A Cross-Domain Recommendation Mechanism for Cold-Start Users Based on Partial Least Squares Regression

CHENG-TE LI, National Cheng Kung University, Taiwan
CHIA-TAI HSU and MAN-KWAN SHAN, National Chengchi University, Taiwan

Recommender systems are common in e-commerce platforms in recent years. Recommender systems are able to help users find preferential items among a large amount of products so that users' time is saved and sellers' profits are increased. Cross-domain recommender systems aim to recommend items based on users' different tastes across domains. While recommender systems usually suffer from the user cold-start problem that leads to unsatisfying recommendation performance, cross-domain recommendation can remedy such a problem. This article proposes a novel cross-domain recommendation model based on regression analysis, partial least squares regression (PLSR). The proposed recommendation models, PLSR-CrossRec and PLSR-Latent, are able to purely use source-domain ratings to predict the ratings for cold-start users who never rated items in the target domains. Experiments conducted on the Epinions dataset with ten various domains' rating records demonstrate that PLSR-Latent can outperform several matrix factorization-based competing methods under a variety of cross-domain settings. The time efficiency of PLSR-Latent is also satisfactory.

CCS Concepts: • **Information systems** → **Recommender systems**; *Data analytics*;

Additional Key Words and Phrases: Cross-domain recommendation, cold start, partial least square regression, transfer learning

ACM Reference format:
Cheng-Te Li, Chia-Tai Hsu, and Man-Kwan Shan. 2018. A Cross-Domain Recommendation Mechanism for Cold-Start Users Based on Partial Least Squares Regression. *ACM Trans. Intell. Syst. Technol.* 9, 6, Article 67 (October 2018), 26 pages.
https://doi.org/10.1145/3231601

1 INTRODUCTION

Online e-commerce services, such as Netflix, Spotify, and Amazon, highly rely on collaborative recommendation systems to suggest a list of relevant items for each user. The essential idea of recommender systems is exploiting observed ratings provided by users to learn user preferences on items and generating scores for the unrated items [1, 3, 4]. The performance of collaborative recommender systems highly depends on not only the richness but also the volume of observed ratings of users. However, not all users have abundant ratings, especially for new users who just

This work was sponsored by Ministry of Science and Technology of Taiwan (MOST) under Grants No. 107-2636-E-006-002, No. 106-3114-E-006-002, and No. 106-3114-E-001-004, and it was also supported by Academia Sinica under Grant No. AS-107-TP-A05.
Authors' addresses: C.-T. Li, Department of Statistics, No.1, University Road, Tainan City 701, Taiwan (R.O.C); email: chengte@mail.ncku.edu.tw; C.-T. Hsu and M.-K. Shan, Department of Computer Science, No.64, Sec.2, Zhi Nan Rd., Wenshan District, Taipei City 11605, Taiwan (R.O.C); emails: tonyqq6760@gmail.com, mkshan@nccu.edu.tw.

https://doi.org/10.1145/3231601

join the services. Making recommendation for users with rare or no observed ratings is the so-called "cold-start" problem, which is a critical and practical issue in commercial recommender systems [1, 16].

Cross-domain recommendation [5, 7] is one of the major approaches to tackle the cold-start problem. Cross-domain recommendation, compared with conventional single-domain recommendation, possesses several merits. The major advantages are being able to address the cold-start problem and to deal with the data sparsity problem in the rating data of the *target* domain. Specifically, in cross-domain recommendation, the rating data can be divided into *source* domain and *target* domain. The former provides the knowledge about the user-rating preferences, while the user rating scores on items in the latter are what we aim to infer. Since the target domains are usually presumed to suffer from insufficient ratings (especially on new users) that lead to the user cold-start problem and the rating sparsity problem, one of the well-known and effective solutions is to transfer the knowledge of rating preferences from the source domain that possesses rich ratings to remedy these problems. According to existing work [5], moreover, cross-domain recommendation can increase the novelty of recommended items and boost the accuracy of recommenders.

The cold-start problem in recommender systems is practical, since it is often that new users join e-commerce services. Due to the lack of the past rating or purchasing records, the preferences of new users cannot be well learned, and thus the recommendation quality is barely satisfactory. Although several existing studies have attempted to tackle the cold-start problem [28, 29, 34], what they consider is unrealistic, i.e., not pure cold-start. They either resort to the contextual information of ratings (e.g., the social network behind users, the tags associated with users and items, and the user profiles) or presume that new users possess a few ratings before the recommendation is performed. Since the recommendation demands are usually required right after new users join the services and the contextual information of ratings are either incomplete or not always accessible due to privacy concern, it is necessary to develop a reliable method that can lead to accurate recommendation performance under the pure cold-start setting. That said, the method is supposed to rely on no contextual information for new users without any ratings.

In this article, we develop a novel cross-domain recommendation mechanism based on the regression analysis, *Partial Least Squares Regression* (PLSR). We propose the *PLSR-CrossRec* model. The central idea is to exploit the users who had ever produced ratings in both source and target domain, to learn the rating knowledge. The knowledge to be learned and transferred is the set of *regression coefficients*, which relates source-domain ratings to target-domain ratings. Then, when any new user joins the target domain, we can use the learned regression coefficients to predict her ratings. Since the sizes of rating matrices are usually too large and thus lead to heavy computation cost in learning the transferring knowledge, we further devise an efficient algorithm, *PLSR-Latent*. PLSR-Latent aims at predicting the latent factors of users in the target domain, instead of directly predicting the target-domain rating scores. The regression coefficients are alternatively learned to relate the *latent factors* of user preferences and item categories in the source domain to the latent factors of users in the target domain. Because the dimensionality of latent factors is much lower than the original rating matrices, the computational complexity at the training stage can be significantly lowered down.

To evaluate the effectiveness of our proposed models, we conduct a number of experiments using Epinions.com datasets with 10 domains of user ratings. Experimental results deliver the following findings. First, PLSR-Latent can outperform typical matrix factorization-based cross-domain rating prediction methods. Second, the superiority of PLSR-Latent is consistently demonstrated whenever source and target domains share similar or distinct properties and when multiple source domains are applied at the same time. Third, PLSR-Latent can stably afford sparse source user-rating matrices with satisfying performance. Fourth, the time efficiency of PLST-Latent at the training stage is promising.

We summarize the contributions of this article in the following.

- We deliver a pioneering study that exploits the technique of partial least squares regression for cross-domain recommendation with cold-start users in the target domain. The tackled problem relies on only user ratings without contextual information such as user profiles and item content.
- We develop two methods, PLSR-CrossRec and PLSR-Latent, to efficiently learn the matrix of regression coefficients. The regression coefficients are designed and learned to relate the latent factors of users in the source domain to their ratings in the target domain.
- Experiments conducted on Epinions datasets with 10 different domains show that PLSR-Latent produces a superiority in predicting target-domain ratings over four competing methods. The evaluation in various cross-domain settings, homogeneous, heterogeneous, and compound, also proves the consistent and promising performance of PLSR-Latent.

The structure of this article is organized as below. We first review the relevant literatures in Section 2, followed by delivering the problem statement in Section 3. Section 4 describes the methodology, in which we first give an approach overview, then present the PLSR-CrossRec and PLSR-Latent methods in order. In Section 5, we conduct the evaluation and exhibit the experimental results. Section 6 concludes this work.

2 RELATED WORK

The study of recommendation systems typically aims to predict user ratings within a single domain and returns items with higher rating scores to the user. There are three most well-known approaches, content-based, collaborative filtering, and hybrid [1, 3]. Content-based recommendation encodes user profiles and items' textual contents as features, then compute the similarity between users and items in the feature space for recommendation [20]. Collaborative filtering takes advantage of the preferences between users based on the user-rating matrix. It can be divided into three solution types, user-based [4], item-based [27], and latent factor [13]. The latent factor approach had been validated to be widely promising. Its idea is to perform matrix factorization to find the latent factor for each user, in which each of the reduced dimensions implies a certain hidden concept. Similarity computation is applied in the space of latent factors to find the items that are most similar to the user's preference.

Cross-domain recommendation aims at using a rich source-domain user-rating information, from which we can learn user preferences and item concepts and transfer the learned knowledge to predict users' ratings in the target domain with sparse existing ratings. Cross-domain knowledge transferring brings several benefits to recommender systems [5, 7]: (a) dealing with the cold-start problem of users, (b) tackling the data sparsity problem for ratings, (c) increasing recommendation novelty, and (d) boosting the accuracy of recommendation. The approaches to cross-domain recommendation consists of two main categories, *knowledge aggregation* and *knowledge linkage* [5], which are illustrated in Figure 1. We briefly review both categories in the following.

Knowledge Aggregation. The idea of knowledge aggregation is to separately learn the rating knowledge from source domain K_{src} and the rating knowledge from target domain K_{trg}, and fuse K_{src} and K_{trg} to the aggregated knowledge $K_{src,trg}$. Then one can predict the ratings of users in the target domain based on the aggregated knowledge $K_{src,trg}$. In terms of how to extract knowledge, the knowledge aggregation approach can be further divided into three types.

- *Merging single-domain user preferences.* By merging multiple domains' user-ratings into an unified matrix, one can apply item-based collaborative filtering to predict the ratings in different domains. Cremonesi et al. [7] propose a weighted transitive closure mechanism to

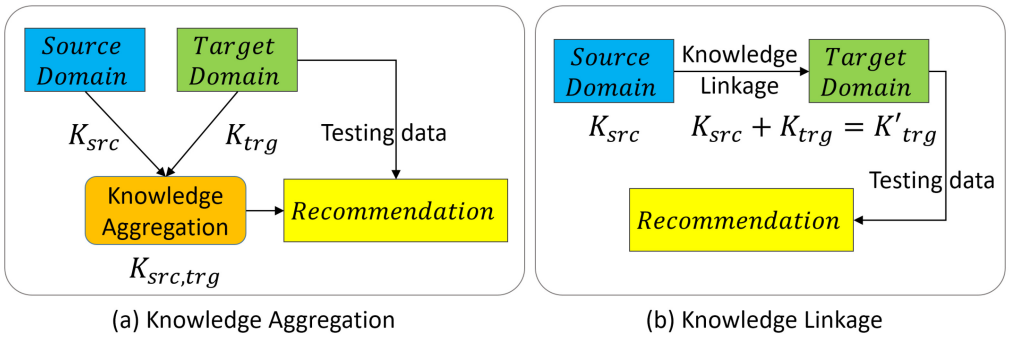

Fig. 1. Two mainstream approaches to cross-domain recommendation.

compute and magnify the similarity between cross-domain items. Zhao et al. [37] aim to model and match users' demographic information detected from their profiles with items' demographics learned from social media and online reviews. By fusing demographic information from different domains, a learning to rank algorithm is applied to generate the items for recommendation.

- *Mediating single-domain user modeling data.* Source-domain user modeling data, such as similar users and items (i.e., the neighborhood of user and item), can be learned together with the rating information to benefit the target-domain recommender. Pan et al. [23] propose to a *transfer by integrative factorization* model to incorporate *uncertain ratings* that represent the distribution of user preferences to boost the performance of target-domain rating prediction.
- *Combining single-domain recommenders.* By constructing the separate recommenders from rating data of source and target domain, one can properly combine their recommendation results to produce the final predicted target-domain ratings. Based on such an idea. Zhuang et al. [38] propose a *consensus regularization* framework for learning from multiple source domains to a target domain. In addition, Singh and Gordon [30] develop a collective matrix factorization (CMF) model that can simultaneously decompose matrices from pairwise domains sharing common users. However, their approach needs exactly the same user set in different domains, and thus cannot deal with cross-domain rating prediction for cold-start users who appear in the target domain and do not appear in the source domain in our work. Therefore, CMF is not suitable for our experimental comparison.

