF at various sparsity levels as compared to several state-of-the-art methods.

1 Introduction

Data sparsity is a major challenge in collaborative filtering methods [Goldberg *et al.*, 1992; Pan *et al.*, 2010] used in recommender systems. 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 predict the missing values. However, we observe that, some auxiliary data of the form "like/dislike" may be more easily obtained; e.g. the favored/disfavored data in Moviepilot[1], the love/ban data in Last.fm[2] and the "Want to see"/"Not Interested" data in Flixster[3]. It is more convenient for users to express such preferences instead of numerical ratings. The question we ask in this paper is: how do we take advantage of our knowledge in the form of binary ratings to alleviate the sparsity problem in numerical ratings when we build a prediction model?

To the best of our knowledge, no previous work answered this question of how to jointly model a target data of numerical ratings and an auxiliary data of like/dislike. There are some work on using both the numerical ratings and *implicit* data of "whether rated" [Koren, 2010; Liu *et al.*, 2010] or "whether purchased" [Zhang and Nie, 2010] to help boost the prediction performance. Among the previous works, Koren ([Koren, 2010]) uses implicit data of "rated" as offsets in a factorization model, Liu et al. [Liu *et al.*, 2010] adapt the collective matrix factorization (CMF) approach [Singh and Gordon, 2008] to integrate the implicit data of "rated", and Zhang et al. [Zhang and Nie, 2010] 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 using auxiliary data of both like and dislike in collaborative filtering in a transfer learning framework.

Most existing transfer learning methods in recommender systems consider auxiliary data from several perspectives, including user-side transfer [Cao *et al.*, 2010; Ma *et al.*, 2011; Vasuki *et al.*, 2012], item-side transfer [Singh and Gordon, 2008], two-side transfer [Pan *et al.*, 2010], or knowledge-transfer using related but not aligned data [Li *et al.*, 2009a; 2009b]. In this paper, we consider the situation where the auxiliary data of like/dislike is such that users and items of the target rating matrix and the auxiliary like/dislike matrix are both aligned. This gives us more precise information on the mapping between auxiliary and target data, which can lead to higher performance. Under this framework, the following questions can be addressed.

1. What to transfer and how to transfer, as raised in [Pan and Yang, 2010], can be answered. Previous works that address this question include approaches that transfer the knowledge of latent features in an adaptive way [Pan *et*

[1] http://www.moviepilot.de

[2] http://www.last.fm

[3] http://www.flixster.com

al., 2010] or collective way [Singh and Gordon, 2008], transfer cluster-level rating patterns in an adaptive manner [Li *et al.*, 2009a] or collective manner [Li *et al.*, 2009b].

2. How to model the data-dependent effect of numerical ratings and like/dislike when sharing the data-independent knowledge? This question is important since clearly the auxiliary and target data may be with different distribution and semantic meaning.

In this paper, we propose a principled matrix-based transfer-learning framework referred as *Transfer by Collective Factorization* (TCF), 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 core matrices. Technically, our main contributions include:

1. We construct a shared latent space (what to transfer) via matrix tri-factorization in a *collective* way (to address the how to transfer question).
2. We model the data-dependent effect of like/dislike and numerical ratings by learning the core matrices of tri-factorizations separately.
3. We introduce orthonormal constraints to the latent feature matrices in TCF to enforce the effect of noise reduction in singular value decomposition (SVD), and thus only transfer the most useful knowledge.

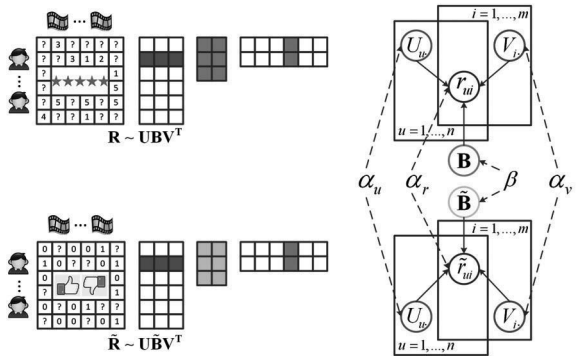

Figure 1: Graphical model of *Transfer by Collective Factorization* (TCF) for transfer learning in recommender systems.

