A Cross-Domain Recommender System With Kernel-Induced Knowledge Transfer for Overlapping Entities

Qian Zhang, *Member, IEEE*, Jie Lu, *Fellow, IEEE*, Dianshuang Wu, *Member, IEEE*, and Guangquan Zhang

***Abstract*— The aim of recommender systems is to automatically identify user preferences within collected data, then use those preferences to make recommendations that help with decisions. However, recommender systems suffer from data sparsity problem, which is particularly prevalent in newly launched systems that have not yet had enough time to amass sufficient data. As a solution, cross-domain recommender systems transfer knowledge from a source domain with relatively rich data to assist recommendations in the target domain. These systems usually assume that the entities either fully overlap or do not overlap at all. In practice, it is more common for the entities in the two domains to partially overlap. Moreover, overlapping entities may have different expressions in each domain. Neglecting these two issues reduces prediction accuracy of cross-domain recommender systems in the target domain. To fully exploit partially overlapping entities and improve the accuracy of predictions, this paper presents a cross-domain recommender system based on kernel-induced knowledge transfer, called KerKT. Domain adaptation is used to adjust the feature spaces of overlapping entities, while diffusion kernel completion is used to correlate the non-overlapping entities between the two domains. With this approach, knowledge is effectively transferred through the overlapping entities, thus alleviating data sparsity issues. Experiments conducted on four data sets, each with three sparsity ratios, show that KerKT has 1.13%–20% better prediction accuracy compared with six benchmarks. In addition, the results indicate that transferring knowledge from the source domain to the target domain is both possible and beneficial with even small overlaps.**

***Index Terms*— Collaborative filtering, cross-domain recommender systems, knowledge transfer, recommender systems.**

I. Introduction

RECOMMENDER systems have developed quickly with the explosion of Web 2.0 technologies [1] and are now in wide use. The aim of recommender systems is to provide users with items, such as products or services, that match their preferences. Generally, recommendation techniques are roughly divided into three categories based on the underlying data used to make the recommendation: collaborative filtering-based [2], content-based [3], and knowledge-based recommendation [4].

Manuscript received March 14, 2018; revised June 21, 2018; accepted October 5, 2018. This work was supported by the Australian Research Council (ARC) under Discovery Grant DP170101632. *(Corresponding author: Jie Lu.)*

The authors are with the Decision Systems and e-Service Intelligence Laboratory, Center for Artificial Intelligence, Faculty of Engineering and Information Technology, University of Technology Sydney, Sydney, NSW 2007, Australia (e-mail: qian.zhang-1@uts.edu.au; jie.lu@uts.edu.au; dianshuang.wu@uts.edu.au; guangquan.zhang@uts.edu.au).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNNLS.2018.2875144

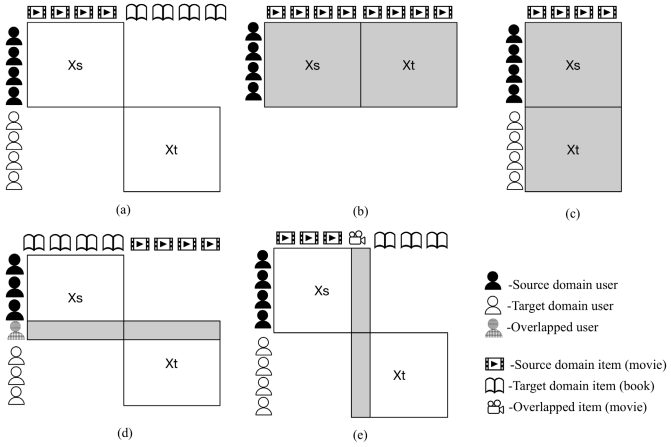


Fig. 1. Different scenarios of overlapping entities. (a) Nonoverlapping. (b) User fully overlapping. (c) Item fully overlapping. (d) User partly overlapping. (e) Item partly overlapping.

Collaborative filtering generates recommendations to one user from the historical records of other users with similar behavior [5]. This approach has advantages when historical data for users and items are actually available, such as ratings or browsing data. Originally designed as basic memory-based methods, collaborative filtering has evolved into model-based methods that commonly involve machine learning techniques, such as matrix factorization [6], probabilistic models [7], and deep neural networks [8]–[10]. Matrix factorization is, perhaps, the most widely used and has been incorporated into many commercial recommender systems [11]. However, observing the interactions between users and items has limitations in practice, and among them, data sparsity is a common problem [12]—a problem that is particularly severe and challenging in newly launched recommender systems. Data sparsity greatly impairs a recommender system's ability to produce accurate recommendation results, which leads to a poor experience for users [13].

To overcome data sparsity issues, some recommender systems based on collaborative filtering are beginning to incorporate transfer learning. Transfer learning extracts shared knowledge from a domain with comparatively denser data [14] and uses that knowledge to improve recommendations in the target domain. In newly launched recommender systems, this technique can significantly improve the performance [15]. Systems that use transfer learning techniques are known as cross-domain recommender systems. These systems are specifically designed to provide recommendations in the target

domain using information extracted from the source domain. However, the most crucial concern in cross-domain recommender systems is how to extract common knowledge that can be shared between the two domains.

The methods for knowledge extraction and transfer are different depending on whether and how the entities in each domain overlap. Existing cross-domain recommender systems usually assume that either none of the entities are common to both domains, or they all are with full one-to-one mapping. Nonoverlapping methods tend to extract shared knowledge based on collective group-level user behavior. Although many of these methods have been designed to suit specific situations, they cannot integrate knowledge from an overlapping entity once new information becomes available. In fully overlapping methods, the original source and target rating matrices are collectively factorized; then, the entities' features are extracted. Constraints on each entity ensure that these features are exactly the same in the source and target domains so they can act as a bridge for knowledge transfer. However, practical situations rarely satisfy the "all" or "none" overlap assumption; rather, they fall somewhere in between as shown in Fig. 1. In fully overlapping methods, information about the overlapping entities is used to establish constraints between two domains. These constraints usually relate to the entities' features. However, even with the same user, there may be small differences in item rating patterns between two different domains, and this is called domain divergence. If the entity feature constraints are not handled delicately, knowledge transfer will suffer and reduce the accuracy of the predictions.

Hence, cross-domain recommendation systems present the following challenges.

1) *Feature Inconsistency Caused by Data Sparsity:* Typically, there are no explicit features, only extracted latent features. And the observed sparse ratings do not fully represent a user's preferences, so features extracted from the same user in two different domains will be inconsistent. Thus, constructing an appropriate feature space is very challenging.
2) *Feature Inconsistency Caused by Domain Heterogeneity:* Extracted latent features from an overlapping entity can be aligned through domain adaptation techniques. But extracted latent features from nonoverlapping entities lack a direct correlation, and their features are heterogeneous.
3) *Partially Overlapping Entities:* The number of overlapping entities can account for a very small part of the total number of entities in the target domain. Whether a small number of overlapping entities is effective in transferring knowledge or not and what the shared constraints should be, remain unsolved problems.

In this paper, we propose a cross-domain recommender system with kernel-induced knowledge transfer (KerKT) as a knowledge transfer method to improve recommendation performance with partially overlapping entities. We first factorize the rating matrices separately to construct the user and item feature matrices. To avoid divergence in the feature space caused by data sparsity, we propose a domain adaptation method to adjust the feature spaces through the overlapping entities. Then, we use a diffusion kernel to construct a full and complete entity similarity matrix, so the similarity measures can be used in heterogeneous settings. Finally, we use a more flexible constraint to jointly factorize the source and target rating matrices. The main contributions of this paper are as follows.

1) A domain adaptation method that aligns the feature spaces of overlapping entities. The method matches the features of each overlapping entity acquired from two different domains. The overlapping entities are projected onto the same subspace to ensure the consistency between the representations. Feature divergence caused by data sparsity is eliminated.
2) A kernel-induced completion method for computing entity similarities in heterogeneous situations. Feature divergence caused by domain heterogeneity is eliminated and domain connection is reinforced. This means the similarities between entities can be determined through a modest amount of overlapping entity data.
3) A new matrix factorization method with constraints that integrates the intradomain and interdomain entity correlations acquired from overlapping entities. The two rating matrices are collectively factorized sharing interdomain knowledge while retaining their own domain-specific characteristics. The constraints are more flexible than previous methods and ensure that more useful knowledge is transferred to the target domain.
4) An adaptive knowledge transfer method, called KerKT, that addresses partially overlapping entities—the most common scenario in practice. Extensive experiments were conducted on four real-world data sets, each with three different sparsity ratios. The results show that KerKT alleviates the impact on recommendation caused by data sparsity and transfers knowledge even when there are only a few overlapping entities.

The remainder of this paper is organized as follows. A review of work related to cross-domain recommender systems is provided in Section II. Section III introduces the preliminaries and formally defines the problem to be solved. In Section IV, we present the KerKT method. Section V contains the empirical experiments. We evaluated four tasks on four real-world data sets with three data sparsity ratios and three different levels of overlapping entities. The results show that our method performs better in prediction accuracy than six existing nontransfer and cross-domain methods. Finally, the conclusion and directions for future study are provided in Section VI.

II. Related Work

In this section, we review the related work on both kernel-based and cross-domain recommender systems.

A. Kernel-Based Recommender Systems

In recommender systems, nonlinear interactions between users and items are modeled using kernels. Integrating kernels into a matrix factorization framework, which is a linear combination of the inner product of a user factor matrix and

an item factor matrix, provides a more general and more flexible method for updating online models [16]. Lawrence and Urtasun [17] adopted a Mercer kernel to a nonlinear Gaussian process model. Ghazanfar *et al.* [18] incorporated metadata, such as genres and descriptions into the matrix factorization framework, along with kernels to solve cold-start problems. Coifman and Lafon [19] used diffusion maps to find representations of data with geometric meanings. Diffusion kernels are a special class of exponential kernels based on the heat equation, which aim to measure the similarities between vertexes or nodes when applied to graphs [20]. However, graph-based diffusion kernel completion is seldom used when dealing with recommendation issues.

B. Cross-Domain Recommender Systems

Cross-domain recommender systems can be divided into three categories.

1) Cross-Domain Recommender Systems With Side Information: In this category, it is assumed that some side information about the entities is available. Collective matrix factorization (CMF) [21] is designed for scenarios where a user–item rating matrix and an item-attribute matrix for the same group of items are available. The two matrices are collectively factorized by sharing the item parameters since the items are the same. The Tagicofi method [22] uses the user–item rating matrix and the user-tag matrix for the same group of users. User similarities extracted from shared tags are used to assist matrix factorization of the original rating matrix. On this basis, TagCDCF [23] extends the Tagicofi method to two domain scenarios, each containing the data of those two matrices by integrating the introdomain and interdomain correlations and matrix factorization simultaneously. In addition to using user-generated tags, hybrid random walk [24] bridges cross-domain knowledge through social information.

2) Cross-Domain Recommender Systems With Nonoverlapping Entities: This category covers the methods that handle two domains with nonoverlapping entities and transfer knowledge at the group level. Users and items are clustered into groups, and knowledge is shared through group-level rating patterns [25]. For example, codebook transfer (CBT) [26] clusters users and items into groups and extracts group-level knowledge as a "codebook." A probabilistic model, called the rating matrix generated model (RMGM) [27], was subsequently extended from CBT, relaxing the hard membership in groups to soft membership. Neither of these methods ensures that the shared information between two groups in different domains is consistent, so the effectiveness of the knowledge transfer is not guaranteed. Consistent information transfer [28] relies on a domain adaptation technique to extract consistent knowledge from the source domain. This method is superior, especially when the data statistics between the source and target domains are divergent.

3) Cross-Domain Recommender Systems With Fully Overlapping Entities: These systems assume that the source and target domains share some common entities. These overlapping entities are used as a bridge, with constraints, to transfer knowledge. Transfer by collective factorization (TCF) [29] was developed to use implicit data in the source domain to help predict the explicit feedback in the target domain, such as ratings. However, the assumptions in TCF are very strict—users and items must have a one-to-one mapping across the domains. So, while this method is able to deal with heterogeneous data, its strict assumptions limit the scope for these types of applications in practice. Cross-domain triadic factorization (CDTF) [30] is a user-item-domain tensor that integrates both explicit and implicit feedback. It assumes that users are fully overlapping and that the user factor matrix is the same, thus bridging the domains. Cluster-based matrix factorization (CBMF) [31] tries to extend CDTF to partially overlapping entities, but the core of the CBMF method is the same as for nonoverlapping entities, which transfers knowledge based on groups rather than using the overlapping entities as a bridge.

Since entity correspondence is not always fully available, some strategies have been developed to match users or items across two domains. Li and Lin [32] used latent space matching to identify unknown user/item mappings. Sometimes, the identifying mappings are time-consuming; hence, Zhao *et al.* [33] developed an active-learning framework to identify the most valuable correspondences between entities. The process of identifying entity correspondence is not included in this paper. In summary, methods developed specifically for partially overlapping entities are rare. In this paper, we introduce KerKT to fill the gap in the literature between fully overlapping and nonoverlapping scenarios.