Knowledge Linkage. The idea of knowledge linkage is to transfer the source-domain rating knowledge K_{src}, along with the target-domain knowledge K_{trg}, to obtain the linked source-target knowledge K'_{trg} for predict the target-domain ratings of users. The knowledge linkage approach is also composed of three types, as below.

- *Linking domains.* Since sometimes the source domain is lack of obvious correlation to the target domain, one may resort to external information, such as Wikipedia[1] knowledge base for `Books` domain and IMDb[2] movie database for `Movies` domain, to strengthen the cross-domain connection. Zhang et al. [36] propose the **multi-domain collaborative filtering**

[1] https://en.wikipedia.org/wiki/Main_Page.
[2] http://www.imdb.com/.

to model external knowledge as new latent factors when transferring the rating preferences between domains.

- *Sharing latent features by domains.* Although different domains have no direct correlation in terms of items (e.g., `Books` vs. `Movies`), they may share common hidden concepts (e.g., `Books` and `Movies` share common item categories). By transforming source and target domains into the same latent space and considering the derived latent factors as hidden concepts, the rating knowledge of the source domain can be transferred to the target domain Pan et al. [22].
- *Transferring rating patterns between domains.* Although users and items across domains are different, close domains have potential to user preferences sampled from the same population. Li et al. [15] discover the *rating patterns* to model the latent correlation between preferences of user clusters for item clusters via the co-clustering technique. Then a *codebook transfer* model is developed to transfer the knowledge from the source to target domain based on the learned rating patterns. Their proposed codebook transfer (CBT) needs the cold-start users to have ratings in the target domain. However, in our problem setting, the cold-start users have no ratings in the target domain. Hence, CBT cannot be applied for our experimental comparison.

The most relevant study to our work is cross-domain clustering-based matrix factorization (CrossCBMF) [19], which is proposed to specially tackle the user cold-start problem in the target domain using only rating information. The predicted rating $R^\star$ of CrossCBMF consists of two predicted parts, clustering-based matrix factorization ratings $\hat{R}^c$ and typical matrix factorization ratings $\hat{R}$, given by: $R^\star = \alpha\hat{R}^c + (1-\alpha)\hat{R}$, where α is the weighting parameter. To derive the score $\hat{R}^c$, a *coarse* matrix is learned to represent the ratings of each user cluster on item clusters. Since a user can belong to various clusters in different domains, the coarse matrix is capable of transferring knowledge across domains. CrossCBMF had been validated to produce satisfying performance in **top-*N* recommendation**. However, as will be shown in Section 5, CrossCMBF cannot deliver good results in predicting target-domain ratings of cold-start users.

It is worthwhile to mention that we think both tasks of rating prediction and top-N recommendation are essential. As pointed by Steck [33], top-N recommendation is an important sub-problem of rating prediction. Improving the performance of rating prediction can better enhance the effectiveness of top-N recommendation. In addition, in real-world practical applications, commercial recommender systems need the predicted rating scores to convince users of the recommended items, because users' decision of purchasing also highly depends on the goodness of item ratings.

Multimodal Learning. The purpose of multimodal learning [32] is to learn the hidden relationships and generate a joint representation between different modalities, such as images, texts, and audios, to simultaneously depict the targeted item. Multimodal learning can be regarded as a kind of knowledge aggregation in the context of cross-domain recommendation as it can predict missing values across modalities of the targeted item. In the literature of multimodal learning, Nie et al. [21] predict future health statuses of patients by jointly learning from modalities, including historical health statuses, past visual scans and textual medical records, and domain knowledge of specific diseases. Qian et al. [24] develop a cross-domain collaborative learning framework based on non-parametric Bayesian dictionary learning model for multimodal data analysis and event detection. Farseev et al. [9] propose a cross-network collaborative recommendation framework, which jointly models the users preferences from multi-source social media data, to boost venue recommendation performance. Jia et al. [11] recommend videos to users by fusing multimodal knowledge from the textual descriptions of their Android applications and videos, and the extracted video content-based features. In short, multimodal learning aims to combine multi-domain

Table 1. Comparison of Relevant Studies with This Work

	K-Aggr	K-Link	S-Users	Cold-Start	C-Info	R-Pred
[30]	√		√			√
[36]		√				√
[37]	√		√		√	
[15]		√				√
[23]	√		√		√	√
[9]	√		√		√	
[19]		√	√	√		
This work		**√**	**√**	**√**		**√**

clues to jointly depict the targeted users or items. However, cold-start users cannot be benefited as they have no data in different modalities, and our work aims at predicting cross-domain ratings for cold-start users. In addition, our problem setting is more challenging, because, in real practice, the user profiles in multiple modalities are not available due to either privacy concern or profile incompleteness.

To clearly present the comparison between this work and some of representative existing studies, we create a table that lists several key points in cross-domain recommendation, as shown in Table 1. We compare six of relevant work discussed above in terms of approach categories, Knowledge Aggregation (**K-Aggr**) and Knowledge Linkage (**K-Link**), Shared Users (**S-Users**, whether it is required to have common users in both source and target domains), Using Content, Context and Profile Information of users or items (**C-Info**), and whether it is for rating prediction (**R-Pred**). It can be obviously observed that our work is different from previous studies: we aim to predict item ratings for target-domain cold-start users without using content, context, and profile information of users and items.

3 PROBLEM STATEMENT

Let D_{src} and D_{trg} be the training data of source domain and target domain, respectively. The user sets involved in D_{src} and D_{trg} are denoted by U_{src} and U_{trg}, respectively, where $U_{src} = U_{trg} = \{u_1, u_2, \ldots, u_n\}$. That said, D_{src} and D_{trg} share the same user set. The item sets for D_{src} and D_{trg} are denoted by I_{src} and I_{trg}, where $I_{src} = \{i_{src,1}, i_{src,2}, \ldots, i_{src,m}\}$ and $I_{trg} = \{i_{trg,1}, i_{trg,2}, \ldots, i_{trg,p}\}$. We can use two matrices, $\mathbf{X} \in \mathcal{R}^{n \times m}$ and $\mathbf{Y} \in \mathcal{R}^{n \times p}$ to represent the user-rating records in D_{src} and D_{trg}, respectively. For example, if user u_a had rated item $i_{src,b}$ with score r in D_{src}, it is denoted by $r = i_{src,a,b}$, and the corresponding matrix is $r = \mathbf{X}_{ab}$ for any user u_a and any item $i_{src,b}$.

Problem Definition. Given an user u_a who just joins a certain target-domain service, where $u_a \notin D_{trg}$ and $a > n$, and let x_a denote the vector that represents her rating records on items in D_{src}, where x_a is supposed to be a sparse vector. The goal of our cross-domain recommendation is to predict user u_a's rating on item $i_{trg,b}$ in D_{trg}, denoted by $r_{trg,a,b}$, where $b = [1, p]$.

4 METHODOLOGY

4.1 Approach Overview

We propose a novel cross-domain recommender system with two algorithms, *PLSR-CrossRec* and *PLSR-Latent*. Both methods aim at predicting user's rating scores on unrated items in the target domain, given her rating scores on items in the source domain and no rating records in the target

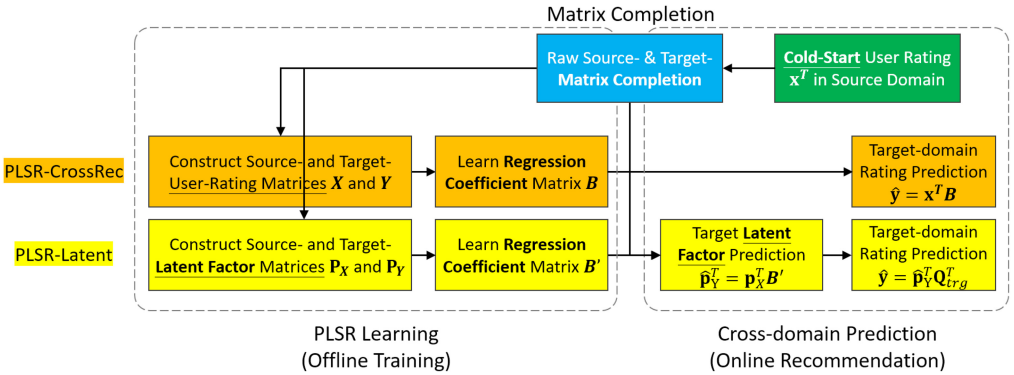

Fig. 2. Approach overview.

domain. *PLSR-Latent* is a more efficient variant of *PLSR-CrossRec*. The methodology consists of the following three phases, as illustrated by the picture of approach overview in Figure 2. We elaborate the details of such an overview in the following.

- **Phase 1: Matrix Completion.** For the missing values in both matrices of the source and the target domain, we plan to exploit the technique of Matrix Factorization (MF) [13] to initialize the unknown rating scores so that the rating matrices $\mathbf{X}$ and $\mathbf{Y}$ can be completed.
- **Phase 2: PLSR Learning.** We aim at finding the set of users who have ever performed rating actions in both the source and the target domain. Their corresponding ratings in source and target domains are also identified. Then, we construct a *Partial Least Squares Regression*-based rating prediction model, which is capable of transferring their rating preferences from the source domain to the target domain.
- **Phase 3. Cross-domain Prediction.** When making the recommendation, based on the user's existing rating records in the source domain, the predictive model is used to predict the rating score for any item in the target domain.

The key idea of the proposed *PLSR-CrossRec* is the technique of *Partial Least Squares* (PLS), which is a kind of multivariate analysis method in Statistics and is commonly used in the area of Chemometics. PLS is quite suitable for modeling the relationship between two variables (i.e., features) even they are not comparable. Specifically, given two incomparable variables, the PLS technique is able to transform both variables into a low-dimensional space so that they can compare with one another and their similarity can be derived. For example, music songs in Spotify and movie videos in NetFlix come from different domains that lead to distinct feature spaces, and thus are incomparable with each other. The reason may be they possess different feature sets, such as Mel-frequency cepstral coefficients (MFCC), beat, and velocity for music and Scale-invariant feature transform (SIFT), Histogram of oriented gradient (HOG), and optical flow for movie videos. By converting music and movies into a single low-dimensional space via some feature transformation powered by PLS, one is allowed to match movies with music songs. We leverage PLS to perform cross-domain recommendation by considering users' ratings as feature values and mapping variables into the same space so that knowledge in both domains can be transferred. In this work, we propose to exploit *Partial Least Squares Regression* (PLSR) [25] to realize PLS for cross-domain recommendation.

4.2 Matrix Completion

Since the techniques of PLS can be applied to matrices without missing values, we need to perform some preprocessing to complete the rating matrices $\mathbf{X}$ and $\mathbf{Y}$ in both source and target domains given they are sparse. Since our goal is to do cross-domain recommendation, we adopt the most well-known recommendation method, Matrix Factorization (MF) [13], to fill in the missing values of rating matrices $\mathbf{X}$ and $\mathbf{Y}$. The intuition of MF is to learn k latent factors that reflect user preferences or item categories based on historical rating records of users (training data). Then, we can take advantage of the inner product of user latent factor and item latent factor to derive the corresponding potential rating scores. Such obtained rating scores will be treated as filled values to complete the rating matrices.