2 Transfer by Collective Factorization

2.1 Problem Definition

In the target data, we have a 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 (unobserved value). Note, the observed rating values in $\mathbf{R}$ are not limited to 5-star grades, instead, they can be any real numbers. We use a mask 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 ($y_{ui} = 0$). Similarly, in the auxiliary data, we have a matrix $\tilde{\mathbf{R}} = [\tilde{r}_{ui}]_{n\times m} \in \{0,1,?\}^{n\times m}$ with $\tilde{q}$ observations, where 1 denotes the observed 'like' value, and 0 denotes the observed 'dislike' value. The question mark is the missing value. Similar to the target data, we have a corresponding mask matrix $\tilde{\mathbf{Y}} = [\tilde{y}_{ui}]_{n\times m} \in \{0,1\}^{n\times m}$. Note that there is an 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 knowledge from $\tilde{\mathbf{R}}$. Note that the implicit data in [Koren, 2010; Liu *et al.*, 2010; Zhang and Nie, 2010] is different in the form of $\{1,?\}$, since implicit data corresponds to positive observations only.

2.2 Model Formulation

We assume that a user u's rating on item i in the target data, r_{ui}, is generated from the user-specific latent feature vector $U_{u\cdot} \in \mathbb{R}^{1\times d_u}$, item-specific latent feature vector $V_{i\cdot} \in \mathbb{R}^{1\times d_v}$, and some data-dependent effect denoted as $\mathbf{B} \in \mathbb{R}^{d_u\times d_v}$. Note that it's different from the PMF model [Salakhutdinov and Mnih, 2008], which only contains $U_{u\cdot}$ and $V_{i\cdot}$. Our graphical model is shown in Figure 1, where $U_{u\cdot}, u = 1,\ldots,n$ and $V_{i\cdot}, i = 1,\ldots,m$ are shared to bridge two data, while $\mathbf{B}$, $\tilde{\mathbf{B}}$ are designed to capture the data-dependent information. We fix $d = d_u = d_v$ for notation simplicity in the sequel. We denote the tri-factorization in the target domain as $\mathcal{F}(\mathbf{R}\sim\mathbf{UBV}^T) = \sum_{u=1}^{n}\sum_{i=1}^{m} y_{ui}[\frac{1}{2}(r_{ui} - U_{u\cdot}\mathbf{B}V_{i\cdot}^T)^2 + \frac{\alpha_u}{2}||U_{u\cdot}||_F^2 + \frac{\alpha_v}{2}||V_{i\cdot}||_F^2] + \frac{\beta}{2}||\mathbf{B}||_F^2$, where regularization terms $||U_{u\cdot}||_F^2$, $||V_{i\cdot}||_F^2$ and $||\mathbf{B}||_F^2$ are used to avoid overfitting. Similarly, in the auxiliary data, we have $\mathcal{F}(\tilde{\mathbf{R}}\sim\mathbf{U}\tilde{\mathbf{B}}\mathbf{V}^T)$. To factorize $\mathbf{R}$ and $\tilde{\mathbf{R}}$ collectively, we obtain the following optimization problem for TCF,

$$\begin{aligned} \min_{\mathbf{U},\mathbf{V},\mathbf{B},\tilde{\mathbf{B}}} \quad & \mathcal{F}(\mathbf{R}\sim\mathbf{UBV}^T) + \lambda\mathcal{F}(\tilde{\mathbf{R}}\sim\mathbf{U}\tilde{\mathbf{B}}\mathbf{V}^T) \\ \text{s.t.} \quad & \mathbf{U},\mathbf{V} \in \mathfrak{D} \end{aligned} \tag{1}$$