III. Preliminaries and Problem Formulation

In this section, a matrix factorization view of the recommender system in one domain is given to clearly describe the problem setting. The problem under study in this paper is then formulated.

A. Recommendation Task Based on Matrix Factorization in One Domain

Suppose there are M users and N items in one domain, the relationship between users and items is given as $\boldsymbol{X} \in \mathbb{R}^{M\times N}$ (bold letter represents a matrix). If a user's preferences are represented as ratings, then $\boldsymbol{X}$ is a rating matrix where $\boldsymbol{X}$ is subject to $X_{ij} \in \{1, 2, 3, 4, 5, ?\}$ ("?" denotes a missing value). By minimizing its Euclidean distance to the original rating matrix $\boldsymbol{X}$ [34], $\boldsymbol{X}$ is approximated by

$$\hat{\boldsymbol{X}} = \boldsymbol{U}\boldsymbol{V}^T. \tag{1}$$

Thus, $\boldsymbol{U} \in \mathbb{R}^{M\times K}$ is the user feature matrix and $\boldsymbol{V} \in \mathbb{R}^{N\times K}$ is the item feature matrix, which are two low-rank matrices for users and items, respectively. The ith user and the jth item are represented by the ith and jth row of the two matrices as U_{i*} and V_{j*}. After matrix factorization, the users and items are mapped to a latent factor feature space of a lower dimensionality K.

The recommendation task is to predict the missing values in the rating matrix based on historical records of the users' preferences. Since the rating matrix $\boldsymbol{X}$ is usually extremely sparse, it is easy to overfit the low-rank approximation matrix factorization. Regularization is usually used on low-rank feature

matrices to avoid this problem. In general, the optimization problem is

$$\min_{\boldsymbol{U},\boldsymbol{V}} \mathcal{L}(f(\boldsymbol{U},\boldsymbol{V}),\boldsymbol{X}) + \lambda\mathcal{R}(\boldsymbol{U},\boldsymbol{V}) \tag{2}$$

where $\mathcal{L}$ is the loss function of the predicted ratings $f(\boldsymbol{U},\boldsymbol{V})$ and the original ratings $\boldsymbol{X}$, $\mathcal{R}(\boldsymbol{U},\boldsymbol{V})$ is the regularization term, and $\lambda \geqslant 0$ is the regularization tradeoff parameter. Similar to probabilistic matrix factorization (PMF), the objective function to measure the loss with regularization terms and a Frobenius norm is [35]

$$J(\boldsymbol{U},\boldsymbol{V}) = \frac{1}{2}\|\boldsymbol{I}\circ(\boldsymbol{X}-\boldsymbol{U}\boldsymbol{V}^T)\|_F + \frac{\lambda}{2}\|\boldsymbol{U}\|_F + \frac{\lambda}{2}\|\boldsymbol{V}\|_F \tag{3}$$

where $\boldsymbol{I}$ is the rating indicator matrix, $I_{ij} \in \{0,1\}$. $I_{ij} = 1$ indicates that the rating is observed, or $I_{ij} = 0$, otherwise. $\circ$ denotes the Hadamard product of the matrices.

B. Problem Definition

The problem in this paper is based on the assumption that ratings in the target domain are very sparse. This raises the question of how to use relatively dense data in the source domain to assist a recommendation task in the target domain with overlapping entities. In practice, corresponding entities are not usually easy to identify. Typically, there are many unique entities between different data sets or platforms and only a few common entities. Thus, in this problem setting, the entities partially overlap. Only a small proportion of the entities in the target domain matrix $\boldsymbol{X}_t$ have observed correspondences in the source rating matrix $\boldsymbol{X}_s$. Even though the entities represent the same user and/or item, the rating a user has given or the rating an item has received can be different in each domain. The overlapping entity indicator matrix is represented by $\boldsymbol{W}^{(s,t)}$, $W_{ij}^{(s,t)} \in \{0,1\}$. $W_{ij}^{(s,t)} = 1$ indicates that the ith entity in the source domain is the same as the jth entity in the target domain, and $W_{ij}^{(s,t)} = 0$ otherwise. Without loss of generality, we require the rating rows of overlapping users to be at the top, and the corresponding users are in the same rows in both matrices. This is achieved by permuting the rows of the original rating matrices. Thus, the form of the entity indicator matrix $\boldsymbol{W}^{(s,t)}$ is

$$\boldsymbol{W}^{(s,t)} = \begin{bmatrix} \boldsymbol{I}_o & 0 \\ 0 & 0 \end{bmatrix}$$

where $\boldsymbol{I}_o$ is an identity matrix of the same dimension as the number of overlapping entities. This problem of partially overlapping entities in cross-domain recommender systems is formally defined in the following. Given a source rating matrix $\boldsymbol{X}_s \in \mathbb{R}^{M_s \times N_s}$ and a target rating matrix $\boldsymbol{X}_t \in \mathbb{R}^{M_t \times N_t}$, a cross-domain recommender system based on partially overlapping entities is to assist with recommendation task $\hat{\boldsymbol{X}}_t = \boldsymbol{U}_t\boldsymbol{V}_t^T$ through an auxiliary source rating matrix $\boldsymbol{X}_s$ and an overlapping entity indicator matrix $\boldsymbol{W}^{(s,t)}$.

IV. Cross-Domain Recommender System by Kernel-Induced Knowledge Transfer

This section introduces our KerKT method. The overlapping entities in each domain may be either users or items. For the purposes of this presentation, we have assumed the users overlap. Overlapping items are handled in the same way and have, therefore, been omitted from this paper. This section begins with an overview of the entire method, and then, each of the five steps is explained in detail, followed by a small-scale example for greater clarity.

A. KerKT Method Overview

To enable knowledge sharing between the source and target domains with overlapping users, constraints on the user feature matrices are added to the collective matrix factorization of the source and target rating matrices. Previous research assumes "identical" factor matrices for overlapping entities, but this assumption is too limiting to satisfy in practice. Instead, we have chosen to constrain the similarities between the entities in each domain as a bridge for knowledge transfer. However, while it is easy to measure the similarities between entities in the same domain, interdomain entity similarities cannot be computed directly.

The overlapping entities are mapped to the same feature space through domain adaptation techniques, while the nonoverlapping entities are connected by diffusion kernel completion. Thus, the similarities between all users in both the domains can be measured. Furthermore, constraining the user features using these similarities may lead to a better optimization result. The optimization problem is formalized as

$$\min_{\boldsymbol{U},\boldsymbol{V}} \mathcal{L}(f(\boldsymbol{U},\boldsymbol{V}),\boldsymbol{X}) + \lambda\mathcal{R}(\boldsymbol{U},\boldsymbol{V}) + \lambda_o\mathcal{R}_o(\boldsymbol{U}) \tag{4}$$

where $\mathcal{R}_o(\boldsymbol{U})$ is the regularization term for the entity similarity constraints derived from overlapping users and $\lambda_o \geqslant 0$ is the regularization tradeoff parameter.

The KerKT method consists of five steps, as shown in Fig. 2.

1) The user features and item features are extracted separately from the source and target domains, and the two sets of user features are aligned with the same feature space through overlapping users.
2) The item features are regulated according to the original rating matrices and the aligned user feature matrices.
3) The user and item feature matrices resulting from the previous two steps are used to measure the user and item similarities in one domain.
4) Kernel-induced completion is conducted to measure the interdomain user similarities.
5) The user/item features are retrained based on the constraints of the entity similarities; then, recommendations are made.

We have selected a specific algorithm to perform each step, but other suitable feature extraction or domain adaptation algorithms could be used as substitutes. The proposed domain adaptation method is contained in Steps 1 and 2, while the kernel-induced completion method is contained in Steps 3 and 4. The matrix factorization method, with constraints on both intradomain and interdomain entity correlations, is contained in Step 5.

B. KerKT Method

Our proposed KerKT method comprises five steps.

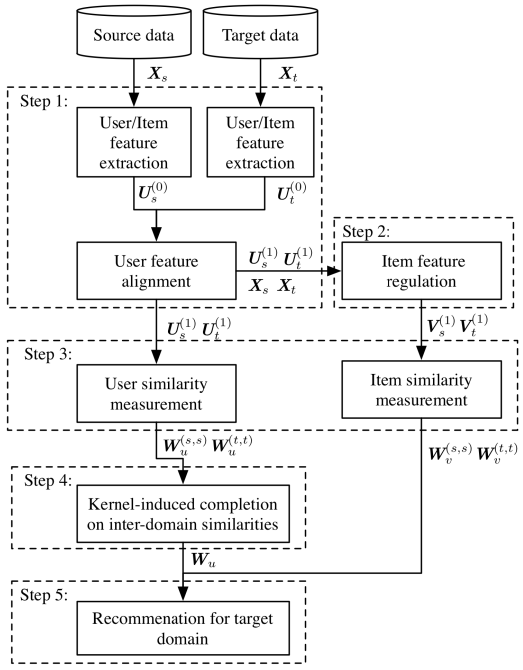


Fig. 2. Procedure of the KerKT method.

1) Step 1 (Extracting and Aligning User features in Both Domains): In this step, the source rating matrix $\boldsymbol{X}_s$ and the target rating matrix $\boldsymbol{X}_t$ are separately factorized, which results in the user feature matrix $\boldsymbol{U}_s$ for the source domain and $\boldsymbol{U}_t$ for the target domain. Recall that the users in the source and target domains partially overlap. Accordingly, each user feature matrix can be divided into two parts: one containing the overlapping users; the other containing nonoverlapping users. The overlapping user feature matrix for the source domain is denoted as $\boldsymbol{U}_{s,o}$, and $\boldsymbol{U}_{s,n}$ denotes the nonoverlapping matrix. The same goes for the target domain, i.e., $\boldsymbol{U}_{t,o}$ and $\boldsymbol{U}_{t,n}$.

Assuming that the overlapping users have similar tastes or preferences in both domains, we can use them as a bridge to transfer knowledge. However, as mentioned in Section I, even the same user's rating patterns may not be completely the same in the two different domains. Data sparsity exacerbates this condition and may lead to two different factorized user feature vectors with different physical meanings. Hence, setting the similarity of the overlapping user entities to 1 may lead to inaccurate similarity measurements, which would eventually negatively impact the effectiveness of the knowledge transfer in the following steps. Therefore, before using the entity correspondences as a strong condition, we need to ensure that the overlapping users in both the domains are represented in the same feature space. This is referred to as "subspace alignment" in transfer learning.

The aim is to map the user feature spaces of two overlapping users into a common subspace where the domain shift has been eliminated, so the overlapping users ultimately share the same feature space across both the domains. In the source domain, the jth column of the overlapping user feature matrix is the representation of the jth user feature. We use a marginal probabilistic distribution of the jth column to represent the characteristics of the user features in each matrix. Thus, the goal is to minimize the differences between the marginal probabilistic distributions of the user features for the source domain and the target domain. If the marginal probability distributions of one user feature are the same in both the domains, then the two user features are considered to have the same physical meaning. In this way, we can align the two user feature spaces. In our previous research, we provided a definition for information consistent trifactorization. However, here, since the scenario and the matrix factorization model are different, this definition has been refined into a definition for consistent matrix factorization with partially overlapping users in the following.

Definition 1 (Consistent Matrix Factorization With Partially Overlapping Users): Given a source rating matrix $\boldsymbol{X}_s \in \mathbb{R}^{M_s \times N_s}$ and a target rating matrix $\boldsymbol{X}_t \in \mathbb{R}^{M_t \times N_t}$, $\boldsymbol{X}_s$ and $\boldsymbol{X}_t$ can be factorized as follows:

$$\boldsymbol{X_s} = \begin{bmatrix} \boldsymbol{U}_{s,o} \\ \boldsymbol{U}_{s,n} \end{bmatrix} \boldsymbol{V}_s^T \tag{5}$$

$$\boldsymbol{X_t} = \begin{bmatrix} \boldsymbol{U}_{t,o} \\ \boldsymbol{U}_{t,n} \end{bmatrix} \boldsymbol{V}_t^T \tag{6}$$

where $\boldsymbol{U}_{s,o}$ and $\boldsymbol{U}_{t,o}$ are the overlapping user feature matrices in the source domain and the target domain and $\boldsymbol{U}_{s,n}$ and $\boldsymbol{U}_{t,n}$ are the nonoverlapping user feature matrices, respectively.

If both factorizations satisfy the following equation, then they are consistent matrix factorizations:

$$P(\boldsymbol{U}_{s,o}) = P(\boldsymbol{U}_{t,o}) \tag{7}$$

where $P(\boldsymbol{U}_{s,o})$ and $P(\boldsymbol{U}_{t,o})$ represent the marginal probability distribution of $\boldsymbol{U}_{s,o}$ and $\boldsymbol{U}_{t,o}$. Thus, the user feature spaces in both the source and target domains are aligned.