Given an incomplete rating matrix $\mathbf{X}$ (or $\mathbf{Y}$), we can learn the latent factors of users and items, denoted by $\hat{\mathbf{P}}$ and $\hat{\mathbf{Q}}$, respectively, through the following optimization equation:

$$\arg\min_{\hat{\mathbf{P}},\hat{\mathbf{Q}}} \sum_{(a,b)\in J} \|\mathbf{X}_{ab} - (\hat{\mathbf{P}}_{a,\cdot}) \cdot (\hat{\mathbf{Q}}_{b,\cdot})^T\|^2, \tag{1}$$

where each row in $\hat{\mathbf{P}}$ and $\hat{\mathbf{Q}}$ represents the latent factors of users and items, respectively, $\hat{\mathbf{P}} \in \mathcal{R}^{n\times k}$ and $\hat{\mathbf{Q}} \in \mathcal{R}^{m\times k}$, k is the number of latent factors (i.e., the number of dimensions), J is the set of all *non-unrated* user-item pairs in $\mathbf{X}$: $(a, b) \in J$ if $\mathbf{X}_{ab}$ is not a missing value. By regarding all of existing ratings as training data, we can derive the user latent factor $\mathbf{P}_{a,\cdot}$ and the item latent factor $\mathbf{Q}_{b,\cdot}$ via minimizing the error between the existing rating scores and the estimated rating scores in Equation (1). Eventually, we can fill in the missing user-item rating values based on the derived latent factors $\hat{\mathbf{P}}$ and $\hat{\mathbf{Q}}$. The rating matrix completion can be done by

$$\mathbf{X}_{ab} = (\hat{\mathbf{P}}_{a,\cdot}) \cdot (\hat{\mathbf{Q}}_{b,\cdot})^T, \forall (a, b) \in U_{src} \times I_{src}, (a, b) \notin J. \tag{2}$$

The missing values in the incomplete target-domain rating matrix $\mathbf{Y}$ can be filled in using the MF method.

4.3 PLSR-based Cross-domain Learning

We aim to extend and modify the technique of *Partial Least Squares Regression* (PLSR) [25] for cross-domain recommendation. PLSR uses the aforementioned properties of PLS to perform regression analysis. PLSR can learn the *regression coefficients* from training data. Specifically, from the perspective of regression analysis, the vectors of regression coefficients can be learned from as *predictor* representation and are used to regress the *response* variables. We propose to consider PLSR in cross-domain recommendation. Recall that we are given source-domain ratings (i.e., training data), and the goal is to predict the target-domain ratings. We can learn the *regression coefficients* by considering the derived latent space (i.e., the reduced dimensions by PLS) from the source-domain training ratings as the predictor representation, and treating the target-domain ratings as the responses. The key in this work is exploiting the derived latent matrix (i.e., the dimension-reduced space) as the predictor, rather than the original source-domain matrix, so that the knowledge shared by both domains can be learned and transferred.

Given two user-item rating matrices, $\mathbf{X} \in \mathcal{R}^{n\times m}$ and $\mathbf{Y} \in \mathcal{R}^{n\times p}$, there are n users who are shared in both source and target domain, in which the source domain contains m items and the target domain possesses p items. The rows in both matrices have correspondence, i.e., row i in $\mathbf{X}$ and $\mathbf{Y}$ refers to the same user. In the following, we first describe how PLS works in our cross-domain regression analysis, and present the new PLSR-based cross-domain recommender.

4.3.1 PLS in Cross-domain Regression Analysis. In this section, we first describe the fundamental knowledge and formulation of partial least squares (PLS), and explain how to extend the

PLS-based dimension reduction from one dimension to multiple dimensions with mathematical notations. This section is the preliminary for our proposed methods in the follow-up sections.

The original idea of PLS is to find a pair of basis (i.e., two directions), $\mathbf{r} \in \mathcal{R}^{m\times 1}$ and $\mathbf{s} \in \mathcal{R}^{p\times 1}$, such that after each user's rating records in $\mathbf{X}$ and $\mathbf{Y}$ are projected into $\mathbf{r}$ and $\mathbf{s}$ with derived vectors $\mathbf{Xr}$ and $\mathbf{Ys}$, the *covariance* value between $\mathbf{Xr}$ and $\mathbf{Ys}$ is maximized, given by

$$(\mathbf{r}', \mathbf{s}') = \arg\max_{|\mathbf{r}|=|\mathbf{s}|=1} cov(\mathbf{Xr}, \mathbf{Ys}). \tag{3}$$

The weight vectors $\mathbf{r}$ and $\mathbf{s}$ can project users and ratings in the source and target domains D_{src} and D_{trg} into the one-dimensional space with the maximum covariance value, in which the raw rating records of each user are reduced to an one-dimensional latent feature. In other words, by the linear transformation with $\mathbf{r}$ and $\mathbf{s}$, each user's ratings can be combined into a certain numerical value, and these values ($\mathbf{Xr}$ and $\mathbf{Ys}$) of all users in source and target domains, respectively, can lead to the maximum covariance.

We extend this PLS-based dimension reduction method from one dimension to k dimensions ($k > 1$), denoted by $(\mathbf{r}_1, \mathbf{s}_1), (\mathbf{r}_2, \mathbf{s}_2), \ldots, (\mathbf{r}_k, \mathbf{s}_k)$, so that more latent user preferences and more latent item categories can be learned. We can obtain the second dimension $(\mathbf{r}_2, \mathbf{s}_2)$ based on Equation (3) under the constraint that $(\mathbf{r}_1, \mathbf{s}_1)$ and $(\mathbf{r}_2, \mathbf{s}_2)$ are orthogonal. By analogy, the solution to the kth dimension $(\mathbf{r}_k, \mathbf{s}_k)$ based on Equation (3) under the constraint that $(\mathbf{r}_1, \mathbf{s}_1)$, $(\mathbf{r}_2, \mathbf{s}_2)$, ..., and $(\mathbf{r}_{k-1}, \mathbf{s}_{k-1})$ are orthogonal.

Let $\mathbf{T} = \mathbf{X\Omega}$ and $\mathbf{U} = \mathbf{Y\Gamma}$, where $\Omega = [\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_k]$ and $\Gamma = [\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_k]$. It is apparent that $\mathbf{T}$ and $\mathbf{U}$ are the results of dimension reduction from the source-domain and target-domain matrices $\mathbf{X}$ and $\mathbf{Y}$, respectively. Based on these notations and formulation, in the next Section 4.3.2, we aim to learn the matrix of regression coefficients $\mathbf{G}$ by considering $\mathbf{T}$ as the predictor and $\mathbf{Y}$ as the response such that $\mathbf{Y} = \mathbf{TG}$. Given the derived regression coefficients $\mathbf{G}$, when there is a user who is new in the target domain (i.e., no ratings in D_{trg}) and has some existing ratings in the source domain D_{src}, we can use $\mathbf{G}$ to predict her ratings on all items in the target domain via $\mathbf{Y} = \mathbf{TG}$.

Here, we give a toy example to elaborate how to perform the cross-domain rating prediction. Assume that there are 300 users who rate 100 movies in D_{src} and 80 movies in D_{trg}. First, our method can learn $k = 5$ latent factors for all users in the source and $k = 5$ latent factors for all users in the target domains. Second, we can learn the regression coefficients $\mathbf{G}$ based on users' latent factors in D_{src} and their existing ratings in D_{trg}. Third, equipped with $\mathbf{G}$, we can predict such new user's ratings on all items in the target domain, even she is a cold-start user in the target domain.

4.3.2 An Extended PLSR for Cross-domain Recommendation. Based on the aforementioned settings about PLS and PLSR, in this section, we aim at learning the regression coefficients from training data by considering source-domain ratings as the predictor and target-domain ratings as the response. First, we can decompose the source-domain and target-domain rating matrices by the following two equations:

$$\mathbf{X} = \mathbf{TP}^T + \mathbf{E}, \tag{4}$$

$$\mathbf{Y} = \mathbf{UQ}^T + \mathbf{F}, \tag{5}$$

where each row in $\mathbf{T}$ and $\mathbf{U}$ is a *latent factor* (or called *score vector*), each row in $\mathbf{P}$ and $\mathbf{Q}$ is called a *loading vector*, and $\mathbf{E}$ and $\mathbf{F}$ are the matrices of residuals. From the perspective of dimension reduction, matrices $\mathbf{T}$ and $\mathbf{U}$ are the results of dimension reduction from matrices $\mathbf{X}$ and $\mathbf{Y}$. To derive $\mathbf{T}$, $\mathbf{U}$, $\mathbf{P}$, and $\mathbf{Q}$ by solving Equations (4) and (5), such that the matrix factorization equation Equation (1) can be minimized, we take advantage of the well-known nonlinear iterative partial least squares (NIPALS) algorithm [35].

Let $\mathbf{D}$ be a linear transformation matrix ($\mathbf{D}$ is a $p \times p$ diagonal matrix) between matrices $\mathbf{T}$ and $\mathbf{U}$. We can have the following equation:

$$\mathbf{U} = \mathbf{T}\mathbf{D}^T + \mathbf{H}, \tag{6}$$

where $\mathbf{H}$ is the matrix of residuals. By putting Equation (6) into Equation (5), we can have the following equation:

$$\mathbf{Y} = \mathbf{T}\mathbf{D}\mathbf{Q}^T + (\mathbf{H}\mathbf{Q}^T + \mathbf{F}). \tag{7}$$

Let $\mathbf{C}^T = \mathbf{D}\mathbf{Q}^T$, Equation (7) can be rewritten as

$$\mathbf{Y} = \mathbf{T}\mathbf{C}^T + \mathbf{F}^\star, \tag{8}$$

where $\mathbf{F}^\star = \mathbf{H}\mathbf{Q}^T + \mathbf{F}$ is the residual matrix. Subsequently, we can find that $\mathbf{C}$ is the matrix of regression coefficients, which can be used to predict target-domain rating matrix $\mathbf{Y}$ based on the source-domain latent factor matrix $\mathbf{T}$. In other words, Equation (8) can be treated as the rating prediction scores $\mathbf{Y}$ in the target domain. We can say that Equation (8) decomposes matrix $\mathbf{Y}$ using ordinary least squares regression with orthogonal predictors $\mathbf{T}$.