where $\lambda > 0$ is a tradeoff parameter to balance the target and auxiliary data and $\mathfrak{D}$ is the range of the latent variables. $\mathfrak{D}$ can be $\mathfrak{D}_{\mathbb{R}} = \{\mathbf{U} \in \mathbb{R}^{n\times d}, \mathbf{V} \in \mathbb{R}^{m\times d}\}$ or $\mathfrak{D}_{\perp} = \mathfrak{D}_{\mathbb{R}} \cap \{\mathbf{U}^T\mathbf{U} = \mathbf{I}, \mathbf{V}^T\mathbf{V} = \mathbf{I}\}$ to resemble the effect of noise reduction in SVD [Keshavan *et al.*, 2010]. Thus we have two variants of TCF, CMTF (collective matrix tri-factorization) for $\mathfrak{D}_{\mathbb{R}}$ and CSVD (collective SVD) for $\mathfrak{D}_{\perp}$. Although 2DSVD or Tucker2 [Ding and Ye, 2005] 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 system.

To solve the optimization problem in Eq.(1), we first *collectively factorize* two data matrices of $\mathbf{R}$ and $\tilde{\mathbf{R}}$ to learn $\mathbf{U}$ and $\mathbf{V}$, and then estimate $\mathbf{B}$ and $\tilde{\mathbf{B}}$ separately. The knowledge of latent features $\mathbf{U}$ and $\mathbf{V}$ is *transferred by collective factorization* of the rating matrices $\mathbf{R}$ and $\tilde{\mathbf{R}}$, and for this reason, we call our approach *Transfer by Collective Factorization*.

2.3 Learning the TCF

Learning U and V in CMTF Given $\mathbf{B}$ and $\mathbf{V}$, we have gradient on the latent feature vector $U_{u\cdot}$ of user u,

$$\frac{\partial[\mathcal{F}(\mathbf{R}\sim\mathbf{UBV}^T) + \lambda\mathcal{F}(\tilde{\mathbf{R}}\sim\mathbf{U}\tilde{\mathbf{B}}\mathbf{V}^T)]}{\partial U_{u\cdot}} = -\boldsymbol{b}_u + U_{u\cdot}\boldsymbol{C}_u,$$

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

Input: $\mathbf{R}$, $\tilde{\mathbf{R}}$, $\mathbf{Y}$, $\tilde{\mathbf{Y}}$.
Output: $\mathbf{U}$, $\mathbf{V}$, $\mathbf{B}$, $\tilde{\mathbf{B}}$.
Step 1. Scale ratings in $\mathbf{R}$.
Step 2. Initialize $\mathbf{U}, \mathbf{V}$.
Step 3. Estimate $\mathbf{B}$ and $\tilde{\mathbf{B}}$.
repeat
 repeat
 Step 4.1.1. Fix $\mathbf{B}$ and $\mathbf{V}$, update $\mathbf{U}$ in CMTF or CSVD.
 Step 4.1.2. Fix $\mathbf{B}$ and $\mathbf{U}$, update $\mathbf{V}$ in CMTF or CSVD.
 until Convergence
Step 4.2. Fix $\mathbf{U}$ and $\mathbf{V}$, update $\mathbf{B}$ and $\tilde{\mathbf{B}}$.
until Convergence

Figure 2: The algorithm of *Transfer by Collective Factorization* (TCF).

$\lambda\tilde{y}_{ui}\tilde{r}_{ui}V_{i\cdot}\tilde{\mathbf{B}}^T$). Thus, we have an update rule similar to alternative least square (ALS) approach in [Bell and Koren, 2007],

$$U_{u\cdot} = \boldsymbol{b}_u \boldsymbol{C}_u^{-1}. \tag{2}$$

Note that Bell et al. [Bell and Koren, 2007] consider bi-factorization in a single matrix, which is different from our tri-factorization of two matrices. We can obtain the update rule for $V_{i\cdot}$ similarly.