Solving a matrix factorization optimization problem that satisfies the above-mentioned constraints is almost impossible. However, according to Definition 1, a mapping function for those two matrices can be found to achieve the following:

$$P\big(\boldsymbol{\Psi_s}\big(\boldsymbol{U}_{s,o}^{(0)}, \boldsymbol{U}_{t,o}^{(0)}\big)\big) = P\big(\boldsymbol{\Psi_t}\big(\boldsymbol{U}_{s,o}^{(0)}, \boldsymbol{U}_{t,o}^{(0)}\big)\big). \tag{8}$$

A geodesic flow kernel (GFK) is a domain adaptation strategy to find a space that two different feature spaces can be projected into, thus eliminating the divergence of two distributions. We can use this strategy to find a mapping function to align the two user feature spaces formed by overlapping users. Once the GFK operators $\boldsymbol{\Psi_s}(\boldsymbol{U}_s^{(0)}, \boldsymbol{U}_t^{(0)})$ are determined, they can be used through the following mapping functions:

$$\boldsymbol{\Psi_s}\big(\boldsymbol{U}_s^{(0)}, \boldsymbol{U}_t^{(0)}\big) = \boldsymbol{U}_s^{(0)} \times \boldsymbol{\Psi_G}\big(\boldsymbol{U}_{s,o}^{(0)}, \boldsymbol{U}_{t,o}^{(0)}\big) \tag{9}$$

$$\boldsymbol{\Psi_t}\big(\boldsymbol{U}_s^{(0)}, \boldsymbol{U}_t^{(0)}\big) = \boldsymbol{U}_t^{(0)} \times \boldsymbol{\Psi_G}\big(\boldsymbol{U}_{s,o}^{(0)}, \boldsymbol{U}_{t,o}^{(0)}\big) \tag{10}$$

where $\boldsymbol{\Psi_G}(\boldsymbol{U}_{s,o}^{(0)}, \boldsymbol{U}_{t,o}^{(0)})$ are the GFK operators. More details can be found in [28] and [36].

With the divergence eliminated through these mappings, the new representations of the overlapping users will satisfy the conditions in Definition 1. Nonoverlapping users also need to be projected onto the same feature space. The mapping functions $\boldsymbol{\Psi_s}$ and $\boldsymbol{\Psi_t}$ are used for this purpose

$$\boldsymbol{U}_s^{(1)} = \boldsymbol{\Psi_s}\big(\boldsymbol{U}_s^{(0)}, \boldsymbol{U}_t^{(0)}\big) \tag{11}$$

$$\boldsymbol{U}_t^{(1)} = \boldsymbol{\Psi_t}\big(\boldsymbol{U}_s^{(0)}, \boldsymbol{U}_t^{(0)}\big) \tag{12}$$

where $\boldsymbol{U}_s^{(1)}$ and $\boldsymbol{U}_t^{(1)}$ are the aligned user feature matrices after mapping, and $\boldsymbol{\Psi_s}$ and $\boldsymbol{\Psi_t}$ are the mapping functions using GFK.

How the new user feature spaces are derived and how $\boldsymbol{U}_s^{(1)}$ and $\boldsymbol{U}_t^{(1)}$ are learned are summarized in Algorithm 1.

Algorithm 1 Consistent User Feature Extraction

Input:
$\boldsymbol{X}_s$, the source rating matrix;
$\boldsymbol{X}_t$, the target rating matrix;
$\boldsymbol{W}_u^{(s,t)}$, the overlapping user indicator matrix;
Output:
$\boldsymbol{U}_s^{(1)}$, the aligned user feature matrix in source domain;
$\boldsymbol{U}_t^{(1)}$, the aligned user feature matrix in target domain;
1: Factorize $\boldsymbol{X}_s$ and get user feature matrix $\begin{pmatrix} \boldsymbol{U}_{s,o}^{(0)} \\ \boldsymbol{U}_{s,n}^{(0)} \end{pmatrix}$ as in equation (3)
2: Factorize $\boldsymbol{X}_t$ and get user feature matrix $\begin{pmatrix} \boldsymbol{U}_{t,o}^{(0)} \\ \boldsymbol{U}_{t,n}^{(0)} \end{pmatrix}$ as in equation (3)
3: Obtain GFK operator $\boldsymbol{\Psi_G}(\boldsymbol{U}_{s,o}^{(0)}, \boldsymbol{U}_{t,o}^{(0)})$ as in equation (A.2) in [28]
4: Obtain mapping functions $\boldsymbol{\Psi_s}$ and $\boldsymbol{\Psi_t}$ as in equation (9)
5: **return** $\boldsymbol{U}_s^{(1)}$ and $\boldsymbol{U}_t^{(1)}$

2) Step 2 (Item Feature Regulation in Both Domains): In matrix factorization, the user feature matrix and the item feature matrix are both low-rank matrices that map users and items to the same k-dimensional feature space. So, once the user feature spaces are aligned, the item feature matrices should be regularized to the new k-dimensional feature space. The new item feature matrices are obtained by minimizing the distance between the approximations of the low-rank matrices and the original data in the rating matrix. A Frobenius norm is used to measure the distance. The cost function in source domain is as follows, the one in the target domain has the same form.

$$J_v\left(\boldsymbol{V}_s^{(1)}\right) = \frac{1}{2}\left\|\boldsymbol{I_s} \circ \left(\boldsymbol{X}_s - \boldsymbol{U}_s^{(1)}\left(\boldsymbol{V}_s^{(1)}\right)^T\right)\right\|_F + \frac{\lambda_{V_s}}{2}\left\|\boldsymbol{V}_s^{(1)}\right\|_F \tag{13}$$

where λ_{V_s} is the regularization parameter. The item feature matrices are learned by optimizing

$$\min\ J_v\left(\boldsymbol{V}_s^{(1)}\right). \tag{14}$$

Gradient descent is used for this optimization. The update rule is

$$\boldsymbol{V}_s^{(1)} \leftarrow \boldsymbol{V}_s^{(1)} - \eta_{V_s}\left[\left(\boldsymbol{U}_s^{(1)}\left(\boldsymbol{V}_s^{(1)}\right)^T - \boldsymbol{X}_s\right)\boldsymbol{U}_s^{(1)} + \lambda_{V_s}\boldsymbol{V}_s^{(1)}\right] \tag{15}$$

where the learning rate is η_{V_s}. $\boldsymbol{V}_t^{(1)}$ can be obtained through the same process.

This step is summarized in Algorithm 2.

3) Step 3 (Entity Similarity Measures in One Domain): This step calculates the user and item similarities in one domain. Since the rating matrix is very sparse, making this calculation directly from the rating matrix can lead to inaccurate results. Hence, using the PMF formulation introduced in Section III-A, one rating X_{ij} is generated from a user latent feature vector U_{i*} and an item latent feature vector V_{*j}. Thus, the source rating matrix and the target rating matrix can be factorized as $\boldsymbol{X}_s = \boldsymbol{U}_s\boldsymbol{V}_s^T$ and $\boldsymbol{X}_t = \boldsymbol{U}_t\boldsymbol{V}_t^T$. This is a dimensionality reduction and data compression process, as users/items are mapped to a lower k-dimensional feature space (usually $k \ll M, k \ll N$). Once complete, users and items are represented as full k-dimensional feature matrices, and the user/item similarities can be calculated from the user/item feature matrices.

Algorithm 2 Item Feature Regularization

Input:
$\boldsymbol{X}_s$, the source rating matrix $\boldsymbol{U}_s^{(1)}$, the user feature matrix
Output:
$\boldsymbol{V}_s^{(1)}$, the regularized item feature matrix
1: Initialize $\boldsymbol{V}_s^{(1)} \in \mathbb{R}^{N_s \times K}$, Initialize $J_v(\boldsymbol{V}_s)$ and $J_v(\boldsymbol{V}_s)^{(pre)}$
2: **while** $J_v(\boldsymbol{V}_s)^{(pre)} - J_v(\boldsymbol{V}_s) > \epsilon$ **do**
3: $J_v(\boldsymbol{V}_s)^{(pre)} = J_v(\boldsymbol{V}_s)$
4: Update $\boldsymbol{V}_s$ as in equation (15)
5: Update $J_v(\boldsymbol{V}_s)$ as in equation (13)
6: **end while**
7: **return** $\boldsymbol{V}_s^{(1)}$

Similarity measurements are easy with user and item feature spaces in one domain since the feature spaces are homogeneous. And there are many suitable choices for performing these calculations, such as cosine similarity, Pearson's similarity, Euclidean measurement, or the radial basis function (RBF) measurement. The choice depends on the situation and the characteristics of the domain. For example, cosine similarity is very popular and effective for word count and text similarity measurements due to the advantages of using angles rather than distance. Pearson's measurement tends to be more effective in memory-based collaborative filtering methods owing to its emphasis on averages. In this problem, we are measuring user similarity from a user feature matrix where the feature values are real numbers, so the following RBF measurement is the most appropriate: $W_{ij} = e^{-(\|U_{i*} - U_{j*}\|^2/\sigma^2)}$, where σ^2 is set to be the median of all the nonzero values calculated by $\|U_{i*} - U_{j*}\|^2$.

4) Step 4 (Kernel-Induced Completion of Interdomain User Similarity): In interdomain user similarity measurement, the user feature spaces are not the same and the user features are heterogeneous, which means that their similarities cannot be calculated directly. However, given the first three steps, some user similarities between the source and target domains are now known. The overlapping entity indicator matrix $\boldsymbol{W}^{(s,t)}$ contains the observed overlapping user information. Hence, a full user similarity matrix can be constructed as

$$\boldsymbol{W}_u = \begin{bmatrix} \boldsymbol{W}_u^{(s,s)}, & \boldsymbol{W}_u^{(s,t)} \\ \boldsymbol{W}_u^{(t,s)}, & \boldsymbol{W}_u^{(t,t)} \end{bmatrix} \tag{16}$$

where $\boldsymbol{W}_u^{(s,s)}$ and $\boldsymbol{W}_u^{(t,t)}$ represent the user similarities in the source and target domains, respectively, and $\boldsymbol{W}_u^{(s,t)} = (\boldsymbol{W}_u^{(t,s)})^T$ represents the interdomain user similarities.

As in Step 2, the two feature spaces of the overlapping users are aligned to eliminate feature space divergence. Therefore, it is reasonable to set the similarity of observed overlapping

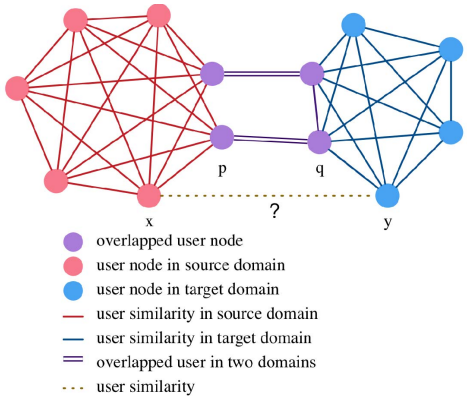


Fig. 3. Graphical view of user relationships in source and target domains.

users to $(\boldsymbol{W}_u^{(s,t)})_{ij} = 1$ in $\boldsymbol{W}^{(s,t)}$. For now, the similarities between the overlapping users are the only known entries in the interdomain similarity matrix. We need to complete $\boldsymbol{W}_u^{(s,t)}$ using the information from $\boldsymbol{W}_u$. Note that, here, the nonoverlapping users and their features are heterogeneous. Thus, their similarities cannot be computed directly.

This matrix completion problem has a strong connection to a bipartite edge completion problem [37]. In Step 3, the user similarities are all measured within one domain. As a result, we have fully connected nodes in the graph representations of both the source and target domains, as indicated by the red and blue nodes in Fig. 3. The overlapping users are shown as purple nodes in the graph, and they act as a "bridge" to couple the two graphs. To complete the user similarity matrix, $\boldsymbol{W}_u$ requires filling in all the edges from the entire graph. The subscript u has been omitted to simplify the notation.