The item recommendation for a new target-domain user consists of two steps, given she has some existing source-domain ratings. The first is learning her latent factor via dimension reduction. The second is learning the matrix of regression coefficients, which will be further used to predict her ratings on target-domain items. Now, we aim at combining such two steps together to facilitate the computation. Since $\mathbf{T}$ is an orthogonal matrix, i.e., $\mathbf{T}^T\mathbf{T} = \mathbf{I}$, we can have the following equation based on Equation (8):

$$\mathbf{C} = \mathbf{Y}^T\mathbf{T}. \tag{9}$$

Then, we can leverage the relationship proposed in Reference [17] to redefine Equation (8) in terms of the original predictors $\mathbf{X}$, given by

$$\mathbf{T} = \mathbf{X}\mathbf{W}(\mathbf{P}^T\mathbf{W})^{-1}, \tag{10}$$

where $\mathbf{W}$ is a weight matrix, $\mathbf{W} = \mathbf{X}^T\mathbf{U}$, and $\mathbf{P}$ is the loading vectors defined in Equation (4). By plugging relations Equations (9) and (10) into Equation (8), we yield

$$\mathbf{Y} = \mathbf{X}\mathbf{B} + \mathbf{F}^\star, \tag{11}$$

where $\mathbf{B}$ is the new matrix of regression coefficients. Equation (11) is the key equation in our method, which represents that we can directly use a user's ratings in the source domain $\mathbf{D}_{src}$ to predict her ratings on items in the target domain $\mathbf{D}_{trg}$. Eventually, we can compute the new matrix of regression coefficients via

$$\mathbf{B} = \mathbf{W}(\mathbf{P}^T\mathbf{W})^{-1}\mathbf{C}^T = \mathbf{X}^T\mathbf{U}(\mathbf{T}^T\mathbf{X}\mathbf{X}^T\mathbf{U})^{-1}\mathbf{T}^T\mathbf{Y}. \tag{12}$$

From Equation (12), it is apparent that we can derive the regression coefficients $\mathbf{B}$ using four matrices $\mathbf{X}$, $\mathbf{Y}$, $\mathbf{T}$, and $\mathbf{U}$, where $\mathbf{X}$ and $\mathbf{Y}$ are the input source-domain and target-domain ratings (i.e., training data), and $\mathbf{T}$ and $\mathbf{U}$ are matrices of latent factors derived by the NIPALS algorithm. Therefore, the matrix of regression coefficients $\mathbf{B}$ can be learned at the training stage.

When making recommendation, we can use the learned regression coefficients $\mathbf{B}$, along with users' source-domain ratings $\mathbf{X}$, to predict their ratings $\hat{\mathbf{Y}}$ on target-domain items. This can be expressed by

$$\hat{\mathbf{Y}} \approx \mathbf{Y} = \mathbf{X}\mathbf{B}. \tag{13}$$

That said, using $\mathbf{T}$ to predict $\mathbf{Y}$ (Equation (8)) can be replaced by using $\mathbf{X}$ to have the predicted $\mathbf{Y}$, in which the old version regression coefficient matrix $\mathbf{C}$ is rewritten as $\mathbf{B}$. In short, equipped with PLS regression and the defined equations, we can simultaneously learn the latent factors and predict the target-domain ratings in a single matrix $\mathbf{B}$. Given a set of new users with their ratings

in the source domain $\mathbf{X}_{new}$, we can use the learned $\mathbf{B}$, we can predict their ratings in the target domain $\hat{\mathbf{Y}}$ via $\hat{\mathbf{Y}} = \mathbf{X}_{new}\mathbf{B}$.

Discussion. We choose to extend PLS regression for cross-domain rating prediction, because its learned regression coefficients can transfer rating knowledge from the source to the target domain. Although some general multiple regression techniques [2, 8] can be used for recommender systems, such as directly applying linear regression to learn the regression coefficients from $\mathbf{X}$ to $\mathbf{Y}$, such methods cannot learn the latent factors of user preferences and item concepts, which can be modeled using PLSR. In other words, what we need is to learn the knowledge that commonly shared by both source and target domains. Through Equation (3), the derived latent factors can reflect the dimensions with higher covariance and higher correlation between domains, and thus the most useful knowledge can be kept and used to predict the target-domain ratings of cold-start users. However, although conventional dimension reduction techniques, such as Principle Component Analysis [12] and Latent Semantic Indexing [5], can be used to learn the most informative dimensions from rating data, the learned latent factors can only belong to separate rating domain, instead of those shared by both source and target domains. Hence, such methods are not suitable to be directly applied for cross-domain recommendation.

4.3.3 The PLSR-CrossRec Algorithm. Here, we present the entire algorithm of the proposed PLSR method. It has two stages, *offline training* and *online recommendation*. Given a set of source-domain users U_{src}, a set of source-domain items I_{src}, a set of target-domain users U_{trg}, and a set of target-domain items I_{trg}, the offline training is devised to learned to matrix of regression coefficients.

The offline training consists of three steps, elaborated as below.

(1) Construct the raw source-domain user-item rating matrix $\mathbf{M}_{src}$ from U_{src} and I_{src}, and construct the raw target-domain user-item rating matrix $\mathbf{M}_{trg}$ from U_{trg} and I_{trg}. Then apply the Matrix Factorization method to complete such two matrices $\mathbf{M}_{src}$ and $\mathbf{M}_{trg}$.
(2) Find the set of users who *commonly appear* in both the source and the target domain, along with their corresponding ratings. Based such common users and their ratings, we construct the source-domain rating matrix $\mathbf{X}$ and the target-domain rating matrix $\mathbf{Y}$.
(3) Learn the source-domain and target-domain latent factor matrices, $\mathbf{T}$ and $\mathbf{U}$, by the NIPALS algorithm [35]. Then using the proposed extended PLS regression technique with Equation (12), together with the derived $\mathbf{X}$, $\mathbf{Y}$, $\mathbf{T}$, and $\mathbf{U}$, the matrix of regression coefficients $\mathbf{B}$ can be learned. $\mathbf{B}$ is considered as the transformation matrix from source to target domain.

The online recommendation stage aims at predicting the new user's rating scores on target-domain items, then recommending items with highest ratings to the user. It consists of the following two steps.

(1) Given a new user, along with all of her ratings in the source domain, represented by $\mathbf{x}^T$, where columns in both $\mathbf{x}^T$ and $\mathbf{X}$ refer to the same set of source-domain items, we complete $\mathbf{x}^T$ via matrix factorization by appending to $\mathbf{X}$ if $\mathbf{x}^T$ contains missing ratings.
(2) Predict the new user's ratings $\hat{\mathbf{y}}$ in the target domain based on Equation (13), i.e., $\hat{\mathbf{y}} = \mathbf{x}^T\mathbf{B}$. Then recommend target-domain items with highest ratings $\hat{\mathbf{y}}$ to the new user.

It can be apparently found that our PLSR-CrossRec algorithm requires no rating information of new users in the target domain. What we need is only their ratings in the source domain. In other words, for new users in the target domain, the proposed method can solve the cold-start problem and potential target-domain items can be recommended to new users.

4.4 A More Efficient Solution: PLSR-Latent

The proposed PLSR-CrossRec method in Section 4.3.3 needs source-domain and target-domain rating matrices $\mathbf{X}$ and $\mathbf{Y}$ at the training stage. The running time of learning the matrix of regression coefficients $\mathbf{B}$ highly depends on the sizes of $\mathbf{X}$ and $\mathbf{Y}$. The time complexity of the NIPALS algorithm [35] is $O(n(m+p))$, where n is the number of rows of user-rating matrices $\mathbf{X}$ and $\mathbf{Y}$ (i.e., number of users), and m and p are the numbers of columns in source-domain matrix $\mathbf{X}$ and target-domain matrix $\mathbf{Y}$, respectively (i.e., numbers of items in such two domains). In the deployment of the cross-domain recommendation in the real world, the numbers n, m, and p are very large. Consequently, it will talk unacceptable running time to executing the NIPALS algorithm so that the matrices of latent factors $\mathbf{T}$ and $\mathbf{U}$ can be derived and used in Equation (12). In this section, therefore, we propose a more efficient solution, PLSR-Latent, to speedup the training phase.

The PLSR-Latent method is developed based on the PLSR-CrossRec. The central idea is exploiting the matrices of latent factors in both source and target domains to learn the matrix of regression coefficient in the PLS regression. Specifically, instead of using the completed rating matrices $\mathbf{X}$ and $\mathbf{Y}$, whose dimensions are too large, we take advantage of the derived latent factors from matrix factorization, denoted by $\mathbf{P}_X$ and $\mathbf{P}_Y$, to compute the regression coefficients $\mathbf{B}$. The latent factor matrices, $\mathbf{P}_X$ and $\mathbf{P}_Y$, can be obtained from matrix factorization, i.e., Equation (1), using the original source-domain and target-domain rating matrices, respectively. That said, $\mathbf{X} \approx \mathbf{P}_X\mathbf{Q}_X$ and $\mathbf{Y} \approx \mathbf{P}_Y\mathbf{Q}_Y$. Then based on Equation (12), we can obtain a new matrix of regression coefficients, given by

$$\mathbf{B}' = \mathbf{P}_X^T\mathbf{U}'(\mathbf{T}'^T\mathbf{P}_X\mathbf{P}_X^T\mathbf{U}')^{-1}\mathbf{T}'^T\mathbf{P}_Y, \tag{14}$$

where $\mathbf{P}_X$ and $\mathbf{P}_Y$ contains the numbers of latent dimensions K_X and K_Y, whose values are usually small. Here, we choose $K_X = K_Y$. Using latent factors $\mathbf{P}_X$ and $\mathbf{P}_Y$ to compute $\mathbf{B}'$ would lead to a similar effect as using $\mathbf{X}$ and $\mathbf{Y}$ to obtain $\mathbf{B}$. It is because both ways are the same to model user preferences (using exact items in the latter and using latent factors in the former). With this formulation, the role of matrix $\mathbf{B}'$ is accordingly changed to *predict the latent factors* of new users in the target domain, rather than using the original $\mathbf{B}$ to predict their rating scores in the target domain.

Here, in the following, we provide the detailed procedure of PLSR-Latent, which is basically similar with PLSR-CrossRec. The offline training consists of three steps.

(1) Use Matrix Factorization to decompose the original source-domain and target-domain matrices $\mathbf{M}_{src}$ and $\mathbf{M}_{trg}$, i.e., $\mathbf{M}_{src} \approx \mathbf{P}_{src}\mathbf{Q}_{src}$ and $\mathbf{M}_{trg} \approx \mathbf{P}_{trg}\mathbf{Q}_{trg}$.
(2) Find the set of users who *commonly appear* in both the source and the target domain, along with their corresponding latent vectors. Then, we can have the source-domain latent factor matrix $\mathbf{P}_X$ and the target-domain latent factor matrix $\mathbf{P}_Y$.
(3) Learn matrices $\mathbf{T}'$ and $\mathbf{U}'$ based on $\mathbf{P}_X$ and $\mathbf{P}_Y$ by the NIPALS algorithm [35]. Then using the proposed extended PLS regression technique with Equation (14), together with the derived $\mathbf{P}_X$, $\mathbf{P}_Y$, $\mathbf{T}'$, and $\mathbf{U}'$, the matrix of regression coefficients $\mathbf{B}'$ can be learned via Equation (14).

The online recommendation stage also consists of three steps.

(1) Given a new user, along with all of her ratings in the source domain, represented by $\mathbf{x}^T$, we use matrix factorization to obtain her latent factor $\mathbf{p_X}^T$.
(2) Based on the derived $\mathbf{B}'$, we can predict the new user's latent factor in the target domain by $\hat{\mathbf{p}}_\mathbf{Y}^T = \mathbf{p_X}^T\mathbf{B}'$.

Table 2. The Top-10 Domains with Most Numbers of Ratings and Their Data Statistics

Index	Domain Name	No. of Users	No. of Items	No. of Ratings	Density
21	Videos & DVDs	25,288	28,972	175,665	0.0240%
7	Books	15,504	59,488	108,879	0.0118%
18	Music	16,001	35,806	96,225	0.0168%
10	Online Store & Services	28,643	5,518	54,734	0.0346%
23	Toys	9,040	18,680	51,151	0.0303%
12	Destinations	9,290	3,615	31,418	0.0936%
57	Personal Care	6,214	10,786	28,945	0.0432%
48	Used Cars	17,041	4,174	28,598	0.0402%
54	Baby Care	5,422	3,165	21,340	0.1244%
97	Sport and Outdoor	6,750	9,597	19,181	0.0296%

(3) Predict the new user's ratings $\hat{\mathbf{y}}$ in the target domain by $\hat{\mathbf{y}} = \hat{\mathbf{p}}_{\mathbf{Y}}^T \mathbf{Q}_{trg}^T$. Then recommend target-domain items with highest ratings $\hat{\mathbf{y}}$ to the new user.