Learning U and V in CSVD Since the constraints $\mathfrak{D}_\perp$ have similar effect of regularization, we remove the regularization terms in Eq.(1) and reach a simplified objective function $g = \frac{1}{2}||\mathbf{Y}\odot(\mathbf{R}-\mathbf{U}\mathbf{B}\mathbf{V}^T)||_F^2 + \frac{\lambda}{2}||\tilde{\mathbf{Y}}\odot(\tilde{\mathbf{R}}-\mathbf{U}\tilde{\mathbf{B}}\mathbf{V}^T)||_F^2$, where the variables $\mathbf{U}$ and $\mathbf{V}$ can be learned via gradient descent on the Grassmann manifold [Edelman *et al.*, 1999; Buono and Politi, 2004],

$$\mathbf{U} \leftarrow \mathbf{U} - \gamma(\mathbf{I}-\mathbf{U}\mathbf{U}^T)\frac{\partial g}{\partial \mathbf{U}} = \mathbf{U} - \gamma\nabla\mathbf{U}, \tag{3}$$

where $\frac{\partial g}{\partial \mathbf{U}} = (\mathbf{Y}\odot(\mathbf{U}\mathbf{B}\mathbf{V}^T-\mathbf{R}))\mathbf{V}\mathbf{B}^T + \lambda(\tilde{\mathbf{Y}}\odot(\mathbf{U}\tilde{\mathbf{B}}\mathbf{V}^T-\tilde{\mathbf{R}}))\mathbf{V}\tilde{\mathbf{B}}^T$ and $\gamma = \frac{-\mathrm{tr}(t_1^T t_2)-\lambda\mathrm{tr}(\tilde{t}_1^T\tilde{t}_2)}{\mathrm{tr}(t_2^T t_2)+\lambda\mathrm{tr}(\tilde{t}_2^T\tilde{t}_2)}$ with $t_1 = \mathbf{Y}\odot(\mathbf{R}-\mathbf{U}\mathbf{B}\mathbf{V}^T)$, $\tilde{t}_1 = \tilde{\mathbf{Y}}\odot(\tilde{\mathbf{R}}-\mathbf{U}\tilde{\mathbf{B}}\mathbf{V}^T)$, and $t_2 = \mathbf{Y}\odot(\nabla\mathbf{U}\mathbf{B}\mathbf{V}^T)$, $\tilde{t}_2 = \tilde{\mathbf{Y}}\odot(\nabla\mathbf{U}\tilde{\mathbf{B}}\mathbf{V}^T)$. Note that [Buono and Politi, 2004; Keshavan *et al.*, 2010] study a single-matrix factorization problem and adopt a different learning algorithm on the Grassmann manifold for searching γ. We can obtain the update rule for $\mathbf{V}$ similarly.

Learning 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,

$$\mathcal{F}(\mathbf{R}\sim\mathbf{U}\mathbf{B}\mathbf{V}^T) \propto \frac{1}{2}||\mathbf{Y}\odot(\mathbf{R}-\mathbf{U}\mathbf{B}\mathbf{V}^T)||_F^2 + \frac{\beta}{2}||\mathbf{B}||_F^2,$$

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

$$\min_{\boldsymbol{w}} \frac{1}{2}||\boldsymbol{r}-\mathbf{X}\boldsymbol{w}||_F^2 + \frac{\beta}{2}||\boldsymbol{w}||_F^2$$

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

$$\boldsymbol{w} = (\mathbf{X}^T\mathbf{X} + \beta\mathbf{I})^{-1}\mathbf{X}^T\boldsymbol{r}. \tag{4}$$

Note that $\mathbf{B}$ or $\boldsymbol{w}$ can be considered as a linear compact operator [Abernethy *et al.*, 2009] and solved efficiently using various existing off-the-shelf tools.

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

Each of the above sub-steps of updating $\mathbf{B}$, $\tilde{\mathbf{B}}$, $\mathbf{U}$ and $\mathbf{V}$ will monotonically decrease the objective function in Eq.(1), and hence ensure convergence to local minimum. The time complexity of TCF and other baseline methods (see Section 3) are: (a) AF: $O(q)$, (b) PMF: $O(Kqd^2 + K\max(n,m)d^3)$, (c) CMF: $O(K\max(q,\tilde{q})d^2 + K\max(n,m)d^3)$, (d) TCF: $O(K\max(q,\tilde{q})d^3 + Kd^6)$, where K is the iteration number, $q, \tilde{q}$ ($q, \tilde{q} > n, m$) is the number of non-zeno entries in the matrix $\mathbf{R}$ and $\tilde{\mathbf{R}}$, respectively, and d is the number of latent features. Note that TCF can be sped up via stochastic sampling or distributed computing.