In network propagation, a random walk is a good way to reach all the nodes. As shown in Fig. 3, one user entity, denoted as a node x in the source domain, is fully connected with all the other node in the source domain $W_{xp}^{(s,s)}$, $p \in \mathcal{U}_s$, so does node y in the target domain $W_{yq}^{(t,t)}, q \in \mathcal{U}_t$. If node p in the source domain and node q in the target domain are overlapping users, i.e., they are the same user, then $W_{pq}^{(s,t)} = 1$. The two nodes x and y are connected, and their similarity can be calculated as: $W_{xy}^{(s,t)} \leftarrow W_{xp}^{(s,s)} W_{pq}^{(s,t)} W_{yq}^{(t,t)}$. By aggregating all the nodes connected to x in the source domain and y in the target domain, the edge can be completed with: $W_{xy}^{(s,t)} \leftarrow \sum_{p \in U_s} \sum_{q \in U_t} W_{xp}^{(s,s)} W_{pq}^{(s,t)} W_{yq}^{(t,t)}$. In a matrix form, this is written as

$$\boldsymbol{W}^{(s,t),(1)} = \boldsymbol{W}^{(s,s)} \boldsymbol{W}^{(s,t)} \boldsymbol{W}^{(t,t)}. \tag{17}$$

The above-mentioned equation can be treated as a one-step random walk from both the source domain and the target domain. Generally, M steps of random walk are taken in total from the source and target sides, and all the possible steps are added together to complete the final graph

$$\boldsymbol{W}^{(s,t),(K)} = \sum_{K=0}^{M} \binom{M}{K} (\boldsymbol{W}^{(s,s)})^K \boldsymbol{W}^{(s,t)} (\boldsymbol{W}^{(t,t)})^{M-K}. \tag{18}$$

However, the goal in this problem is to find all the similarities between all the users in both the domains. Therefore, a finite number of random walk steps may not identify all the possible relationships, but it would be more likely to associate all the indirectly connected users if K was infinite. Hence, the diffusion kernel completion method [38] is used to complete the user similarity matrix

$$\boldsymbol{W}^{(s,t)} = e^{(\beta_s \boldsymbol{W}^{(s,s)})} \boldsymbol{W}^{(s,t)} e^{(\beta_t \boldsymbol{W}^{(t,t)})} \tag{19}$$

where β_s and β_t are two positive scalars to regulate the weights of the source and target domains.

5) Step 5 (Collective Matrix Factorization With User Similarity Constraints): With all similarities measured and all the pairs of nodes connected, a fully connected graph can be constructed. A very common strategy for increasing computational speed is to remove edges, leaving a sparse graph. This approach tends to achieve better performance empirically, as it emphasizes local information with high similarity while ignoring information that is likely to be false. Hence, the k-nearest neighbors of each node are retained in a similar way to the original memory-based collaborative filtering strategy.

In the scenario of this paper, only overlapping users are observed but items are nonoverlapping. The users have both intradomain and interdomain similarities, but the items only have intradomain similarities. Based on the assumption that "similar users have similar tastes and will thus choose similar items to consume," both the intradomain and interdomain similarities are used to constrain the proposed matrix factorization as prior knowledge. In terms of intradomain similarities, although the data in the target domain are very sparse, they are still very valuable for measuring the similarities between users/items so as to constrain the matrix factorization. As for interdomain similarities, users in the target domain are not only correlated with users in their own domain but also in the source domain via the overlapping users. As a consequence, users in the source domain with similar preferences to users in the target domain are transferred as knowledge to improve the performance of the recommender system.

The constraints result in users who are similar tends to have similar latent factors. We have used the regularization form in [22]

$$\mathcal{R}_o(\boldsymbol{U}) = \mathrm{tr}(\boldsymbol{U}^T \boldsymbol{L} \boldsymbol{U}) \tag{20}$$

where $\boldsymbol{L}$ denotes a Laplacian matrix and $\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{W}$. $\boldsymbol{W}$ is the user similarity matrix, and $\boldsymbol{D}$ is a diagonal matrix defined as $D_{ii} = \sum_j W_{ij}$.

Note that, although the form of regularization is quite similar, our method is different to [33]. In [33], the similarities between entities in the source domain are directly used in the target domain as prior knowledge without considering the inconsistencies between the features in each domain. By contrast, in our method, the similarities between entities in the target domain are learned through the domain adaption technique, which ensures that the features are consistent, and through the diffusion kernel technique, which makes the similarity constraints more accurate and complete. Our proposed constraints are both more flexible and more reasonable to satisfy in practice. This goal is achieved by minimizing the following objective function:

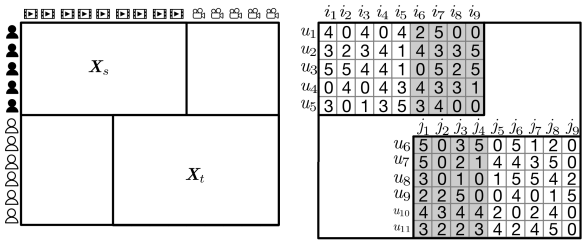


Fig. 4. Small-scale example.

$$
\begin{aligned}
&f(\boldsymbol{U}_s, \boldsymbol{V}_s, \boldsymbol{U}_t, \boldsymbol{V}_t) \\
&= \frac{\alpha}{2}\left\|\boldsymbol{I}_s \circ \left(\boldsymbol{X}_s - \boldsymbol{U}_s \boldsymbol{V}_s^T\right)\right\|_F + \frac{1}{2}\left\|\boldsymbol{I}_t \circ \left(\boldsymbol{X}_t - \boldsymbol{U}_t \boldsymbol{V}_t^T\right)\right\|_F \\
&\quad + \frac{\lambda_u}{2}\left[\mathrm{tr}\left(\boldsymbol{U}_s^T \boldsymbol{L}_u^{(s,s)} \boldsymbol{U}_s\right) + \mathrm{tr}\left(\boldsymbol{U}_s^T \boldsymbol{L}_u^{(s,t)} \boldsymbol{U}_t\right)\right. \\
&\quad \left. + \mathrm{tr}\left(\boldsymbol{U}_t^T \left(\boldsymbol{L}_u^{(s,t)}\right)^T \boldsymbol{U}_s\right) + \mathrm{tr}\left(\boldsymbol{U}_t^T \boldsymbol{L}_u^{(t,t)} \boldsymbol{U}_t\right)\right] \\
&\quad + \frac{\lambda_v}{2}\left[\mathrm{tr}\left(\boldsymbol{V}_s^T \boldsymbol{L}_v^{(s,s)} \boldsymbol{V}_s\right) + \mathrm{tr}\left(\boldsymbol{V}_t^T \boldsymbol{L}_v^{(t,t)} \boldsymbol{V}_t\right)\right] \\
&\quad + \frac{\lambda}{2}(\|\boldsymbol{U}_s\|_F + |\boldsymbol{V}_s\|_F + \|\boldsymbol{U}_t\|_F + \|\boldsymbol{V}_t\|_F) \qquad (21)
\end{aligned}
$$

where tr is the trace of a matrix, $\alpha \in (0, 1)$ is a tradeoff parameter to balance the source and target domain data, and λ_u, λ_u, and λ_u are the regularization parameters to control the influence of the constraints on the user similarities, item similarities, and algorithm complexity. Details on how these parameters affect the proposed method are presented in Section V. Using gradient descent, the objective function is minimized with the following update rules:

$$
\begin{aligned}
\boldsymbol{U}_s &\leftarrow \boldsymbol{U}_s - \eta_{U_s}\left[\alpha\left(\boldsymbol{U}_s \boldsymbol{V}_s^T - \boldsymbol{X}_s\right)\boldsymbol{V}_s\right. \\
&\qquad \left. + \lambda_u \boldsymbol{L}_u^{(s,s)} \boldsymbol{U}_s + \frac{\lambda_u}{2} \boldsymbol{L}_u^{(s,t)} \boldsymbol{U}_t + \lambda \boldsymbol{U}_s\right] \qquad (22) \\
\boldsymbol{V}_s &\leftarrow \boldsymbol{V}_s - \eta_{V_s}\left[\alpha\left(\boldsymbol{V}_s \boldsymbol{U}_s^T - \boldsymbol{X}_s^T\right)\boldsymbol{U}_s + \lambda_v \boldsymbol{L}_v^{(s,s)} \boldsymbol{V}_s + \lambda \boldsymbol{U}_s\right] \qquad (23) \\
\boldsymbol{U}_t &\leftarrow \boldsymbol{U}_t - \eta_{U_t}\left[\left(\boldsymbol{U}_t \boldsymbol{V}_t^T - \boldsymbol{X}_t\right)\boldsymbol{V}_t + \lambda_u \boldsymbol{L}_u^{(t,t)} \boldsymbol{U}_t\right. \\
&\qquad \left. + \frac{\lambda_u}{2}\left(\boldsymbol{L}_u^{(s,t)}\right)^T \boldsymbol{U}_s + \lambda \boldsymbol{U}_t\right] \qquad (24) \\
\boldsymbol{V}_t &\leftarrow \boldsymbol{V}_t - \eta_{V_t}\left[\left(\boldsymbol{V}_t \boldsymbol{U}_t^T - \boldsymbol{X}_t^T\right)\boldsymbol{U}_t + \lambda_v \boldsymbol{L}_v^{(t,t)} \boldsymbol{V}_t + \lambda \boldsymbol{U}_t\right]. \qquad (25)
\end{aligned}
$$

By updating $\boldsymbol{U}_s$, $\boldsymbol{V}_s$, $\boldsymbol{U}_t$, and $\boldsymbol{V}_t$ iteratively, we arrive at a final optimized approximation of $\hat{\boldsymbol{X}}_t = \boldsymbol{U}_t \boldsymbol{V}_t^T$. Recommendations are given according to the rating prediction in the target domain.

C. Small-Scale Example

To better illustrate our method, this section outlines a small-scale example. Suppose the source domain of a recommender system contains five users, denoted as $\mathcal{U}_s = \{u_1, u_2, u_3, u_4, u_5\}$, and nine items, denoted as $\mathcal{I}_s = \{i_1, i_2, i_3, i_4, i_5, i_6, i_7, i_8, i_9\}$. The target domain contains six users, denoted as $\mathcal{U}_t = \{u_6, u_7, u_8, u_9, u_{10}, u_{11}\}$, and nine items, denoted as $\mathcal{I}_t = \{j_1, j_2, j_3, j_4, j_5, j_6, j_7, j_8, j_9\}$. Four of the nine items in the source domain i_6, i_7, i_8, i_9 correspond to four of the nine items in the target domain - j_1, j_2, j_3, j_4, respectively, as shown in Fig. 4.

Step 1 (Extracting and Aligning Features): With the lower-dimension K set to 4, the source rating matrix $\boldsymbol{X}_s$ and the target rating matrix $\boldsymbol{X}_t$ are factorized into user feature matrix and item feature matrix according to (5), respectively. Then, the features of the overlapping items are projected onto the same feature space using GFK. $\boldsymbol{V}_{s,o}$ and $\boldsymbol{V}_{t,o}$ are used to find the proper projection matrix, and the item feature matrices $\boldsymbol{V}_s^{(0)}$ and $\boldsymbol{V}_t^{(0)}$ are transformed into $\boldsymbol{V}_s^{(1)}$ and $\boldsymbol{V}_t^{(1)}$. The example in Fig. 5 shows the domain adaptation process using GFK. In this small-scale example, there is not sufficient data to generate meaningful distribution. Therefore, we have used the result and item feature matrices from our Task 2 experiment in Section V. Fig. 5(a) shows the distribution of one item feature for the items that overlap in the source and target domains. It is apparent that a divergence exists between these two distributions. Fig. 5(b) shows the adjustment to the feature after domain adaptation. Here, the feature distributions for the source domain and the target domain are far better aligned.

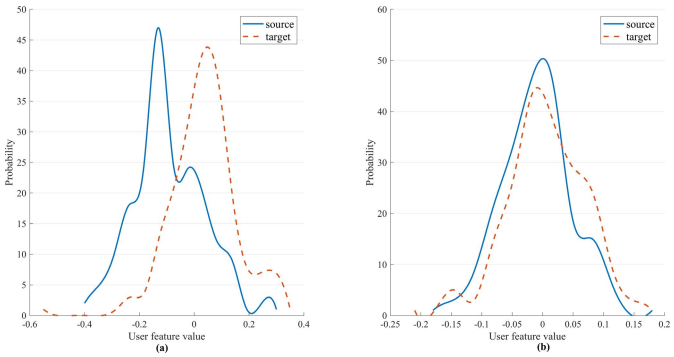


Fig. 5. Example of domain adaptation on user feature matrix with Netflix data set. (a) Marginal probability distribution of one column in $\boldsymbol{V}_s^{(0)}$ and $\boldsymbol{V}_t^{(0)}$ before domain adaptation with Netflix data set experiment. (b) Marginal probability distribution of one column in $\boldsymbol{V}_s^{(1)}$ and $\boldsymbol{V}_t^{(1)}$ after domain adaptation with Netflix data set experiment.

Step 2 (Item Feature Regulation): $\boldsymbol{U}_s^{(1)}$ and $\boldsymbol{U}_t^{(1)}$ are updated according to (13).

Step 3 (Similarity Measures): This step shows how the item similarity matrix $\boldsymbol{W}_v$ is calculated, given that item overlaps exist in our example. With item feature matrices from the source domain and the target domain, the item similarity in the same domain can be calculated directly through RBF measurement. Together with the overlap information, the item similarity matrix $\boldsymbol{W}_v$ is

$$
\boldsymbol{W}_v = \begin{bmatrix} \boldsymbol{W}_v^{(s,s)}, & \boldsymbol{W}_v^{(s,t)} \\ \boldsymbol{W}_v^{(t,s)}, & \boldsymbol{W}_v^{(t,t)} \end{bmatrix}
$$

where $\boldsymbol{W}_v^{(s,s)}$, $\boldsymbol{W}_v^{(t,t)}$, and $\boldsymbol{W}_v^{(s,t)}$ are shown at the top of the next page.