The time complexity of PLSR-Latent will be highly reduced, since the sizes of input matrices of PLSR have become $m \times k$ and $m \times k$, instead of $m \times n$ and $m \times p$, where $k = K_X = K_Y$, $k \ll n$, and $k \ll p$. Owing to the small-sized matrices, the computation efficiency of the NIPALS algorithm will be significantly boosted.

5 EVALUATION

To evaluate the effectiveness and efficiency of the proposed methods, we conduct a series of experiments based on a collection of cross-domain rating data. The experiments are designed to answer the following questions. (1) Can the proposed PLSR-Latent method lead to better performance of cross-domain rating prediction than typical matrix factorization-based methods? (2) How will PLSR-Latent perform given source and target domains share similar and distinct characteristics of ratings? (3) Can the performance of PLSR-Latent get further boosted by considering multiple source domains together? (4) To what extent the elapsed time of PLSR-Latent can be reduced and derived, compared with PLSR-CrossRec?

5.1 Dataset and Its Settings

Dataset. We employ the Epinions dataset [18] to conduct all of the experiments. The dataset contains users' ratings on all kinds of items in a general consumer review site Epinions.com. Each record of the dataset contains user ID, item ID, and user-item rating. Each rating score r is an integer, ranging from 1 to 5 ($r \in [1, 5]$). There are 131,228 users, 317,755 items, and 1,127,673 ratings in total. Each item exactly belongs to a certain domain. Domains in this dataset include Books, Music, Videos, Cars, Sports, and so on. Totally, there are 587 domains. We use the top-10 domains containing the most numbers of ratings in the experiments. The detailed data statistics of such 10 domains are listed in Table 2. Note that in this table, the value of Density is calculated via $Density = \frac{\#Ratings}{(\#Users)\times(\#Items)}$.

Data Partitioning. We need to divide all users and their ratings into training and testing sets. The basic idea is that both training and testing sets are required to contain a set of users who have ratings in both of the source and the target domains. Specifically, all users in the dataset can be categorized into three types.

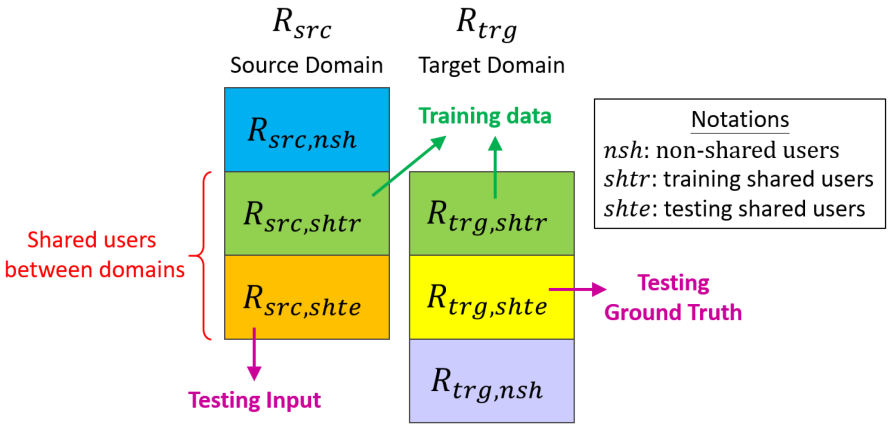

Fig. 3. An illustration on how users are partitioned into training and testing sets.

Table 3. Numbers of Shared Users between Domains Used in the Experiments

	Books	Music	Videos & DVDs	Toys	Personal Care	Used Cars
Books						
Music	6,102					
Videos & DVDs	9,614	9,081				
Toys	4,142	2,800	4,984			
Personal Care	2,734	1,923	2,949	1,900		
Used Cars	4,166	3,692	6,170	2,923	1,798	

(1) *Shared Users*, denoted by U_{sh}, refer to those users who have ever rated items in both source and target domains.
(2) *Source Non-shared Users*, denoted by $U_{src,nsh}$, indicate those users who have only rated source-domain items.
(3) *Target Non-shared Users*, denoted by $U_{trg,nsh}$, stand for those who have only rated target-domain items.

It is apparent that every user in the dataset exactly belongs to one of these three user types.

We exploit *fivefold cross-validation* in the experiments. All of the users are equally divided into five sets. One is used as the set of testing users, denoted by $U_{shte} \subseteq U_{sh}$, and the remaining four are considered as the set of training users, denoted by $U_{shtr} \subseteq U_{sh}$. The entire training set is comprised by the users $U_{shtr} \cup U_{src,nsh} \cup U_{trg,nsh}$ and their corresponding ratings. The source-domain ratings of the user set U_{shte} are treated as the input of testing data. And the target-domain ratings of users U_{shte} are considered as the ground truth. We create Figure 3 as an illustration to exhibit how users are partitioned into training and testing sets. Here, we report the number of shared users between combinations of domains that are used in the following experiments in Table 3.

5.2 Evaluation Settings

To evaluate the performance of the proposed methods, we consider two evaluation metrics that are widely used in examining the effectiveness of recommender systems. One is *Mean Absolute*

Error (MAE), and the other is *Root Mean Square Error* (RMSE). The formulas of MAE and RMSE are provided as follows:

$$MAE = \frac{1}{|T|} \sum_{(u,i) \in T} |\hat{r}_{ui} - r_{ui}|, \tag{15}$$

$$RMSE = \frac{1}{|T|} \sqrt{\sum_{(u,i) \in T} (\hat{r}_{ui} - r_{ui})^2}, \tag{16}$$

where T is the testing set containing all user-item ratings in the test data, r_{ui} is the actual rating score (ground truth), and $\hat{r}_{ui}$ is the predicted rating score by a certain method. The difference between r_{ui} and $\hat{r}_{ui}$ refers to the error between prediction and ground truth. Therefore, lower values of MAE and RMSE indicate better performance.

5.3 Competing Methods

To validate the recommendation performance of the proposed methods, we aim to make the comparison with several competing methods. Since our goal is to do cross-domain recommendation, the competing methods are required to meet the following two conditions. First, the competing methods must be capable of tackling the cold-start problems of user ratings in the target domain. Second, rating information is the only input of a certain method. That said, the competing methods need to accept ratings as the sole input, and do not rely on other meta data, such as tags, user profiles, social networks, and item categories. The assumption is that such meta data is not always accessible in the real world. We compare the proposed PLSR-Latent with the following four competing methods. Note that in our PLSR-Latent, there are two important parameters. One is the number K_{mf} of preserved dimensions in matrix factorization while the other is the number K_{pls} of preserved dimensions in the partial least squares. We will empirically report how K_{pls} and K_{mf} values affect on the performance. We set $K_{mf} = K_{pls} = 50$ by default in PLSR-Latent.

- **CrossMF.** Matrix factorization (MF) [13] is one of the popular collaborative filtering techniques for single-domain recommender systems. The idea of MF is to learn the latent factors of users and items from training data, and accordingly make the rating prediction. To extend the conventional MF technique for cross-domain recommendation, we combine both training and testing data, and combine both source and target domains. An unified cross-domain user-rating matrix is then constructed. By factorizing this unified matrix, we can derive the latent factors, then accordingly obtain the complete matrix, in which new users' ratings in the target domain can be derived. We call this method CrossMF. Since we empirically find that the number of dimensions $K = 5$ leads to the best performance in a variety of domain pairs, we choose to set $K = 5$ in CrossMF for all of the experimental comparison.
- **CrossNMF.** Non-negative matrix factorization (NMF) [14] is one of the variants of matrix factorization techniques. NMF adds a constraint when factorizing the matrix. The constraint is enforcing all the values in the latent factors to be non-zero values. The reason of making such a constraint is that negative values in the latent factors cannot preserve the physical meaning of each dimension, i.e., difficult to be explained. We employ the strategy similar with CrossMF to modify NMF for cross-domain recommendation. That said, we apply NMF to factorize an unified matrix combining training and testing data and source- and target-domain data. We call this method CrossNMF. Likewise, we set the number of dimensions $K = 5$, since it empirically leads to the best performance.
- **CrossCBMF.** Clustering-based matrix factorization (CrossCBMF) [19] is the state-of-the-art method that purely considers user-item rating information for cross-domain recommendation and for cold-start users in the target domain. The central idea of CrossCBMF is

leveraging the partially overlapped users and items between multiple domains to transfer their cluster-level preferences as auxiliary sources of information so that the cross-domain latent factors can be learned. There are a number of parameters in CrossCBMF. After several empirical tests, we find CrossCBMF has two of the most influential parameters: w_m and α. The parameter $w_m \in [0, 1]$ determines the contribution of unobserved ratings in the conventional matrix factorization. The parameter $\alpha \in [0, 1]$ controls the weight of clustering-based matrix factorization in predicting the target-domain ratings. Since CrossCBMF does not report the sensitivity of such two parameters, we implement and conduct CrossCBMF, and test how they affect the performance. We find different w_m values have no impact on the performance. Hence, we set $w_m = 0$. In addition, we also find higher α values lead to better effectiveness. To have the most competing method, we set $\alpha = 1$ in the following experiments.

- **CrossUNN.** Cross-domain User-based Nearest Neighbor (CrossUNN) [10] is another state-of-the-art method for cross-domain item recommendation. CrossUNN had been shown that incorporating additional ratings from the source domain can boost the performance of recommendation in the target domain. The key idea of CrossUNN is exploiting *Jaccard Coefficient* to find the most relevant source-domain users for each target-domain user by considering source-domain rated items. Then, *user-based k nearest-neighbors* is applied to estimate the rating scores in the target domain. In the following experimental comparison, we follow CrossUNN to set $k = 50$.

Note that it can be found that for CrossMF and CrossNMF, K is set to a very small value of 5. We think that because our rating prediction task is cross-domain, MF and NMF needs more and fewer "general" concepts to cover hidden common topics in both domains. Therefore, comparing to conventional MF and NMF with larger K, the K values in CrossMF and CrossNMF are relatively small.

5.4 Experiments and Results

5.4.1 Homogeneous Cross-domain Rating Prediction. In this section, we aim at examining the recommendation performance for multiple domains with similar characteristics. We refer to this experiment as *homogeneous cross-domain rating prediction*. Suppose that we are transferring the rating knowledge from a source domain A to a target domain B. This experiment consider domain A and B share similar rating preferences or item categories. Similar domains are supposed to have strong correlation between user-rating behaviors. In the studies of cross-domain recommendation, there is no universal criterion to quantify the similarity between two domains. Therefore, we refer to the usages in existing studies to determine which domains belong to similar ones. In the literatures [5, 7, 19], items of Books, Videos & DVDs, and Music are regarded as belonging to similar domains. The reason is these types of items are commonly to depict some stories and narratives with introduction, elucidation of the story, transition to another viewpoint and summing up. So they tend to convey sentiments from a common collection. Here, we choose all pairs of domains from Videos & DVDs, Books, and Music to be the source and the target domain. Totally, there are six source-target pairs. Then based on the data setting in Section 5.1, we construct the training and testing sets, and conduct fivefold cross-validation.