3 Experimental Results

3.1 Data Sets and Evaluation Metric

We evaluate the proposed method using two movie rating data sets, Moviepilot and Netflix[4], and compare to some state-of-the-art baseline algorithms.

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 [Said *et al.*, 2010]. The data set used in the experiments is constructed as follows,

1. we first randomly extract a $2,000 \times 2,000$ dense rating matrix $\mathfrak{R}$ from the Moviepilot data, and 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. $T_R, T_E \subset \{(u, i, r_{ui}) \in \mathbb{N}\times\mathbb{N}\times[1,5] | 1 \le u \le n, 1 \le i \le m\}$. 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 levels of 0.2%, 0.4%, 0.6% and 0.8% correspondingly;

[4] http://www.netflix.com

3. we get the auxiliary data $\tilde{\mathbf{R}}$ (sparsity 2%) from favoured/disfavoured records of users expressed on movies. The overlap between $\tilde{\mathbf{R}}$ and $\mathbf{R}$ ($\sum_{i,j} y_{ij}\tilde{y}_{ij}/n/m$) is 0.035%, 0.070%, 0.10% and 0.14% correspondingly.

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 first randomly extract a $5,000 \times 5,000$ dense rating matrix $\mathfrak{R}$ from the Netflix data;
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 heterogenous auxiliary and target data, we adopt the pre-processing approach [Sindhwani *et al.*, 2009] 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 $\mathbf{R}$ ($\sum_{i,j} y_{ij}\tilde{y}_{ij}/n/m$) is 0.035%, 0.071%, 0.11% and 0.14% correspondingly.

The final data sets are summarized in Table 1.

Table 1: Description of Moviepilot (MP) data ($n = m = 2000$) and Netflix (NF) data ($n = m = 5000$).

Data set		Form	Sparsity
MP	target (training)	$[1, 5] \cup \{?\}$	$< 1\%$
	target (test)	$[1, 5] \cup \{?\}$	11.4%
	auxiliary	{0,1,?}	2%
NF	target (training)	{1,2,3,4,5,?}	$< 1\%$
	target (test)	{1,2,3,4,5,?}	11.3%
	auxiliary	{0,1,?}	2%

Evaluation Metric We adopt the evaluation metric of Mean Absolute Error (MAE),

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

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 three random trials when generating the required number of observed ratings from T_R, and averaged results are reported. The results on RMSE are similar.

3.2 Baselines and Parameter Settings

We compare our TCF method with two non-transfer learning methods: the average filling method (AF), PMF [Salakhutdinov and Mnih, 2008], as well as one transfer learning method: CMF [Singh and Gordon, 2008].

For the average filling (AF) method, we use the empirically best approach [Pan *et al.*, 2010], $\hat{r}_{ui} = \bar{r} + b_{u\cdot} + b_{\cdot i}$, where $\bar{r} = \sum_{u,i} y_{ui} r_{ui} / \sum_{u,i} y_{ui}$ is the global average rating, $b_{u\cdot} = \sum_i y_{ui}(r_{ui} - \bar{r}_{\cdot i})/\sum_i y_{ui}$ is the bias of user u, and $b_{\cdot i} = \sum_u y_{ui}(r_{ui} - \bar{r}_{u\cdot})/\sum_u y_{ui}$ is the bias of item i. For PMF, CMF 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 CMF, 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 $\sigma(U_{u\cdot}V_{i\cdot}^T)$ was embedded in the auxiliary data matrix factorization of CMF, where $\sigma(x) = \frac{1}{1+e^{-\gamma(x-0.5)}}$, and different parameters $\gamma \in \{1, 10, 20\}$ are tried.