Step 4 (Kernel Induced Completion): The similarity matrix $\boldsymbol{W}_v^{(s,t)}$ is completed according to (19). In more detail, the low-rank eigendecomposition for both $\boldsymbol{W}_v^{(s,s)}$ and $\boldsymbol{W}_v^{(t,t)}$ is

$$
\begin{aligned}
e^{\left(\beta_s \boldsymbol{W}_v^{(s,s)}\right)} &\approx \boldsymbol{Q}_s e^{(\beta_s \boldsymbol{D}_s)} \boldsymbol{Q}_s^T \\
e^{\left(\beta_t \boldsymbol{W}_v^{(t,t)}\right)} &\approx \boldsymbol{Q}_t e^{(\beta_t \boldsymbol{D}_t)} \boldsymbol{Q}_t^T
\end{aligned}
$$

where $\boldsymbol{D}_s$ and $\boldsymbol{D}_t$ are the k_s, k_t leading eigenvalues for $\boldsymbol{W}_v^{(s,s)}$ and $\boldsymbol{W}_v^{(t,t)}$ and $\boldsymbol{Q}_s$ and $\boldsymbol{Q}_t$ are the corresponding eigenvalues.

$$\boldsymbol{W}_v^{(s,s)} = \begin{bmatrix} 1 & 0.859 & 0.930 & 0.842 & 0.301 & 0.731 & 0.870 & 0.446 & 0.630 \\ 0.859 & 1 & 0.808 & 0.934 & 0.345 & 0.831 & 0.924 & 0.510 & 0.542 \\ 0.930 & 0.808 & 1 & 0.787 & 0.281 & 0.684 & 0.810 & 0.416 & 0.668 \\ 0.842 & 0.934 & 0.787 & 1 & 0.356 & 0.868 & 0.949 & 0.529 & 0.535 \\ 0.301 & 0.345 & 0.281 & 0.356 & 1 & 0.410 & 0.345 & 0.673 & 0.193 \\ 0.731 & 0.831 & 0.684 & 0.868 & 0.410 & 1 & 0.833 & 0.608 & 0.466 \\ 0.870 & 0.924 & 0.810 & 0.949 & 0.345 & 0.833 & 1 & 0.512 & 0.555 \\ 0.446 & 0.510 & 0.416 & 0.529 & 0.673 & 0.608 & 0.512 & 1 & 0.286 \\ 0.630 & 0.542 & 0.668 & 0.535 & 0.193 & 0.466 & 0.555 & 0.286 & 1 \end{bmatrix}$$

$$\boldsymbol{W}_v^{(t,t)} = \begin{bmatrix} 1 & 0.997 & 0.991 & 0.997 & 0.995 & 0.996 & 0.986 & 0.988 & 0.987 \\ 0.997 & 1 & 0.991 & 0.994 & 0.995 & 0.998 & 0.989 & 0.991 & 0.989 \\ 0.991 & 0.991 & 1 & 0.994 & 0.996 & 0.992 & 0.982 & 0.982 & 0.994 \\ 0.997 & 0.994 & 0.994 & 1 & 0.997 & 0.995 & 0.984 & 0.985 & 0.989 \\ 0.995 & 0.995 & 0.996 & 0.997 & 1 & 0.996 & 0.986 & 0.986 & 0.992 \\ 0.996 & 0.998 & 0.992 & 0.995 & 0.996 & 1 & 0.989 & 0.990 & 0.990 \\ 0.986 & 0.989 & 0.982 & 0.984 & 0.986 & 0.989 & 1 & 0.995 & 0.984 \\ 0.988 & 0.991 & 0.982 & 0.985 & 0.986 & 0.990 & 0.995 & 1 & 0.982 \\ 0.987 & 0.989 & 0.994 & 0.989 & 0.992 & 0.990 & 0.984 & 0.982 & 1 \end{bmatrix}$$

$$\boldsymbol{W}_v^{(s,t)} = \left(\boldsymbol{W}_v^{(t,s)}\right)^T = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\hat{\boldsymbol{W}}_v^{(s,t)} = \begin{bmatrix} 0 & 0 & 0 & 0.003 & 0 & 0 & 0 & 0 & 0 \\ 0.004 & 0.004 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.003 & 0 & 0 & 0 & 0 & 0 \\ 0.004 & 0.005 & 0.002 & 0 & 6.7 \times 10^{-5} & 6.7 \times 10^{5} & 6.7 \times 10^{5} & 6.7 \times 10^{5} & 6.7 \times 10^{5} \\ 0 & 0 & 0.003 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0.009 & 0.008 & 0.007 & 0.005 & 0.005 & 0.005 & 0.005 & 0.005 \\ 0.009 & 1 & 0.007 & 0.007 & 0.005 & 0.005 & 0.005 & 0.005 & 0.005 \\ 0.008 & 0.007 & 1 & 0.006 & 0.005 & 0.005 & 0.005 & 0.005 & 0.005 \\ 0.007 & 0.007 & 0.006 & 1 & 0.005 & 0.005 & 0.005 & 0.005 & 0.005 \end{bmatrix}$$

Diffusion completion is then conducted as

$$\hat{\boldsymbol{W}}_v^{(s,t)} = \left(\boldsymbol{Q}_s e^{(\beta_s \boldsymbol{D}_s)} \boldsymbol{Q}_s^T\right) \boldsymbol{W}_v^{(s,t)} \left(\boldsymbol{Q}_t e^{(\beta_t \boldsymbol{D}_t)} \boldsymbol{Q}_t^T\right).$$

The final result of $\hat{\boldsymbol{W}}_v^{(s,t)}$ in this example is shown at the top of the next page.

Step 5 (Collective Matrix Factorization): With the user and item similarity matrix available, the target rating matrix is approximated as indicated in (21), and recommendations can be given accordingly.

V. Experiments and Analysis

This section presents the experimental results and the related analysis. The data sets and evaluation metrics are introduced first, followed by the experimental settings and baseline methods. Then, we present the results of the empirical experiments, with a parameter analysis to conclude this section.

A. Data Sets and Evaluation Metrics

Our method was tested under the conditions that the source and target domains share some overlapping users and/or items. For a fair comparison, we chose movies and books as the recommendation subject—two commonly used categories in previous research on cross-domain recommender systems. Four real-world data sets were used in our experiments: Movielens,[1] Netflix,[2] AmazonBook[3] [39], and Douban[4] [40]. Each of these data sets is publicly available and has been tested on single domain recommendation in a variety of situations, but rarely in cross-domain recommendation. Our experiments are a supplement to the lack of tests in this

[1] https://grouplens.org/datasets/movielens/20m/
[2] https://netflixprize.com/index.html
[3] http://jmcauley.ucsd.edu/data/amazon
[4] https://sites.google.com/site/erhengzhong/datasets

TABLE I
STATISTICS OF ORIGINAL DATA SETS

	Movielens20M	Netflix	Amazon_book	Douban_movie	Douban_book
#user	138493	480189	8026324	28718	26877
#item	26744	17770	2330066	57424	187520
#rating	20000263	100480507	22507155	2828585	1097148
sparsity	0.54%	1.18%	0.0001%	0.17%	0.02%
rating_range	0.5-5	1-5	1-5	1-5	1-5

specific problem setting. The statistical information for these data sets is presented in Table I.

From AmazonBooks, we removed all users who had given exactly the same rating for every book, as these data are not effective for constructing a recommender system [28]. Movielens20M was normalized to the range of $\{1, 2, 3, 4, 5\}$. The following four cross-domain recommendation tasks were designed for experiments as shown in Table II.

Task 1: Movie $\rightarrow$ movie, user-overlap, Movielens20M.
Task 2: Movie $\rightarrow$ movie, item-overlap, Netflix.
Task 3: Book $\rightarrow$ book, item-overlap, AmazonBook.
Task 4: Movie $\rightarrow$ book, user-overlap, Douban.

In the first three tasks, we used the data from one data set and split the entities into the source domain and the target domain to simulate entity overlaps. The fourth task was designed for Douban, a real-world rating website where users can rate movies, books, and music. We now turn to Task 1 as an example to describe how the data were selected. The process for Task 2 and Task 3 was similar but with overlapping items. With the source domain data, we filtered out the users who had given less than a total of 20 ratings and items who had received less than 10 ratings. We randomly selected 2000 items and 2000 users, constraining the sparsity to 2% to ensure that the source domain data were relatively dense. In the 2000 users, we randomly chose 200 users as overlapping users. We then randomly selected 1800 users with no correspondence to the 2000 users in the source domain. In total, these users composed the 2000 users in the target domain data. In addition, we also randomly chose 2000 items for the target domain data that had no intersections with items in the source domain. We tested the target domain data with three sparsity ratios to compare the different algorithms in different circumstances.

Mean absolute error (MAE) and root mean square error (RMSE) were used as evaluation metrics

$$\text{MAE} = \sum_{u,v,X_{uv}\in Y} \frac{|\hat{X}_{uv} - X_{uv}|}{|Y|}$$

$$\text{RMSE} = \sqrt{\sum_{u,v,X_{uv}\in Y} \frac{(\hat{X}_{uv} - X_{uv})^2}{|Y|}}$$

where X_{uv} and $\hat{X}_{uv}$ are the true and predicted ratings, Y is the test set, and $|Y|$ is the number of the test set. The smaller the errors, the better the performance.

B. Experimental Settings and Baselines

Three nontransfer learning methods were chosen for comparison: Pearson's correlation coefficient (PCC) [41], flexible mixture model (FMM) [42], and PMF [35], along with three cross-domain recommendation methods, CBT [26], RMGM [27], and PMF transfer learning (PMFTL) [33]. PCC is a classical memory-based collaborative filtering method. FMM is a graphical model designed to allow one user/item to be clustered into several groups simultaneously. Empirically, FMM has been proven to be more effective in providing recommendations to users with a few historical ratings. RMGM is a cross-domain recommendation method that evolved out of the single domain FMM. CBT is also a cross-domain recommendation method. Both of these methods were designed for scenarios with no overlapping users or items. PMFTL is a transfer learning method for cross-domain scenarios with entity overlap as proposed in [33]. However, for a fair comparison, we removed the active-learning module in the originally proposed method. PMFTL was developed on the basis of PMF with partially overlapping entities. PMFTL has more relaxed constraints than TCF [29]. TCF was designed for problems where users and items have a one-to-one mapping. The constraints in TCF are strict. One constraint requires that the user and item feature matrices in the source and target domains are exactly the same, whereas PMFTL uses similarities estimated in the source domain directly as constraints in the target domain. Since TCF cannot be used to solve the problem presented in this paper, we did not select it for comparison.

TABLE II
DESCRIPTION OF DATA SUBSETS FOR FOUR TASKS

Task	Data_name	Data_source	Domain	Sparsity	No. of overlapping
Task 1	task1_s1	Movielens20M	source	2.00%	200, 100, 50
	task1_t1	Movielens20M	target	0.50%	200, 100, 50
	task1_t2	Movielens20M	target	1.00%	200, 100, 50
	task1_t3	Movielens20M	target	1.50%	200, 100, 50
Task 2	task2_s1	Netflix	source	2.00%	200, 100, 50
	task2_t1	Netflix	target	0.50%	200, 100, 50
	task2_t2	Netflix	target	1.00%	200, 100, 50
	task2_t3	Netflix	target	1.50%	200, 100, 50
Task 3	task3_s1	AmazonBook	source	2.00%	200, 100, 50
	task3_t1	AmazonBook	target	0.50%	200, 100, 50
	task3_t2	AmazonBook	target	0.63%	200, 100, 50
	task3_t3	AmazonBook	target	0.75%	200, 100, 50
Task 4	task4_s1	DoubanMovie	source	2.00%	200, 100, 50
	task4_t1	DoubanBook	target	0.50%	200, 100, 50
	task4_t2	DoubanBook	target	1.00%	200, 100, 50
	task4_t3	DoubanBook	target	1.50%	200, 100, 50

User-based collaborative filtering was used for PCC, with the number of neighboring users set to 50. For FMM, CBT, and RMGM, the number of user and item groups was both set to 50. For PMF and PMFTL, we set $\lambda = \{0.01, 0.05, 0.1, 0.2, 0.3, 0.5, 1\}$. The parameter settings for KerKT were $\alpha = 0.5$, $\lambda_u = \{0.001, 0.01\}$, $\lambda_v = \{0.001, 0.01\}$, and $\lambda = \{0.0001, 0.001, 0.01\}$. KerKT and all the baselines, except for PCC, were randomly initialized. The results of 20 random initializations were averaged; standard deviations are reported.