Experimental results are shown in Tables 4 and 5. We can find that the proposed PLSR-Latent clearly outperforms the competing methods in both MAE and RMSE. It can be also observed that our PLSR-Latent can consistently outperform the state-of-the-art CrossCBMF in all cases, and outperofrm another state-of-the-art CrossUNN in most cases. Such results prove that PLSR-Latent can tackle the cold-start problem of users' ratings in target domains given their ratings in similar

Table 4. Rating Prediction in MAE for Recommendation between Similar Domains

Source	Books	Music	Videos & DVDs	Music	Videos & DVDs	Books
Target	Videos & DVDs	Videos & DVDs	Books	Books	Music	Music
CrossMF	0.8709	0.8794	0.8880	0.8299	0.8676	0.8262
CrossNMF	0.8712	0.8794	0.8880	0.8292	0.8669	0.8252
CrossCBMF	0.9246	0.9309	0.9208	0.8593	0.9174	0.8731
CrossUNN	0.8912	0.9098	0.7399	0.7402	0.7739	0.7635
PLSR-Latent	0.8511	0.8537	0.8285	0.8145	0.7731	0.7582

Table 5. Rating Prediction in RMSE for Recommendation between Similar Domains

Source	Books	Music	Videos & DVDs	Music	Videos & DVDs	Books
Target	Videos & DVDs	Videos & DVDs	Books	Books	Music	Music
CrossMF	1.1413	1.1505	1.1482	1.0887	1.1347	1.1005
CrossNMF	1.1339	1.1508	1.1483	1.0882	1.1508	1.0991
CrossCBMF	1.2331	1.2453	1.2664	1.1907	1.2507	1.1931
CrossUNN	1.2101	1.2464	1.0833	1.1339	1.1033	1.1330
PLSR-Latent	1.0684	1.0720	1.0438	1.0331	0.9965	0.9845

domains as the sources. In addition, we find the performance of CrossMF and CrossNMF is quite close. The merit of NMF does not exhibit here. It is also worthwhile noticing the state-of-the-art CrossCBMF [19] generates the worse results (than CrossMF and CrossNMF) in terms of both MAE and RMSE. After carefully looking into the details of CrossCBMF, we find it is deliberately devised for top-N recommendation, i.e., ranking items with highest rating scores, in terms of high scores of *Recall*. However, it would be more practical to have accurate prediction of ratings in cross-domain recommendation, which can be achieved by our PLSR-Latent.

5.4.2 Heterogeneous Cross-domain Rating Prediction. We also conduct the cross-domain recommendation between domains with distinct characteristics, in addition to similar domains discussed in the previous paragraph. We refer to this experiment as *heterogeneous cross-domain rating prediction*. That said, given a source domain C, we aim at predicting the ratings of new users in a target domain D whose rating preferences and item categories are much different from source domain C. Here, we consider the domains {`Videos & DVDs`, `Books`, `Music`} and the domains {`Toys`, `Personal Care`, `Used Cars`} are two sets of domains with distinct characteristics. Using such six domains, there are 18 source-target pairs in total for the experiments.

The results are shown in Tables 6 and 7. It can be apparently found that the proposed PLSR-Latent outperforms the competitors in most source-target pairs and `Books-Personal Care`. We think the a little bit inconsistency between MAE and RMSE results from the square penalty of RMSE that makes the error values less than those of MAE. Even in such cases, our PLSR-Latent is still the best choice in predicting rating scores between distinct domains.

5.4.3 Compound Cross-domain Rating Prediction. In the real-world recommender systems, it may commonly happen that we have rating data from a variety of domains, and aim to recommend items for cold-start users in a certain target domain. We consider such cases as *compound* cross-domain recommendation: combining multiple domains as a compound domain to learn the latent factors and to transfer knowledge towards the target domain. Here, we aim to examine whether

Table 6. Rating Prediction in MAE for Recommendation between Distinct Domains

Source	Target	CrossMF	Cross-NMF	Cross-CBMF	Cross-UNN	PLSR-Latent
Videos & DVDs	Toys	0.8508	0.8519	0.8883	0.7686	0.8494
Books	Toys	0.7845	0.7842	0.8242	0.7688	0.8252
Music	Toys	0.8163	0.8163	0.8411	0.7937	0.8591
Videos & DVDs	Personal Care	1.0363	1.0366	1.0906	0.9790	0.9953
Books	Personal Care	0.9709	0.9711	1.0323	1.0459	0.9936
Music	Personal Care	1.0113	1.0108	1.0873	1.0414	1.0026
Videos & DVDs	Used Cars	1.0198	1.0185	1.1245	0.9312	0.8476
Books	Used Cars	1.0974	1.0980	1.0707	0.9653	0.8629
Music	Used Cars	1.0215	1.0212	0.9993	0.9360	0.8373
Toys	Videos & DVDs	0.9201	0.9188	0.9783	0.8601	0.8265
Personal Care	Videos & DVDs	0.9645	0.9641	1.0033	0.9862	0.8326
Used Cars	Videos & DVDs	1.0529	1.0542	1.0642	1.0288	0.8812
Toys	Books	0.8723	0.8724	0.8925	0.6506	0.7866
Personal Care	Books	0.9249	0.9247	0.9820	0.7588	0.7954
Used Cars	Books	1.2195	1.2213	1.0628	0.7105	0.8413
Toys	Music	0.9642	0.9655	1.0233	0.7475	0.7626
Personal Care	Music	0.9531	0.9511	1.0020	0.8049	0.7608
Used Cars	Music	1.0972	1.0985	1.0078	1.0447	0.7766

the proposed PLSR-Latent can again generate satisfying performance of rating prediction in the domain compound setting. We also wonder whether the compound domain can further boost the performance of cross-domain recommendation. By treating `Videos & DVDs`, `Books`, and `Music` as the target domains, respectively, and using the remaining nine domains (listed in Table 2 together as the compound domain, we have three experimental sets. The fivefold cross-validation with the training/testing setting as explained in Section 5.1 is applied again.

The results in terms of MAE and RMSE are exhibited in Tables 8 and 9, respectively. We can find that in the results of MAE, the superiority of PLSR-Latent is not displayed. PLSR-Latent is slightly worse than the other four competing methods. But in the results of RMSE, PLSR-Latent is able to stably outperform the competing methods. Again such inconsistency between MAE and RMSE could be due to the square penalty of RMSE that makes the error values less than those of MAE. In general, PLSR-Latent can better exploit compound source domains for cross-domain rating prediction.

5.4.4 Effects of Rating Density. In this section, we aim at investigating how the sparsity of the rating matrix affects the performance of rating prediction. Since in the real-world applications, sometimes the rating actions can be rarely performed by users in either source or target domain,

Table 7. Rating Prediction in RMSE for Recommendation between Distinct Domains

Source	Target	CrossMF	Cross-NMF	Cross-CBMF	Cross-UNN	PLSR-Latent
Videos & DVDs	Toys	1.1086	1.1095	1.2032	1.0886	1.0511
Books	Toys	1.0540	1.0537	1.1401	1.1124	1.0309
Music	Toys	1.0902	1.0900	1.1628	1.1660	1.0662
Videos & DVDs	Personal Care	1.3191	1.3191	1.4251	1.3920	1.2348
Books	Personal Care	1.2743	1.2747	1.3674	1.4465	1.2375
Music	Personal Care	1.3143	1.3144	1.4273	1.4906	1.2393
Videos & DVDs	Used Cars	1.3517	1.3502	1.4658	1.2916	1.0742
Books	Used Cars	1.4484	1.4483	1.4007	1.3484	1.0972
Music	Used Cars	1.3705	1.3704	1.3180	1.3462	1.0675
Toys	Videos & DVDs	1.2028	1.2018	1.3001	1.2023	1.0456
Personal Care	Videos & DVDs	1.2497	1.2493	1.3176	1.3554	1.0518
Used Cars	Videos & DVDs	1.3157	1.3171	1.3873	1.4318	1.0855
Toys	Books	1.1367	1.1366	1.2266	1.0237	1.0030
Personal Care	Books	1.2010	1.2009	1.3120	1.1426	1.0214
Used Cars	Books	1.4630	1.4648	1.3960	1.1500	1.0476
Toys	Music	1.2669	1.2679	1.3888	1.1507	0.9917
Personal Care	Music	1.2341	1.2315	1.3287	1.2491	0.9800
Used Cars	Music	1.3398	1.3407	1.3257	1.5488	0.9971

Table 8. Rating Prediction in MAE Using Compound Source Domains

Source	Other Nine Domains	Other Nine Domains	Other Nine Domains
Target	Videos & DVDs	Books	Music
CrossMF	0.8601	0.8181	0.8132
CrossNMF	0.8595	0.8181	0.8130
CrossCBMF	0.8601	0.8181	0.8132
CrossUNN	0.8820	0.7319	0.7589
PLS-Latent	0.8638	0.8392	0.7727

it is practical to examine whether the proposed PLSR-Latent can still have better performance and to what extent PLSR-Latent can deal with the rating sparsity. We conduct such experiments by varying the density of source-domain and target-domain user-rating matrices. Here, we consider two domains `Videos & DVDs` and `Books` for the experiments, i.e., `Videos & DVDs` as the source and `Books` as the target, and `Books` as the source and `Videos & DVDs` as the target. We vary the density of a rating matrix by randomly removing its ratings while ensuring every user has at least

Table 9. Rating Prediction in RMSE Using Compound Source Domains

Source	Other Nine Domains	Other Nine Domains	Other Nine Domains
Target	Videos & DVDs	Books	Music
CrossMF	1.1125	1.0679	1.0690
CrossNMF	1.1120	1.0681	1.0692
CrossCBMF	1.1125	1.0679	1.0690
CrossUNN	1.1862	1.0735	1.0859
PLS-Latent	1.0772	1.0508	0.9933

Table 10. Rating Prediction in MAE by Adjusting the Density of the User-Rating Matrix (Source: `Videos & DVDs`, Target: `Books`)

		Adjusted Density of Source			Adjusted Density of Target		
	original	0.01%	0.009%	0.008%	0.01%	0.009%	0.008%
CrossMF	0.8880	1.0004	1.0206	1.0520	0.9119	0.9170	0.9338
CrossNMF	0.8881	1.0007	1.0200	1.0523	0.9117	0.9176	0.9332
CrossCBMF	0.9208	1.0036	1.0276	1.0553	0.9292	0.9475	0.9777
PLS-Latent	0.8285	0.8214	0.8252	0.8220	0.8656	0.8922	0.9340

Table 11. Rating Prediction in MAE by Adjusting the Density of the User-Rating Matrix (Source: `Books`, Target: `Videos & DVDs`)

		Adjusted Density of Source			Adjusted Density of Target		
	original	0.01%	0.009%	0.008%	0.01%	0.009%	0.008%
CrossMF	0.8709	0.9028	0.9249	0.9742	0.9465	0.9680	0.9993
CrossNMF	0.8712	0.9031	0.9257	0.9744	0.9464	0.9678	0.9995
CrossCBMF	0.9246	0.9368	0.9477	0.9535	0.9913	1.0020	1.0158
PLS-Latent	0.8511	0.8520	0.8527	0.8546	0.9490	0.9703	1.0066

one rating and every item is rated by at least one user, i.e., each row and each column contains at least one non-zero value. Then the density of the matrix will be lowered down. Since the density values of original rating matrices are quite low, as shown in Table 2, we need to make sure that the adjusted density values must be higher than the *ideal lower bound* of density $Density_{ilb}$ of the matrix:

$$Density_{ilb} = \frac{\text{Minimal number of ratings}}{\text{Size of matrix}} = \frac{max(nRow, nCol)}{nRow \times nCol}, \tag{17}$$

where $nRow$ is the number of users and $nCol$ is the number of items. For example, the original density of `Videos & DVDs` user-rating matrix is 0.024%, and the corresponding $Density_{ilb}$ is $\frac{28,972}{25,288 \times 28,972} = 0.004\%$. The original density of `Books` user-rating matrix is 0.0118%, and the corresponding $Density_{ilb}$ is $\frac{59,488}{15,504 \times 59,488} = 0.0065\%$. Here, we choose to adjust the density values to be 0.01%, 0.009, and 0.008% for the experiments.