3.3 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 parameters and convergence condition for PMF, CMF and TCF. The results on test data (unavailable during training) are reported in Table 2. We can make the following observations:

1. TCF performs significantly better than all other baselines at all sparsity levels;
2. For the transfer learning method of CMF, we can see that it is significantly better than PMF 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 (1) the heterogeneity of the auxiliary binary rating data and target numerical rating data, and (2) the usefulness of smoothing (AF) for sparse data;
3. For the transfer learning methods of CMF and CMTF, we can see that CMTF performs better than CMF in all cases, which shows the advantages of modeling the data-dependent effect in CMTF.
4. For the two variants of TCF, we can see that introducing orthonormal constraints (CSVD) improves the performance over CMTF in all cases, which shows the effect of noise reduction, and thus selectively transfer the most useful knowledge from the auxiliary data.

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 Netflix. The results are shown in Figure 3. 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.

4 Related Works

PMF Probabilistic matrix factorization (PMF) [Salakhutdinov and Mnih, 2008] is a recently proposed method for missing value prediction in a *single* matrix. The RSTE model [Ma

Table 2: Prediction performance on Moviepilot and Netflix of average filling (AF), probabilistic matrix factorization (PMF), collective matrix factorization with logistic link function (CMF-link), and two variants of *Transfer by Collective Factorization*, CMTF (TCF) and CSVD (TCF). Numbers in boldface (i.e. **0.7087**) and in Italic (i.e. *0.7415*) are the best and second best results among all methods, respectively.

Data set	Sparsity of $\mathbf{R}$ (*Observed* tr. #, val. #)	Without transfer		With transfer		
		AF	PMF	CMF-link	CMTF (TCF)	CSVD (TCF)
Moviepilot	0.2% (tr. 3, val. 1)	0.7942±0.0047	0.8118±0.0014	0.9956±0.0149	*0.7415*±0.0018	**0.7087**±0.0035
	0.4% (tr. 7, val. 1)	0.7259±0.0022	0.7794±0.0009	0.7632±0.0005	*0.7021*±0.002	**0.6860**±0.0023
	0.6% (tr. 11, val. 1)	0.6956±0.0017	0.7602±0.0009	0.7121±0.0007	*0.6871*±0.0013	**0.6743**±0.0048
	0.8% (tr. 15, val. 1)	0.6798±0.0010	0.7513±0.0005	0.6905±0.0007	*0.6776*±0.0006	**0.6612**±0.0028
Netflix	0.2% (tr. 9, val. 1)	0.7765±0.0006	0.8879±0.0008	0.7994±0.0017	*0.7589*±0.0175	**0.7405**±0.0007
	0.4% (tr. 19, val. 1)	0.7429±0.0006	0.8467±0.0006	0.7508±0.0008	*0.7195*±0.0055	**0.7080**±0.0002
	0.6% (tr. 29, val. 1)	0.7308±0.0005	0.8087±0.0188	0.7365±0.0004	*0.7031*±0.0005	**0.6948**±0.0007
	0.8% (tr. 39, val. 1)	0.7246±0.0003	0.7642±0.0003	0.7295±0.0003	*0.6962*±0.0009	**0.6877**±0.0007

et al., 2011] generalizes PMF and factorizes a *single* rating matrix with a regularization term from the *user-side* social data. The PLRM model [Zhang and Nie, 2010] generalizes PMF 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 that only considers numerical ratings and *implicit feedback* can be considered as a special case of our TCF framework, CMTF for $\mathfrak{D} = \mathfrak{D}_{\mathbb{R}}$, but the learning algorithm is different (CMTF has closed-form solutions for all steps). CSVD (with $\mathfrak{D} = \mathfrak{D}_{\perp}$) performs better than CMTF via selectively transferring the most useful knowledge.

CMF Collective matrix factorization (CMF) [Singh and Gordon, 2008] is proposed for jointly factorizing two matrices with the constraints of sharing *one-side* (user or item) latent features. However, in our problem setting as shown in Figure 1, both users and items are aligned. To alleviate the data heterogeneity in CMF, we embed a logistic link function in the auxiliary data matrix factorization in our experiments.

DPMF Dependent probabilistic matrix factorization (DPMF) [Adams *et al.*, 2010] 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 latent features.

CST Coordinate system transfer (CST) [Pan *et al.*, 2010] 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.