C. Results

The results with these four data sets are shown in Tables III–VI, and a visual comparison is shown in Fig. 6. KerKT delivered the best performance of all the comparison methods on all four cross-domain recommendation tasks. This verifies the conclusion that using overlapping entities as a bridge for transferring knowledge is useful in cross-domain

TABLE III
OVERALL COMPARISON RESULT ON THE MOVIELENS DATA

	Method	1.00%			1.50%			2.00%		
		200	100	50	200	100	50	200	100	50
MAE	PCC	0.7877	0.7822	0.7825	0.7227	0.7201	0.7176	0.6923	0.6929	0.6928
	FMM	0.6708 (±0.0013)	0.6727 (±0.0015)	0.6704 (±0.0019)	0.6527 (±0.0013)	0.6532 (±0.0009)	0.6496 (±0.0009)	0.6458 (±0.0009)	0.6471 (±0.0007)	0.6498 (±0.0010)
	PMF	0.7007 (±0.0006)	0.7003 (±0.0008)	0.7026 (±0.0012)	0.6797 (±0.0013)	0.6853 (±0.0026)	0.6777 (±0.0010)	0.6560 (±0.0007)	0.6567 (±0.0005)	0.6601 (±0.0009)
	RMGM	0.6659 (±0.0012)	0.6698 (±0.0010)	0.6668 (±0.0024)	0.6524 (±0.0010)	0.6531 (±0.0009)	0.6500 (±0.0012)	0.6466 (±0.0008)	0.6473 (±0.0006)	0.6506 (±0.0006)
	CBT	0.7489 (±0.0030)	0.7527 (±0.0017)	0.7513 (±0.0020)	0.7499 (±0.0041)	0.7509 (±0.0037)	0.7456 (±0.0030)	0.7469 (±0.0029)	0.7478 (±0.0033)	0.7491 (±0.0029)
	PMFTL	0.7222 (±0.0023)	0.7182 (±0.0017)	0.7227 (±0.0008)	0.6931 (±0.0042)	0.6951 (±0.0050)	0.6819 (±0.0047)	0.6822 (±0.0036)	0.6764 (±0.0015)	0.6719 (±0.0037)
	KerKT	**0.6566** (±0.0029)	**0.6553** (±0.0009)	**0.6563** (±0.0017)	**0.6411** (±0.0009)	**0.6423** (±0.0005)	**0.6405** (±0.0037)	**0.6411** (±0.0007)	**0.6371** (±0.0018)	**0.6403** (±0.0005)
RMSE	PCC	1.0087	1.0028	1.0009	0.9259	0.9203	0.9172	0.8826	0.8839	0.8830
	FMM	0.8575 (±0.0014)	0.8604 (±0.0017)	0.8566 (±0.0019)	0.8339 (±0.0016)	0.8339 (±0.0010)	0.8314 (±0.0012)	0.8254 (±0.0010)	0.8294 (±0.0010)	0.8280 (±0.0011)
	PMF	0.8808 (±0.0007)	0.8812 (±0.0012)	0.8821 (±0.0007)	0.8613 (±0.0014)	0.8588 (±0.0029)	0.8574 (±0.0013)	0.8312 (±0.0008)	0.8341 (±0.0007)	0.8338 (±0.0010)
	RMGM	0.8509 (±0.0014)	0.8563 (±0.0010)	0.8511 (±0.0013)	0.8327 (±0.0014)	0.8330 (±0.0012)	0.831 (±0.0011)	0.8259 (±0.0009)	0.8291 (±0.0007)	0.8284 (±0.0010)
	CBT	0.9663 (±0.0051)	0.9703 (±0.0036)	0.9683 (±0.0047)	0.9668 (±0.0065)	0.9650 (±0.0054)	0.9611 (±0.0052)	0.9613 (±0.0050)	0.9624 (±0.0050)	0.9641 (±0.0050)
	PMFTL	0.9083 (±0.0023)	0.9050 (±0.0025)	0.9041 (±0.0031)	0.8715 (±0.0045)	0.8712 (±0.0053)	0.8574 (±0.0040)	0.8592 (±0.0041)	0.8566 (±0.0020)	0.8482 (±0.0037)
	KerKT	**0.8355** (±0.0025)	**0.8352** (±0.0007)	**0.8346** (±0.0013)	**0.8180** (±0.0016)	**0.8169** (±0.0008)	**0.8181** (±0.0064)	**0.8158** (±0.0008)	**0.8156** (±0.0038)	**0.8143** (±0.0018)

TABLE IV
OVERALL COMPARISON RESULT ON THE NETFLIX DATA

	Method	1.00%			1.50%			2.00%		
		200	100	50	200	100	50	200	100	50
MAE	PCC	1.0191	1.0108	1.0026	0.9099	0.9079	0.9154	0.8139	0.8188	0.8224
	FMM	0.7460 (±0.0015)	0.7452 (±0.0015)	0.7431 (±0.0016)	0.7303 (±0.0012)	0.7337 (±0.0016)	0.7349 (±0.0012)	0.7225 (±0.0010)	0.7251 (±0.0010)	0.7297 (±0.0011)
	PMF	0.8121 (±0.0017)	0.8119 (±0.0015)	0.8111 (±0.0012)	0.7681 (±0.0006)	0.7729 (±0.0005)	0.7750 (±0.0004)	0.7474 (±0.0016)	0.7514 (±0.0013)	0.7567 (±0.0011)
	RMGM	0.7432 (±0.0012)	0.7427 (±0.0010)	0.7396 (±0.0014)	0.7300 (±0.0010)	0.7338 (±0.0008)	0.7360 (±0.0015)	0.7237 (±0.0009)	0.7265 (±0.0010)	0.7319 (±0.0010)
	CBT	0.8596 (±0.0073)	0.8558 (±0.0079)	0.8600 (±0.0050)	0.8546 (±0.0051)	0.8505 (±0.0063)	0.8582 (±0.0080)	0.8497 (±0.0068)	0.8541 (±0.0077)	0.8579 (±0.0061)
	PMFTL	0.8285 (±0.0024)	0.8231 (±0.0054)	0.8243 (±0.0055)	0.7827 (±0.0037)	0.7814 (±0.0019)	0.7843 (±0.0045)	0.7649 (±0.0035)	0.7669 (±0.0042)	0.7655 (±0.0056)
	KerKT	**0.7364** (±0.0004)	**0.7375** (±0.0037)	**0.7293** (±0.0014)	**0.7183** (±0.0012)	**0.7253** (±0.0013)	**0.7250** (±0.0006)	**0.7137** (±0.0016)	**0.7172** (±0.0027)	**0.7213** (±0.0032)
RMSE	PCC	1.2968	1.2835	1.2710	1.1571	1.1567	1.1636	1.0325	1.0384	1.0447
	FMM	0.9490 (±0.0019)	0.9473 (±0.0015)	0.9417 (±0.0023)	0.9278 (±0.0010)	0.9307 (±0.0019)	0.9319 (±0.0012)	0.9172 (±0.0009)	0.9217 (±0.0011)	0.9253 (±0.0009)
	PMF	0.9978 (±0.0016)	0.9973 (±0.0013)	0.9949 (±0.0013)	0.9499 (±0.0007)	0.9541 (±0.0005)	0.9567 (±0.0005)	0.9360 (±0.0018)	0.9431 (±0.0018)	0.9461 (±0.0014)
	RMGM	0.9437 (±0.0013)	0.9432 (±0.0012)	0.9365 (±0.0014)	0.9267 (±0.0010)	0.9301 (±0.0009)	0.9322 (±0.0015)	0.9175 (±0.0008)	0.9228 (±0.0010)	0.9268 (±0.0008)
	CBT	1.0448 (±0.0027)	1.0392 (±0.0030)	1.0417 (±0.0021)	1.0363 (±0.0026)	1.0318 (±0.0030)	1.0399 (±0.0037)	1.0330 (±0.0029)	1.0367 (±0.0036)	1.0403 (±0.0030)
	PMFTL	1.0078 (±0.0021)	1.0031 (±0.0054)	1.0061 (±0.0052)	0.9675 (±0.0038)	0.9665 (±0.0016)	0.9684 (±0.0030)	0.9467 (±0.0027)	0.9509 (±0.0033)	0.9491 (±0.0042)
	KerKT	**0.9349** (±0.0004)	**0.9351** (±0.0032)	**0.9236** (±0.0012)	**0.9125** (±0.0007)	**0.9199** (±0.0016)	**0.9219** (±0.0008)	**0.9057** (±0.0008)	**0.9116** (±0.0024)	**0.9132** (±0.0028)

recommender systems. Our analysis of the results revealed the following observations.

1) *Comparison With Nontransfer Learning Methods:* The performance of nontransfer learning methods was relatively poor on sparse data. As the basis of CBT and RMGM, FMM was designed to predict ratings for users with little available data. Generally, FMM performed better than PMF and the memory-based method PCC without transfer learning techniques. PMF is the basis of our proposed method KerKT. In all the experiments,

TABLE V
OVERALL COMPARISON RESULT ON THE AMAZONBOOK DATA

	Method	0.75%			0.63%			0.50%		
		200	100	50	200	100	50	200	100	50
MAE	PCC	0.7705	0.7551	1.0026	0.7704	0.7618	0.7615	0.7882	0.7718	0.9154
	FMM	0.6648 (±0.0029)	0.6630 (±0.0026)	0.7431 (±0.0019)	0.6846 (±0.0028)	0.6806 (±0.0031)	0.6822 (±0.0032)	0.7171 (±0.0034)	0.7132 (±0.0041)	0.7349 (±0.0012)
	PMF	0.6740 (±0.0011)	0.6751 (±0.0004)	0.8111 (±0.0012)	0.6934 (±0.0003)	0.6846 (±0.0010)	0.6843 (±0.0006)	0.6855 (±0.0001)	0.6962 (±0.0001)	0.7750 (±0.0004)
	RMGM	0.6581 (±0.0016)	0.6578 (±0.0021)	0.6521 (±0.0024)	0.6706 (±0.0024)	0.6671 (±0.0024)	0.6657 (±0.0026)	0.6855 (±0.0021)	0.6831 (±0.0025)	0.7316 (±0.0015)
	CBT	0.6677 (±0.0019)	0.6678 (±0.0026)	0.6717 (±0.0020)	0.6734 (±0.0022)	0.6712 (±0.0014)	0.6714 (±0.0021)	0.6753 (±0.0015)	0.6719 (±0.0030)	0.6731 (±0.0013)
	PMFTL	0.6759 (±0.0006)	0.6796 (±0.0016)	0.6815 (±0.0008)	0.6855 (±0.0003)	0.6863 (±0.0006)	0.6844 (±0.0011)	0.6886 (±0.0006)	0.6882 (±0.0004)	0.6902 (±0.0004)
	KerKT	**0.6562** (±0.0034)	**0.6511** (±0.0038)	**0.6420** (±0.0039)	**0.6529** (±0.0030)	**0.6580** (±0.0070)	**0.6502** (±0.0018)	**0.6601** (±0.0039)	**0.6611** (±0.0061)	**0.6611** (±0.0066)
RMSE	PCC	0.9203	0.9129	0.9949	0.9277	0.9337	0.9295	0.9462	0.9476	0.9567
	FMM	0.8732 (±0.0049)	0.8668 (±0.0015)	0.9417 (±0.0023)	0.8951 (±0.0034)	0.8916 (±0.0040)	0.8956 (±0.0036)	0.9402 (±0.0049)	0.9361 (±0.0051)	0.9319 (±0.0012)
	PMF	0.9203 (±0.0021)	0.9129 (±0.0012)	0.9949 (±0.0013)	0.9277 (±0.0006)	0.9337 (±0.0017)	0.9295 (±0.0009)	0.9462 (±0.0001)	0.9476 (±0.0003)	0.9567 (±0.0005)
	RMGM	0.8636 (±0.0023)	0.8576 (±0.0031)	0.8532 (±0.0032)	0.8748 (±0.0028)	0.8717 (±0.0027)	0.8734 (±0.0031)	0.8976 (±0.0028)	0.8971 (±0.0034)	0.9322 (±0.0015)
	CBT	0.9159 (±0.0031)	0.9084 (±0.0041)	0.9110 (±0.0034)	0.9143 (±0.0031)	0.9156 (±0.0024)	0.9124 (±0.0030)	0.9194 (±0.0032)	0.9716 (±0.0042)	0.9165 (±0.0020)
	PMFTL	0.9057 (±0.0016)	0.9029 (±0.0016)	0.9163 (±0.0012)	0.9292 (±0.0005)	0.9399 (±0.0012)	0.9130 (±0.0019)	0.9382 (±0.0013)	0.9442 (±0.0006)	0.9449 (±0.0003)
	KerKT	**0.8597** (±0.0049)	**0.8430** (±0.0091)	**0.8362** (±0.0054)	**0.8483** (±0.0054)	**0.8602** (±0.0137)	**0.8485** (±0.0039)	**0.8572** (±0.0067)	**0.8641** (±0.0101)	**0.8572** (±0.0104)