The results are shown in Tables 10 and 11. We can accordingly have the following findings. First, in general PLST-Latent outperforms the competing methods as the density decreases toward 0.008% in most cases (except for adjusting the density of target-domain `Videos & DVDs` matrix

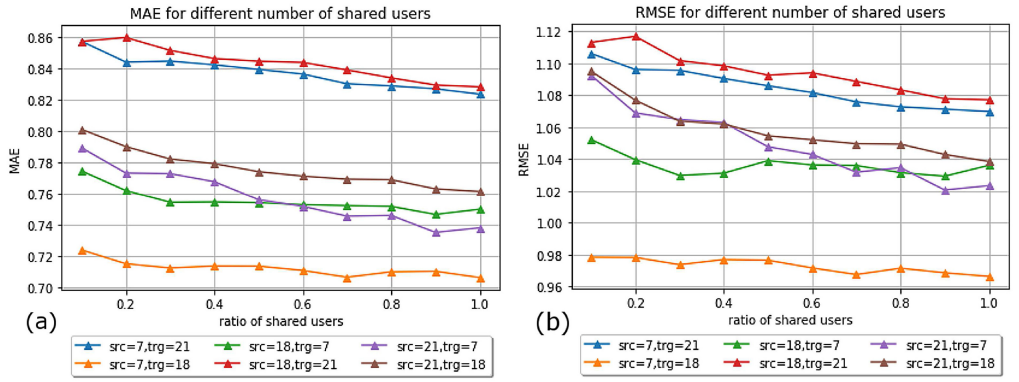

Fig. 4. MAE and RMSE values by varying the number of shared users between source and target domains. Note that the indices in source (src) and target (trg) can be referred to in Table 2.

in Table 11). Second, the most important finding is that as the density values of source-domain matrices decrease (the left parts in both tables), the MAE values of our PLSR-Latent nearly remain the same. In other words, the sparsity of user-rating matrix in the source domain hardly influence the performance. Such outcome differs from existing cross-domain rating prediction. We think it is because of the property of PLSR—when learning the regression coefficients, the response is more influential than the predictor. By applying PLSR to cross-domain recommendation, such property of PLSR implies that target-domain information leads to higher impact in terms of prediction performance than source-domain information. In other words, PLSR provides the benefit allowing sparse source matrices for cross-domain recommendation. Third, as the density values of target-domain matrices decrease (the right parts in both tables), the MAE values tend to go sightly higher for all of the methods, compared to the original matrices. Such outcome is reasonable due to the aforementioned property of PLSR.

5.4.5 Performance vs. Number of Shared Users. In this experiment, we would like to understand how the overlapping of users between source and targets domains affect the performance of cross-domain rating prediction using the proposed PLSR-Latent. Here is the experimental setting. Let the number of shared users between source and target domains be s. In the training of PLSR-Latent, we vary the number of shared users ρs, where $\rho = 0.1, 0.2, 0.3, \ldots, 0.9$ is the ratio of shared users. Under different ratio values of shared users, we report the MAE and RMSE values for six combinations of source and target domains. The results are presented in Figure 4. It can be apparently found that lower percentages of shared users between domains lead to worse performance. Such kind of tendency is exhibited across various source-target combinations. Nevertheless, the differences of MAE and RMSE for $\rho = 0.1$ and $\rho = 0.9$ are small, i.e., around 0.02 to 0.05. Such small differences also demonstrate the prediction power of the proposed PLSR-Latent.

5.4.6 Time Efficiency vs. Dimensionality. We aim to study how different dimensionality values k affect the time efficiency (in second) at the training stage. The results reported in this experiment also represent the differences of running time between PLSR-Latent and PLSR-CrossRec. We take advantage of the *PLSRegression* technique in scikit-learn[3] to execute this experiment. In addition, the experiments are conducted in the machine equipped with DELL PowerEdge R620 and 64 GB RAM. We vary the dimensionality k from 2 to 400 and 28,972. Setting $k = 28{,}972$ means using the

[3]http://scikit-learn.org/.

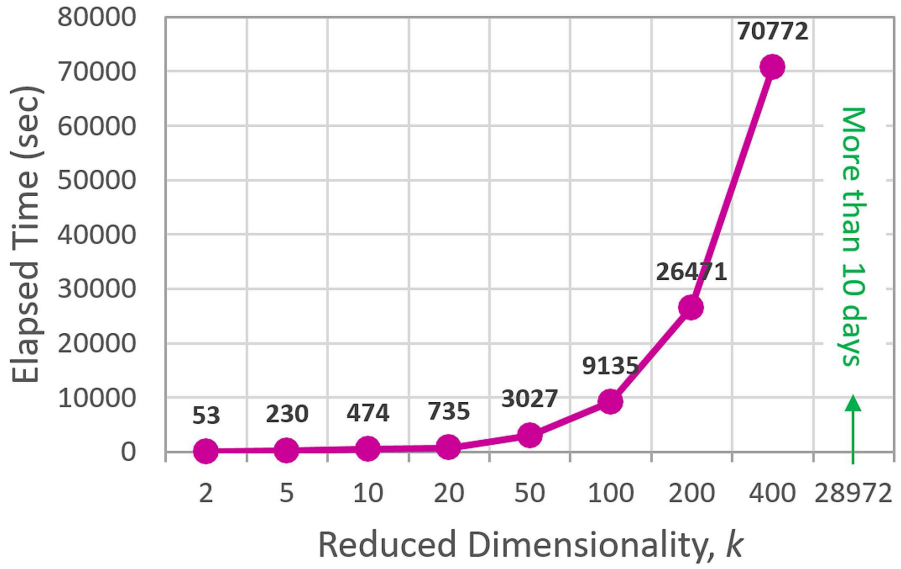

Fig. 5. Elapsed time (in seconds) by varying the dimensionality k for the proposed PLSR-based methods.

original matrix without dimension reduction for PLSR computation. In other words, PLSR with $k = 28{,}972$ can be treated as the PLSR-CrossRec method. The results are presented in Figure 5. It can be apparently found that the elapsed time rapidly grows when k increases. The elapsed time of $k = 28{,}972$ (i.e., PLSR-CrossRec) is definitely unacceptable (it takes more than 10 days) and cannot use for real applications. By reducing the dimensionality to $k = 5$, it takes around 6 minutes to complete the PLSR computation. Also under the dimensionality $k = 5$, we can derive satisfying performance exhibited in all of the previous experiments. This is also why we do not report the results for PLSR-CrossRec in all previous experiments. In short, PLSR-Latent with lower dimensionality for latent factors can generate promising rating prediction using acceptable running time.

5.4.7 Performance vs. Dimensionality. Here, we also study how the dimensionality values of our PLSR-Latent affect the performance. There are two important dimensionality values. One is the number K_{mf} of preserved dimensions in matrix factorization while the other is the number K_{pls} of preserved dimensions in the partial least squares. Since we empirically find that the value of K_{pls} does not affect the performance (i.e., different K_{pls} values lead to quite similar results), we tend to set $k = K_{mf} = K_{pls}$ and vary k in this experiment. By setting $k = 2, 5, 10, 20, 40, 60, \ldots, 200$, the results in terms of MAE and RMSE are shown in Figure 6. We can find that for different combinations of source and target domains, the best performance tends to locate at between $k = 40$ and $k = 60$. The dimensionality values k lower than 40 and higher than 60 generate worse performance. Therefore, we are suggested to set $k = 50$ in general.

5.4.8 Time Efficiency. Last, we aim at comparing the time efficiency of the proposed PLSR-Latent with the competing methods, CrossMF, CrossNMF, and CrossCBMF. The results of running time (seconds) are exhibited in Figure 7. We can find that CrossMF is the most efficient. However, as reported in previous sections, CrossMF leads to the worse performance. The time efficiency of our PLSR-Latent is close to that of CrossNMF. The state-of-the-art CrossCBMF method is the most inefficient, around two times longer than our PLSR-Latent. As deploying cross-domain rating

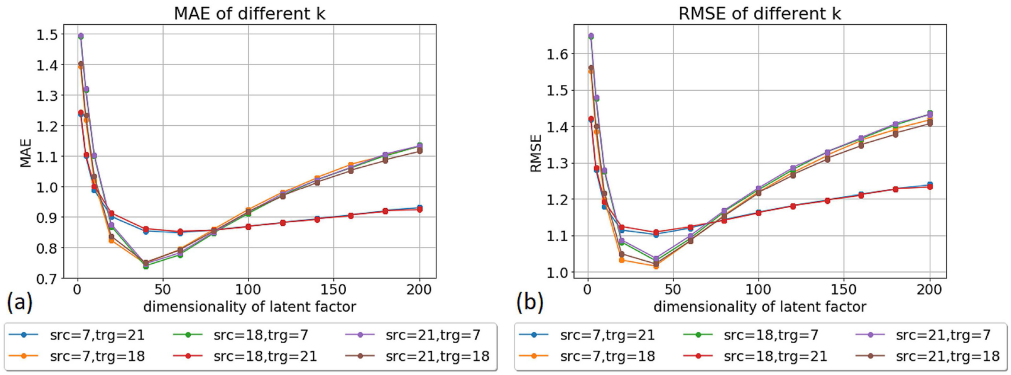

Fig. 6. MAE and RMSE values by varying the dimensionality k for the proposed PLSR-Latent method. Note that the indices in source (src) and target (trg) can be referred to in Table 2.

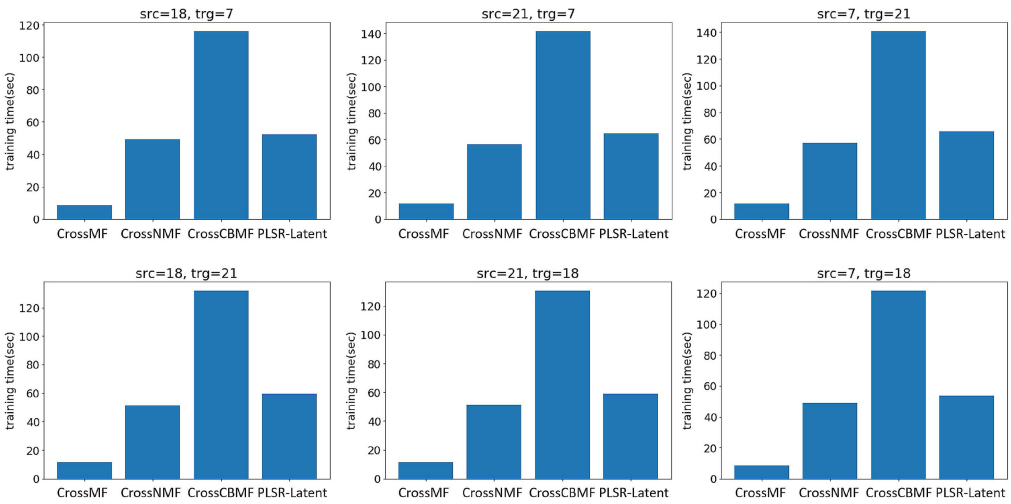

Fig. 7. Running time (in seconds) of PLSR-Latent, compared with CrossMF, CrossNMF, and CrossCBMF. Note that the indices in source (src) and target (trg) can be referred to in Table 2.

prediction for practical applications, in which our PLSR-Latent can generate lowest values of MAE and RMSE, we believe PLSR-Latent can better strike a balance between effectiveness and efficiency.