Parallel to the PMF family of CMF and DPMF, there is a corresponding NMF [Lee and Seung, 2001] family with *non-negative* constraints:

1. Tri-factorization method of WNMCTF [Yoo and Choi, 2009] is proposed to factorize three matrices of *user-item*, *item-content* and *user-demographics*, and
2. Codebook sharing methods of CBT [Li *et al.*, 2009a] and RMGM [Li *et al.*, 2009b] share cluster-level rating patterns of two rating matrices.

Models in the NMF family usually have better interpretability, while the top ranking models [Koren, 2010] in collaborative filtering are from the PMF family. We summarize the above related work in Table 3, in the perspective of whether having non-negative constraints on the latent variables, and what & how to transfer in transfer learning [Pan and Yang, 2010].

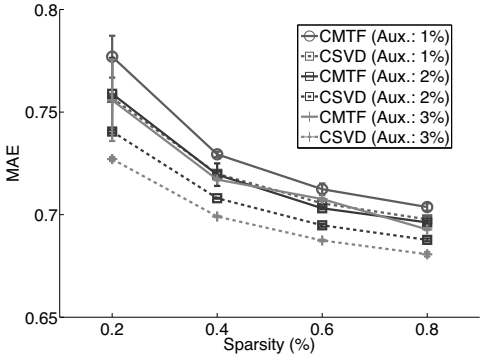

Figure 3: Prediction performance of TCF (CMTF, CSVD) on Netflix at different sparsity levels with different auxiliary data.

5 Conclusions and Future Work

In this paper, we presented a novel transfer learning framework, *Transfer by Collective Factorization* (TCF), to transfer knowledge from auxiliary data of explicit binary ratings (like and dislike), which alleviates the data sparsity problem in numerical ratings. Our method constructs the shared latent space $\mathbf{U}, \mathbf{V}$ in a collective manner, captures the data-dependent effect via learning core matrices $\mathbf{B}, \tilde{\mathbf{B}}$ separately, and selectively transfer the most useful knowledge via noise reduction by introducing orthonormal constraints. The novelty of our algorithm includes generalizing transfer learning methods in recommender systems in a principled way. Experimental results show that TCF performs significantly better than several state-of-the-art baseline algorithms at various sparsity levels.

In the future, we will extend the TCF framework to include more theoretical analysis and large-scale experiments.

Table 3: Summary of related work on transfer learning in recommender systems.

	Knowledge (what to transfer)	Algorithm style (how to transfer) Adaptive	Collective
PMF [Salakhutdinov and Mnih, 2008] family	Covariance		DPMF [Adams *et al.*, 2010]
	Latent features	CST [Pan *et al.*, 2010]	CMF [Singh and Gordon, 2008], **TCF**
NMF [Lee and Seung, 2001] family	Codebook	CBT [Li *et al.*, 2009a]	RMGM [Li *et al.*, 2009b]
	Latent features		WNMCTF [Yoo and Choi, 2009]

Acknowledgement

We thank the support of Hong Kong RGC/NSFC N_HKUST624/09 and Hong Kong RGC grant 621010.

References

[Abernethy *et al.*, 2009] Jacob Abernethy, Francis Bach, Theodoros Evgeniou, and Jean-Philippe Vert. A new approach to collaborative filtering: Operator estimation with spectral regularization. *JMLR*, 10:803–826, 2009.

[Adams *et al.*, 2010] Ryan P. Adams, George E. Dahl, and Iain Murray. Incorporating side information into probabilistic matrix factorization using Gaussian processes. In *UAI*, pages 1–9, 2010.

[Bell and Koren, 2007] Robert M. Bell and Yehuda Koren. Scalable collaborative filtering with jointly derived neighborhood interpolation weights. In *ICDM*, pages 43–52, 2007.

[Buono and Politi, 2004] Nicoletta Del Buono and Tiziano Politi. A continuous technique for the weighted low-rank approximation problem. In *ICCSA*, pages 988–997, 2004.