TABLE VI
OVERALL COMPARISON RESULT ON THE DOUBAN DATA

	Method	1.00%			1.50%			2.00%		
		200	100	50	200	100	50	200	100	50
MAE	PCC	0.6695	0.6700	0.6700	0.6317	0.6306	0.6298	0.6076	0.6079	0.6154
	FMM	0.5950 (±0.0011)	0.6013 (±0.0015)	0.5936 (±0.0009)	0.5834 (±0.0008)	0.5857 (±0.0011)	0.5834 (±0.0006)	0.5775 (±0.0008)	0.5828 (±0.0010)	0.5787 (±0.0008)
	PMF	0.6256 (±0.0008)	0.6288 (±0.0008)	0.6248 (±0.0011)	0.5962 (±0.0008)	0.6033 (±0.0011)	0.5993 (±0.0011)	0.5848 (±0.0004)	0.5864 (±0.0005)	0.5867 (±0.0003)
	RMGM	0.5929 (±0.0008)	0.6004 (±0.0013)	0.5912 (±0.0011)	0.5835 (±0.0009)	0.5860 (±0.0009)	0.5832 (±0.0005)	0.5773 (±0.0010)	0.5835 (±0.0011)	0.5795 (±0.0006)
	CBT	0.6422 (±0.0138)	0.6627 (±0.0126)	0.6453 (±0.0145)	0.6410 (±0.0149)	0.6524 (±0.0114)	0.6324 (±0.0135)	0.6346 (±0.0096)	0.6418 (±0.0109)	0.6373 (±0.0115)
	PMFTL	0.6308 (±0.0034)	0.6353 (±0.0009)	0.6256 (±0.0011)	0.6101 (±0.0009)	0.6139 (±0.0033)	0.6077 (±0.0032)	0.6009 (±0.0019)	0.5984 (±0.0058)	0.5949 (±0.0015)
	KerKT	**0.5863** (±0.0013)	**0.5922** (±0.0020)	**0.5835** (±0.0012)	**0.5778** (±0.0013)	**0.5812** (±0.0020)	**0.5788** (±0.0019)	**0.5721** (±0.0005)	**0.5752** (±0.0016)	**0.5757** (±0.0011)
RMSE	PCC	0.8618	0.8686	0.8655	0.8052	0.8082	0.8021	0.7672	0.7724	0.7744
	FMM	0.7508 (±0.0014)	0.7585 (±0.0020)	0.7504 (±0.0011)	0.7358 (±0.0007)	0.7373 (±0.0012)	0.7337 (±0.0006)	0.7267 (±0.0007)	0.7332 (±0.0012)	0.7278 (±0.0008)
	PMF	0.7876 (±0.0011)	0.7980 (±0.0015)	0.7857 (±0.0015)	0.7500 (±0.0009)	0.7609 (±0.0014)	0.7498 (±0.0013)	0.7327 (±0.0005)	0.7394 (±0.0009)	0.7332 (±0.0003)
	RMGM	0.7461 (±0.0009)	0.7557 (±0.0016)	0.7452 (±0.0010)	0.7344 (±0.0006)	0.7371 (±0.0012)	0.7316 (±0.0007)	0.7261 (±0.0008)	0.7341 (±0.0015)	0.7273 (±0.0007)
	CBT	0.8292 (±0.0054)	0.8484 (±0.0050)	0.8312 (±0.0074)	0.8239 (±0.0071)	0.8378 (±0.0057)	0.8189 (±0.0083)	0.8159 (±0.0041)	0.8290 (±0.0041)	0.8158 (±0.0054)
	PMFTL	0.8059 (±0.0044)	0.8095 (±0.0017)	0.7896 (±0.0013)	0.7761 (±0.0017)	0.7761 (±0.0046)	0.7635 (±0.0046)	0.7605 (±0.0033)	0.7548 (±0.0085)	0.7454 (±0.0018)
	KerKT	**0.7369** (±0.0013)	**0.7429** (±0.0027)	**0.7352** (±0.0013)	**0.7268** (±0.0008)	**0.7272** (±0.0024)	**0.7246** (±0.0022)	**0.7185** (±0.0007)	**0.7258** (±0.0036)	**0.7248** (±0.0023)

KerKT significantly outperformed all the nontransfer learning recommendation techniques.

2) *Comparison With Cross-Domain Recommendation Methods for nonoverlapping Entities:* RMGM showed improved precision in recommendations over its basis, FMM, but sometimes the improvement was not significant (see Table VI). CBT did not always improve the performance of the recommender system and

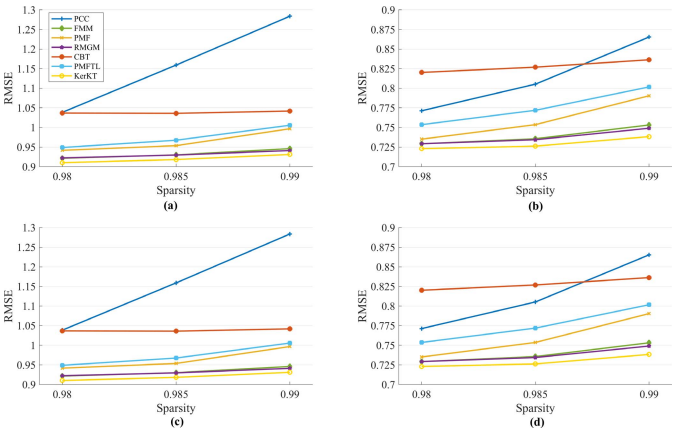


Fig. 6. Comparison results of four Data Sets.

sometimes suffered from negative transfer, indicating that CBT is not stable when transferring knowledge (see Table IV). Neither of these methods uses nonoverlapping entity information explicitly, but rather extracts cluster-based knowledge to share between the source and target domains. KerKT outperformed both these methods in all recommendation tasks. Methods designed for scenarios with nonoverlapping entities can be applied to solve the problem proposed in this paper as a substitution. But, we can see from the results that they did not show advantages since they did not use the overlapping information.

3) *Comparison With Cross-Domain Recommendation Methods for Partially Overlapping Entities:* As methods designed for partially overlapping entities are rare and methods developed for fully overlapping entities cannot be used to solve the problem proposed in this paper, only PMFTL serves as a representative of the cross-domain recommendation method for partially overlapping entities. PMFTL was developed on the basis of PMF with partially overlapping entities. The results of the experiments show that PMFTL was not effective in every situation since it ignores the divergence between the source and target domains. As a result, KerKT outperformed PMFTL in each of the four tasks with all three data sparsities.
4) *The Number of Overlapping Entities:* We tested three different levels of overlapping entities in these experiments: 200, 100, and 50. There were 2000 users/items in the target domain, so these proportions of overlapping entities represent just a small number of the overall total. However, even these small proportions still allowed knowledge to be transferred from the source domain to the target domain. We did not observe a very obvious increase/decrease in the precision of KerKT as the number of overlapping entities increased.

To further study the overall effectiveness of the KerKT method on these four tasks, we calculated the average MAEs and RMSEs as shown in Table VII. The results show that in each task, KerKT outperformed all the other baselines. Table VIII contains the percentage of improvement, which shows that KerKT delivered the greatest improvement

TABLE VII
OVERALL AVERAGE RESULT ON FOUR TASKS

		Non-transfer			Cross-domain			
		PCC	PMF	FMM	CBT	PMFTL	RMGM	KerKT
MAE	Task1	0.9134	0.7773	0.7338	0.8541	0.7913	0.7333	**0.7247**
	Task2	0.7696	0.6848	0.6872	0.6712	0.6840	0.6704	**0.6566**
	Task3	0.7330	0.6798	0.6571	0.7495	0.6979	0.6559	**0.6456**
	Task4	0.6369	0.6040	0.5868	0.6433	0.6131	0.5864	**0.5803**
RMSE	Task1	1.1608	0.9630	0.9323	1.0370	0.9738	0.9307	**0.9200**
	Task2	0.9314	0.9314	0.9005	0.9242	0.9267	0.8771	**0.8554**
	Task3	0.9374	0.8579	0.8401	0.9654	0.8786	0.8380	**0.8228**
	Task4	0.8139	0.7597	0.7394	0.8278	0.7757	0.7375	**0.7292**

TABLE VIII
OVERALL IMPROVEMENT OF KERKT ON FOUR TASKS

		Non-transfer			Cross-domain		
		PCC	PMF	FMM	CBT	PMFTL	RMGM
MAE	Task1	20.66%	6.76%	1.24%	15.14%	8.41%	1.17%
	Task2	14.69%	4.12%	4.46%	2.18%	4.01%	2.06%
	Task3	11.92%	5.03%	1.75%	13.87%	7.49%	1.57%
	Task4	8.89%	3.92%	1.11%	9.79%	5.34%	1.04%
RMSE	Task1	20.75%	4.47%	1.32%	11.28%	5.53%	1.15%
	Task2	8.16%	8.16%	5.01%	7.44%	7.69%	2.47%
	Task3	12.22%	4.09%	2.05%	14.76%	6.35%	1.81%
	Task4	10.41%	4.02%	1.38%	11.91%	6.00%	1.13%

over the memory-based recommendation method PCC by around 20%. Compared with the cross-domain recommendation method RMGM, KerKT attained an improvement of 2%.

D. Complexity Analysis

This complexity analysis covers each step of the proposed method KerKT. For simplicity, the dimensions of the user and item features in both the domains have all been set to k. The time complexity of each step is listed as follows.

Step 1: $O(kn^2)$, for each iteration to update $\boldsymbol{U}_s^{(0)}$, $\boldsymbol{V}_s^{(0)}$, $\boldsymbol{U}_t^{(0)}$, and $\boldsymbol{V}_t^{(0)}$ and for each iteration to approximate $\boldsymbol{X}_s$ and $\boldsymbol{X}_t$ for the matrix factorization, and $O(k^2n)$, for the domain adaptation process to adjust the feature matrices of the overlapped entities.

Step 2: $O(kn^2)$ for each iteration to update the feature matrices without the overlapped entities.

Step 3: $O(kn^2)$ to calculate the similarity matrix of users and items.

Step 4: $O(n^3)$ for the diffusion kernel completion.

Step 5: $O(kn^2)$ for each iteration to update (22).

Since the number of iterations in Steps 3 and 5 are not infinite, KerKT's the total complexity is $O(n^3)$. We have also listed the time taken for each of the methods to complete Task 2 in Table IX. This experiment was conducted with 200 overlapping entities and 0.5% sparsity on a computer with 16-GB memory and 2.2-GHz Intel Core i7. We can see that the nontransfer methods were faster. This is due to their general simplicity, because they do not take cross-domain data into consideration.

Most of the time was spent on the user and item similarity matrix calculations. However, some parallel computing could be used to speed up these computations. Alternatively, these matrices could be precalculated and stored

TABLE IX
ELAPSED TIME COMPARISON ON NETFLIX DATA SET

Method	Non-transfer			Cross-domain			
	PCC	PMF	FMM	CBT	PMFTL	RMGM	KerKT
Time(s)	12.32	11.31	5.92	525.95	65.99	220.43	188.33

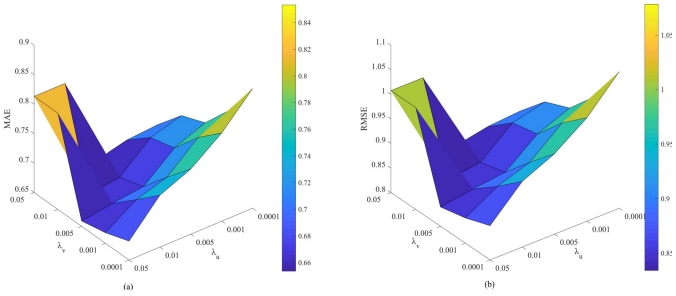


Fig. 7. Result of Movielens data set with different λ_u and λ_v parameter settings.

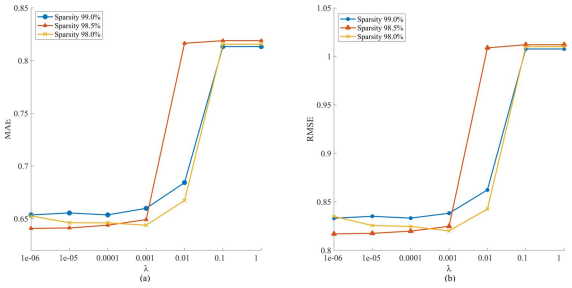


Fig. 8. Result of Movielens data set with different λ parameter settings.

so as not to affect the speed of online recommendation. Overall, the time complexity analysis shows that the KerKT method can be used with large-scale data sets and for online e-commerce or business-to-business systems.

E. Parameter Analysis

There are three important parameters in KerKT: λ_u, λ_v, and λ. Each is a tradeoff parameter in (21). For simplicity, we have only presented the results for the Movielens data set. This experiment was conducted with a sparsity ratio of 99.0% and 200 overlapping entities. MAE and RMSE were used as metrics. The results are presented in Figs. 7 and 8.