6 CONCLUSIONS AND DISCUSSION

This article develops a novel cross-domain recommendation mechanism for cold-start users by modifying and extending the technique of partial least squares regression (PLSR). The key is leveraging users performing ratings in both source and target domains to learn the matrix of regression coefficients, which is capable of transferring knowledge from source to target domain. To improve the time efficiency of PLSR-based cross-domain recommendation, we further devise the PLSR-Latent model that employs the latent factors of rating matrices, whose dimensionality is much fewer than the original matrices, to learn the regression coefficients, rather than the original ones. Experiments conducted on well-known Epinions.com datasets across 10 domains exhibit several superiority of the proposed PLSR-Latent method. (1) PLSR-Latent can outperform

typical cross-domain rating prediction methods. (2) The superiority of PLSR-Latent is displayed under a variety of cross-domain settings, including homogeneous, heterogeneous, and compound. (3) PLSR-Latent is validated to stably and promisingly deal with sparse source user-rating matrices. (4) The time efficiency of PLST-Latent at the training stage is also promising.

Here are three points we would like to discuss. First, the first stage of our method is using the traditional matrix factorization (MF) to fill in missing values of raw user-rating matrices. In fact, our method allows different implementations of missing data imputation methods, which will not affect our PLSR-based algorithms for cross-domain recommendation. We agree with that there are other advanced methods that can be used to fill in the missing values. The technique of *Non-negative Matrix Factorization* (NMF) [14] you suggested is a good alternative, since its performance tends to be better than the traditional MF. Applying NMF for missing data imputation will not change our PLSR-based methods. We think an advanced missing data imputation method can lead to better performance for cross-domain rating prediction.

Second, in the proposed PLSR-Latent method, we need to perform MF again at the online stage. One may question that the two MF results can be different, and in this case, can we use the regression coefficient matrix learned from the training phase for the new latent matrix? In fact, two user-rating matrices are slightly different from only new entries of new users. In other words, given using MF to fill in missing values of the new users vectors, with the same setting of the dimensionality of latent factors, most of MF results will not be affected (except for the few new entries). In addition, since the regression coefficient matrix is learned from the source-domain training data, and the target-domain testing cold-start users is assumed to share similar rating knowledge with source-domain users. Therefore, the regression coefficient matrix is capable of transferring the rating knowledge from the source domain to the target domain for the new latent matrix.

Third, in our current work, we perform MF at the online stage to obtain the new user's latent factor $\mathbf{p}_X{}^T$, rather than derive the latent factor of new users from the old latent matrix. In fact, we can derive the new latent factor of the new user via the technique of *incremental matrix factorization* [31]. The incremental matrix factorization can allow us to derive the latent factor of new user without performing MF again at the online stage. In the practical application, the incremental matrix factorization will be suggested.

This study for cross-domain recommendation is based on regression analysis. There are a number of extensive future work. First, PLSR belongs to prediction by linear regression. For those user-rating data possessing the non-linearity property, the proposed PLSR-Latent may not lead to satisfying performance. We plan to apply a kernel PLSR [26] to allow prediction with non-linear regression. Second, an limitation of PLSR-based methods is needing to first complete the rating matrices. That said, it is necessary to contain no missing values in the input matrices of PLSR. Artificially filling in missing fields may distort the data property. In the future, we will integrate PLSR with the *Orthogonal Matching Pursuit* technique [6] to afford incomplete and sparse rating matrices.

REFERENCES

[1] G. Adomavicius and A. Tuzhilin. 2015. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Trans. Knowl. Data Eng.* 17, 6 (2015), 736–749.

[2] H. Aguinis and R. K. Gottfredson. 2010. Best-practice recommendations for estimating interaction effects using moderated multiple regression. *J. Org. Behav.* 31, 6 (2010), 776–786.

[3] J. Bobadilla, F. Ortega, A. Hernando, and A. Gutierrez. 2013. Recommender systems survey. *Knowl.-Based Syst.* 46 (2013), 109–132.

[4] J. S. Breese, D. Heckerman, and C. Kadie. 1998. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence (UAI'98).*

[5] I. Cantador, I. Fernandez-Tobías, S. Berkovsky, and P. Cremonesi. 2015. Cross-domain recommender systems. *Recommend. Syst. Handbook* (2015), 919–959.
[6] Y. Chen and C. Caramanis. 2013. Noisy and missing data regression: Distribution-oblivious support recovery. In *Proceedings of the 30th International Conference on Machine Learning (ICML'13)*. 383–391.
[7] P. Cremonesi, A. Tripodi, and R. Turrin. 2011. Cross-domain recommender systems. In *Proceedings of the IEEE 11th International Conference on Data Mining Workshops*.
[8] M. A. Efroymson. 1960. Multiple regression analysis. *Math. Methods for Dig. Comput.* 23 (1960), 191–203.
[9] A. Farseev, I. Samborskii, A. Filchenkov, and T.-S. Chua. 2017. Cross-domain recommendation via clustering on multi-layer graphs. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'17)*. 195–204.
[10] I. Fernandez-Tobias and I. Cantador. 2015. On the use of cross-domain user preferences and personality traits in collaborative filtering. In *Proceedings of the International Conference on User Modeling, Adaptation, and Personalization (UMAP'15)*. 343–349.
[11] X. Jia, A. Wang, X. Li, G. Xun, W. Xu, and A. Zhang. 2015. Multi-modal learning for video recommendation based on mobile application usage. In *Proceedings of IEEE International Conference on Big Data (BigData'15)*. 837–842.
[12] I. Jolliffe. 2002. Principal component analysis. *Springer Series in Statistics* 2.
[13] Y. Koren, R. Bell, and C. Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 42, 8 (2009), 30–37.
[14] D. D. Lee and H. S. Seung. 2001. Algorithms for non-negative matrix factorization. In *Proceedings of the Conference on Advances in Neural Information Processing Systems (NIPS'01)*.
[15] B. Li, Q. Yang, and X. Xue. 2009. Can movies and books collaborate?: Cross-domain collaborative filtering for sparsity reduction. In *Proceedings of the 21st International Joint Conference on Artifical Intelligence (IJCAI'09)*.
[16] B. Lika, K. Kolomvatsos, and S. Hadjiefthymiades. 2014. Facing the cold start problem in recommender systems. *Expert Syst. Appl.* 41, 4 (2014), 2065–2073.
[17] R. Manne. 1987. Analysis of two partial-least-squares algorithms for multivariate calibration. *Chemometr. Intell. Lab. Syst.* 2, 1 (1987), 187–197.
[18] S. Meyffret, E. Guillot, L. Medini, and F. Laforest. 2002. RED: A rich epinions dataset for recommender systems. Research Report, *Universite de Lyon* (2002).
[19] N. Mirbakhsh and C. X. Ling. 2015. Improving top-N recommendation for cold-start users via cross-domain information. *ACM Trans. Knowl. Discov. Data* 9, 4 (2015), 33.
[20] R. J. Mooney and L. Roy. 2000. Content-based book recommending using learning for text categorization. In *Proceedings of the 5th ACM conference on Digital Libraries (JCDL'00)*.
[21] L. Nie, L. Zhang, Y. Yang, M. Wang, R. Hong, and T.-S. Chua. 2015. Beyond doctors: Future health prediction from multimedia and multimodal observations. In *Proceedings of the 23rd ACM International Conference on Multimedia (MM'15)*. 591–600.
[22] W. Pan, E. W. Xiang, N. N. Liu, and Q. Yang. 2010. Transfer learning in collaborative filtering for sparsity reduction. In *Proceedings of the 24th AAAI Conference on Artificial Intelligence (AAAI'10)*.
[23] W. Pan, E. W. Xiang, and Q. Yang. 2012. Transfer learning in collaborative filtering with uncertain ratings. In *Proceedings of the 26th AAAI Conference on Artificial Intelligence (AAAI'12)*.
[24] S. Qian, T. Zhang, R. Hong, and C. Xu. 2015. Cross-domain collaborative learning in social multimedia. In *Proceedings of the 23rd ACM International Conference on Multimedia (MM'15)*. 99–108.
[25] R. Rosipal and N. Kramer. 2006. Overview and recent advances in partial least squares. In *Proceedings of the International Conference on Subspace, Latent Structure and Feature Selection*.
[26] R. Rosipal and L. J. Trejo. 2002. Kernel partial least squares regression in reproducing kernel hilbert space. *J. Mach. Learn. Res.* 2 (2002), 97–123.
[27] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th International Conference on World Wide Web (WWW'01)*.
[28] B. Shapira, L. Rokach, and S. Freilikhman. 2013. Facebook single and cross domain data for recommendation systems. *User Model. User-Adapt. Interact.* 23, 2 (2013), 211–247.
[29] Y. Shi, M. Larson, and A. Hanjalic. 2011. Tags as bridges between domains: Improving recommendation with tag-induced cross-domain collaborative filtering. In *Proceedings of the International Conference on User Modeling, Adaptation, and Personalization*.
[30] A. P. Singh and G. J. Gordon. 2008. Relational learning via collective matrix factorization. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'08)*. 650–658.
[31] Q. Song, J. Cheng, and H. Lu. 2015. Incremental matrix factorization via feature space re-learning for recommender system. In *Proceedings of the 9th ACM Conference on Recommender Systems (RecSys'15)*. 277–280.

[32] N. Srivastava and R. Salakhutdinov. 2014. Multimodal learning with deep Boltzmann machines. *J. Mach. Learn. Res.* 15 (2014), 2949–2980.
[33] H. Steck. 2013. Evaluation of recommendations: Rating-prediction and ranking. In *Proceedings of the 7th ACM Conference on Recommender Systems (RecSys'13)*. 213–220.
[34] A. Tiroshi, S. Berkovsky, M. A. Kaafar, T. Chen, and T. Kuflik. 2013. Cross social networks interests predictions based on graph features. In *Proceedings of the 7th ACM Conference on Recommender Systems*.
[35] H. Wold. 1975. Path models with latent variables: The NIPALS approach. *Quant. Sociol.: Int. Perspect. Math. Stat. Model Build.* (1975), 307–357.
[36] Y. Zhang, B. Cao, and D.-Y. Yeung. 2012. Multi-domain collaborative filtering. In *Proceedings of the 26th Conference on Uncertainty in Artificial Intelligence (UAI'12)*.
[37] X. W. Zhao, Y. Guo, Y. He, H. Jiang, Y. Wu, and X. Li. 2014. We know what you want to buy: A demographic-based system for product recommendation on microblogs. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'14)*. 1935–1944.
[38] F. Zhuang, P. Luo, H. Xiong, Y. Xiong, Q. He, and Z. Shi. 2010. Cross-domain learning from multiple sources: A consensus regularization perspective. *IEEE Trans. Knowl. Data Eng.* 22, 12 (2010), 1664–1678.

Received November 2017; revised April 2018; accepted June 2018