[Cao *et al.*, 2010] Bin Cao, Nathan Nan Liu, and Qiang Yang. Transfer learning for collective link prediction in multiple heterogenous domains. In *ICML*, pages 159–166, 2010.

[Ding and Ye, 2005] Chris H. Q. Ding and Jieping Ye. 2-dimensional singular value decomposition for 2d maps and images. In *SDM*, pages 32–43, 2005.

[Edelman *et al.*, 1999] Alan Edelman, Tomás A. Arias, and Steven T. Smith. The geometry of algorithms with orthogonality constraints. *SIAM SIMAX*, 20(2):303–353, 1999.

[Goldberg *et al.*, 1992] David Goldberg, David Nichols, Brian M. Oki, and Douglas Terry. Using collaborative filtering to weave an information tapestry. *CACM*, 35(12):61–70, 1992.

[Keshavan *et al.*, 2010] Raghunandan H. Keshavan, Andrea Montanari, and Sewoong Oh. Matrix completion from noisy entries. *JMLR*, 99:2057–2078, 2010.

[Koren, 2010] Yehuda Koren. Factor in the neighbors: Scalable and accurate collaborative filtering. *ACM TKDD*, 4(1):1:1–1:24, 2010.

[Lee and Seung, 2001] Daniel D. Lee and H. Sebastian Seung. Algorithms for non-negative matrix factorization. In *NIPS*, pages 556 – 562, 2001.

[Li *et al.*, 2009a] Bin Li, Qiang Yang, and Xiangyang Xue. Can movies and books collaborate? cross-domain collaborative filtering for sparsity reduction. In *IJCAI*, pages 2052–2057, 2009.

[Li *et al.*, 2009b] Bin Li, Qiang Yang, and Xiangyang Xue. Transfer learning for collaborative filtering via a rating-matrix generative model. In *ICML*, pages 617–624, 2009.

[Liu *et al.*, 2010] Nathan N. Liu, Evan W. Xiang, Min Zhao, and Qiang Yang. Unifying explicit and implicit feedback for collaborative filtering. In *CIKM*, pages 1445–1448, 2010.

[Ma *et al.*, 2011] Hao Ma, Irwin King, and Michael R. Lyu. Learning to recommend with explicit and implicit social relations. *ACM TIST*, 2(3), 2011.

[Pan and Yang, 2010] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *TKDE*, 22(10):1345–1359, 2010.

[Pan *et al.*, 2010] Weike Pan, Evan W. Xiang, Nathan N. Liu, and Qiang Yang. Transfer learning in collaborative filtering for sparsity reduction. In *AAAI*, pages 230–235, 2010.

[Said *et al.*, 2010] Alan Said, Shlomo Berkovsky, and Ernesto W. De Luca. Putting things in context: Challenge on context-aware movie recommendation. In *RecSys: CAMRa*, pages 2–6, 2010.

[Salakhutdinov and Mnih, 2008] Ruslan Salakhutdinov and Andriy Mnih. Probabilistic matrix factorization. In *NIPS*, pages 1257–1264, 2008.

[Sindhwani *et al.*, 2009] Vikas Sindhwani, S.S. Bucak, J. Hu, and A. Mojsilovic. A family of non-negative matrix factorizations for one-class collaborative filtering. In *RecSys: RIA*, 2009.

[Singh and Gordon, 2008] Ajit P. Singh and Geoffrey J. Gordon. Relational learning via collective matrix factorization. In *KDD*, pages 650–658, 2008.

[Vasuki *et al.*, 2012] Vishvas Vasuki, Nagarajan Natarajan, Zhengdong Lu, Berkant Savas, and Inderjit Dhillon. Scalable affiliation recommendation using auxiliary networks. *ACM TIST*, 2012.

[Yoo and Choi, 2009] Jiho Yoo and Seungjin Choi. Weighted nonnegative matrix co-tri-factorization for collaborative prediction. In *ACML*, pages 396–411, 2009.

[Zhang and Nie, 2010] Yi Zhang and Jiazhong Nie. Probabilistic latent relational model for integrating heterogeneous information for recommendation. Technical report, School of Engineering, UCSC, 2010.