To analyze the parameters λ_u and λ_v, we set parameter λ to 0.0001. From Fig. 7, we can see that the MAE and RMSE change with different settings for λ_u and λ_v. These parameters reflect the influence of the user and item similarities on the matrix factorization, while the parameter λ restricts the complexity of the algorithm to avoid overfitting. We used a grid search to find the optimized settings for each of these parameters, λ, λ_u, and λ_v, which resulted in a setting of 0.01 for all.

VI. CONCLUSION AND FURTHER STUDY

This paper presents a novel cross-domain recommendation method for knowledge transfer, called KerKT. This method exploits overlapping entities as a bridge between the source and target domains and is applicable to e-commerce websites, such as Amazon, where book rating data are very dense but data in other categories are sparse. Unlike previous research, KerKT does not require that the entities be fully overlapped; it performs well in scenarios with partially overlapping entities. One advantage of this method is that it aligns the latent features of the entities extracted from the original ratings matrix. This fixes shifts in the entity feature space caused by user preference deviations between the domains. Furthermore, the entity similarity matrix is completed through diffusion kernel completion to tackle the inconsistencies caused by heterogeneous feature spaces between two domains. The similarity matrix is extended into matrix factorization with more flexible constraints to integrate the overlapping entity information. Experimental results from a comparison with six nontransfer learning and cross-domain recommendation methods show that KerKT achieved the best performance. Even with a small ratio of overlapping entities, it was still possible to transfer knowledge from the source domain to the target domain.

There is practical significance in studying and developing cross-domain recommender systems. Smart BizSeeker, a B2B recommender system, aims to recommend appropriate business partners to businesses in Australia [43]. In a future study, we plan to implement our proposed method into their system. Furthermore, there are still some interesting issues to be explored. For example, how to choose the best source domain if several domains are available? And which sparsity ratio in the target domain benefits transfer learning the most? In future work, we intend to apply our method to other kinds of data beyond ratings, such as Web browser records and social media records.

ACKNOWLEDGMENT

The authors would like to thank F. Liu for his valuable discussions on transfer learning and J. Moore for her proofreading of this paper.

REFERENCES

[1] J. Lu *et al.*, "Recommender system application developments: A survey," *Decision Support Syst.*, vol. 74, pp. 12–32, Jun. 2015.
[2] X. Luo *et al.*, "A nonnegative latent factor model for large-scale sparse matrices in recommender systems via alternating direction method," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 3, pp. 579–592, Mar. 2016.
[3] S. Boutemedjet and D. Ziou, "Predictive approach for user long-term needs in content-based image suggestion," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 23, no. 8, pp. 1242–1253, Aug. 2012.
[4] F. Zhang *et al.*, "Collaborative knowledge base embedding for recommender systems," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2016, pp. 353–362.
[5] Y. Koren, "Factorization meets the neighborhood: A multifaceted collaborative filtering model," in *Proc. 14th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2008, pp. 426–434.
[6] N. Guan *et al.*, "Online nonnegative matrix factorization with robust stochastic approximation," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 23, no. 7, pp. 1087–1099, Jul. 2012.
[7] B. Liu *et al.*, "A general geographical probabilistic factor model for point of interest recommendation," *IEEE Trans. Knowl. Data Eng.*, vol. 27, no. 5, pp. 1167–1179, May 2015.
[8] H. Wang, N. Wang, and D.-Y. Yeung, "Collaborative deep learning for recommender systems," in *Proc. 21st ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2015, pp. 1235–1244.
[9] A. Stuhlsatz, J. Lippel, and T. Zielke, "Feature extraction with deep neural networks by a generalized discriminant analysis," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 23, no. 4, pp. 596–608, Apr. 2012.
[10] J. Lu *et al.*, "Structural property-aware multilayer network embedding for latent factor analysis," *Pattern Recognit.*, vol. 76, pp. 228–241, Apr. 2018.
[11] J.-D. Zhang, C.-Y. Chow, and J. Xu, "Enabling kernel-based attribute-aware matrix factorization for rating prediction," *IEEE Trans. Knowl. Data Eng.*, vol. 29, no. 4, pp. 798–812, Apr. 2017.

[12] S. Deng *et al.*, "On deep learning for trust-aware recommendations in social networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 5, pp. 1164–1177, May 2017.

[13] B. Yang *et al.*, "Social collaborative filtering by trust," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 8, pp. 1633–1647, Aug. 2017.

[14] M. Long *et al.*, "Adaptation regularization: A general framework for transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 5, pp. 1076–1089, May 2014.

[15] J. Liu *et al.*, "Domain-sensitive recommendation with user-item subgroup analysis," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 4, pp. 939–950, Apr. 2016.

[16] S. Rendle and L. Schmidt-Thieme, "Online-updating regularized kernel matrix factorization models for large-scale recommender systems," in *Proc. ACM Conf. Rec. Syst.*, 2008, pp. 251–258.

[17] N. D. Lawrence and R. Urtasun, "Non-linear matrix factorization with Gaussian processes," in *Proc. 26th Annu. Int. Conf. Mach. Learn.*, 2009, pp. 601–608.

[18] M. A. Ghazanfar, A. Prügel-Bennett, and S. Szedmak, "Kernel-mapping recommender system algorithms," *Inf. Sci.*, vol. 208, pp. 81–104, Nov. 2012.

[19] R. R. Coifman and S. Lafon, "Diffusion maps," *Appl. Comput. Harmon. Anal.*, vol. 21, no. 1, pp. 5–30, 2006.

[20] R. I. Kondor and J. Lafferty, "Diffusion kernels on graphs and other discrete input spaces," in *Proc. ICML*, vol. 2, 2002, pp. 315–322.

[21] A. P. Singh and G. J. Gordon, "Relational learning via collective matrix factorization," in *Proc. 14th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2008, pp. 650–658.

[22] Y. Zhen, W.-J. Li, and D.-Y. Yeung, "TagiCoFi: Tag informed collaborative filtering," in *Proc. 3rd ACM Conf. Rec. Syst.*, 2009, pp. 69–76.

[23] P. Hao *et al.*, "Regularizing knowledge transfer in recommendation with tag-inferred correlation," *IEEE Trans. Cybern.*, to be published, doi: 10.1109/TCYB.2017.2764918.

[24] M. Jiang *et al.*, "Social recommendation with cross-domain transferable knowledge," *IEEE Trans. Knowl. Data Eng.*, vol. 27, no. 11, pp. 3084–3097, Nov. 2015.

[25] M. Pratama, J. Lu, and G. Zhang, "Evolving type-2 fuzzy classifier," *IEEE Trans. Fuzzy Syst.*, vol. 24, no. 3, pp. 574–589, Jun. 2016.

[26] B. Li, Q. Yang, and X. Xue, "Can movies and books collaborate?: cross-domain collaborative filtering for sparsity reduction," in *Proc. IJCAI*, vol. 9, 2009, pp. 2052–2057.

[27] B. Li, Q. Yang, and X. Xue, "Transfer learning for collaborative filtering via a rating-matrix generative model," in *Proc. 26th Annu. Int. Conf. Mach. Learn.*, 2009, pp. 617–624.

[28] Q. Zhang *et al.*, "A cross-domain recommender system with consistent information transfer," *Decision Support Syst.*, vol. 104, pp. 49–63, Dec. 2017.

[29] W. Pan and Q. Yang, "Transfer learning in heterogeneous collaborative filtering domains," *Artif. Intell.*, vol. 197, pp. 39–55, Apr. 2013.

[30] L. Hu *et al.*, "Personalized recommendation via cross-domain triadic factorization," in *Proc. 22nd Int. Conf. World Wide Web*, 2013, pp. 595–606.

[31] N. Mirbakhsh and C. X. Ling, "Improving top-N recommendation for cold-start users via cross-domain information," *ACM Trans. Knowl. Discovery Data*, vol. 9, no. 4, p. 33, 2015.

[32] C.-Y. Li and S.-D. Lin, "Matching users and items across domains to improve the recommendation quality," in *Proc. 20th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2014, pp. 801–810.

[33] L. Zhao, S. J. Pan, and Q. Yang, "A unified framework of active transfer learning for cross-system recommendation," *Artif. Intell.*, vol. 245, pp. 38–55, Apr. 2017.

[34] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, no. 8, pp. 30–37, Aug. 2009.

[35] A. Mnih and R. R. Salakhutdinov, "Probabilistic matrix factorization," in *Proc. Adv. Neural Inf. Process. Syst.*, 2008, pp. 1257–1264.

[36] B. Gong, K. Grauman, and F. Sha, "Learning kernels for unsupervised domain adaptation with applications to visual object recognition," *Int. J. Comput. Vis.*, vol. 109, nos. 1–2, pp. 3–27, 2014.

[37] C. H. Nguyen and H. Mamitsuka, "Latent feature kernels for link prediction on sparse graphs," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 23, no. 11, pp. 1793–1804, Nov. 2012.

[38] W.-C. Chang *et al.*, "Cross-domain kernel induction for transfer learning," in *Proc. AAAI*, 2017, pp. 1763–1769.

[39] R. He and J. McAuley, "Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering," in *Proc. 25th Int. Conf. World Wide Web*, 2016, pp. 507–517.

[40] E. Zhong, W. Fan, and Q. Yang, "User behavior learning and transfer in composite social networks," *ACM Trans. Knowl. Discovery Data*, vol. 8, no. 1, pp. 6:1–6:32, Feb. 2014.

[41] M. Deshpande and G. Karypis, "Item-based top-N recommendation algorithms," *ACM Trans. Inf. Syst.*, vol. 22, no. 1, pp. 143–177, 2004.

[42] L. Si and R. Jin, "Flexible mixture model for collaborative filtering," in *Proc. 20th Int. Conf. Mach. Learn.*, 2003, pp. 704–711.

[43] D. Wu, G. Zhang, and J. Lu, "A fuzzy preference tree-based recommender system for personalized business-to-business E-services," *IEEE Trans. Fuzzy Syst.*, vol. 23, no. 1, pp. 29–43, Feb. 2015.

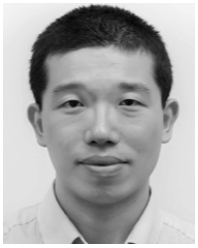
Qian Zhang (M'18) is currently pursuing the Ph.D. degree with the Decision Systems and e-Service Intelligent (DeSI) Research Laboratory, Center for Artificial Intelligence, Faculty of Engineering and Information Technology, University of Technology Sydney, Sydney, NSW, Australia.

She is also a member of the DeSI Research Laboratory, Center for Artificial Intelligence, Faculty of Engineering and Information Technology, University of Technology Sydney. Her current research interests include recommender systems and personalized techniques. She specializes in cross-domain recommender systems.

Jie Lu (F'18) received the Ph.D. degree in information systems from Curtin University, Perth, WA, Australia, in 2000.

She is currently a Distinguished Professor, the Director of the Center for Artificial Intelligence, and the Associate Dean (Research Excellence) of the Faculty of Engineering and Information Technology, University of Technology Sydney, Sydney, NSW, Australia. She has published 10 research books and over 400 papers in refereed journals and conference proceedings, with over 170 papers in IEEE TRANSACTIONS and other international journals. She has received over 20 Australian Research Council Discovery Project grants and many other research projects. Her current research interests include fuzzy transfer learning, decision support systems, recommender systems, and concept drift.

Dr. Lu is a fellow of the *International Journal of Fuzzy Systems*. She has served as a guest editor of 12 special issues and general/PC/organization chairs for 10 international conferences and delivered 20 keynote/plenary speeches at the IEEE and other international conferences. She serves as an Editor-in-Chief for *Knowledge-Based Systems* (Elsevier) and the *International Journal on Computational Intelligence Systems* (Atlantis) and an Associate Editor for the IEEE TRANSACTIONS ON FUZZY SYSTEMS.



Dianshuang Wu (M'18) is currently a Post-Doctoral Research Fellow and a member of the Decision Systems and e-Service Intelligent Research Laboratory, Center for Artificial Intelligence, Faculty of Engineering and Information Technology, University of Technology Sydney, Sydney, NSW, Australia. His current research interests include tree similarity measure, recommender systems, and business intelligence.

Guangquan Zhang received the Ph.D. degree in applied mathematics from Curtin University, Perth, WA, Australia, in 2001.

He is currently an Associate Professor and the Director of the Decision Systems and e-Service Intelligent Research Laboratory, Center for Artificial Intelligence, Faculty of Engineering and Information Technology, University of Technology Sydney, Sydney, NSW, Australia. He has authored five monographs, five textbooks, and 400 papers including over 200 refereed international journal papers. His current research interests include fuzzy sets and systems, fuzzy optimization, fuzzy transfer learning, and fuzzy modeling in machine learning and data analytics.

Dr. Zhang received seven Australian Research Council (ARC) Discovery Projects grants and many other research grants. He received an ARC QEII fellowship in 2005. He has co-chaired several international conferences and workshops in the area of fuzzy decision-making and knowledge engineering. He has served as a member of the editorial boards of several international journals and a guest editor of eight special issues of the IEEE TRANSACTIONS and other international journals.