Transfer Learning in Collaborative Filtering for Sparsity Reduction

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

Abstract

Data sparsity is a major problem for collaborative filtering (CF) techniques in recommender systems, especially for new users and items. We observe that, while our target data are sparse for CF systems, related and relatively dense auxiliary data may already exist in some other more mature application domains. In this paper, we address the data sparsity problem in a target domain by transferring knowledge about both users and items from auxiliary data sources. We observe that in different domains the user feedbacks are often heterogeneous such as ratings vs. clicks. Our solution is to integrate both user and item knowledge in auxiliary data sources through a principled matrix-based transfer learning framework that takes into account the data heterogeneity. In particular, we discover the principle coordinates of both users and items in the auxiliary data matrices, and transfer them to the target domain in order to reduce the effect of data sparsity. We describe our method, which is known as *coordinate system transfer* or CST, and demonstrate its effectiveness in alleviating the data sparsity problem in collaborative filtering. We show that our proposed method can significantly outperform several state-of-the-art solutions for this problem.

Introduction

Collaborative Filtering (CF) (Resnick et al. 1994) was proposed to predict the missing values in an incomplete matrix, i.e. user-item rating matrix. A major difficulty in CF is the data sparsity problem, because most users can only access a limited number of items. This is especially true for newly created online services, where overfitting can easily happen when we learn a model, causing significant performance degradation.

For a typical recommendation service provider such as movie rental services, there may not be sufficient user-item rating records of a new customer or a new product. Mathematically, we call such data *sparse*, where the useful information is scattered and few. Using these data matrices for recommendation may result in low-quality results due to overfitting. To address this problem, some service providers turn to explicitly ask the newly registered customers to rate some selected items, such as some most sparsely rated jokes in a joke recommender system (Nathanson, Bitton, and Goldberg 2007). However, methods like this may degrade the customer's experience and satisfaction with the system, or even cause the customer churn if the customer is pushed too much. Other methods, such as those that make use of the implicit user feedbacks (Hu, Koren, and Volinsky 2008), may produce good results, but still rely on tracking the users' behavior on the products to be predicted continuously.

More recently, researchers have introduced transfer learning methods for solving the data sparsity problem (Singh and Gordon 2008), (Li, Yang, and Xue 2009a), (Li, Yang, and Xue 2009b). These methods are aimed at making use of the data from other recommender systems, referred to as the auxiliary domain, and transfer the knowledge that are consistent in different domains to the target domain. (Phuong and Phuong 2008) proposed to apply multi-task learning to collaborative filtering, but their studied problem was different, as it does not consider any auxiliary information sources. Instead, it formulates a multiple binary classification problem in the same CF matrix, one for each user. For the transfer learning methods, (Singh and Gordon 2008) uses common latent features when factorizing multiple matrices and (Li, Yang, and Xue 2009a; 2009b) consider the cluster-level rating patterns as potential candidates to be transferred from the auxiliary domain. However, there are two limitations of these methods due to certain assumptions that are often not met in practice. Firstly, they require the user preferences expressed in the auxiliary and target domains to be homogeneous such as a common rating scale of 1-5. In practice, the data from the auxiliary domain may not be ratings at all but implicit feedbacks, such as user click records, represented as 0/1. Secondly, methods like (Li, Yang, and Xue 2009a; 2009b) assume that both users and items in an auxiliary data source are related to the target data, while in practice it is often much easier to find an auxiliary data source with either similar users or similar items but not both. For example, there exist complementary systems such as LaunchCast and Youtube serving different content (music vs. video) to the same users or competitive services such as Amazon and Barnes&Nobel selling similar products to different users.

In this paper, we propose a principled matrix factorization based framework named as *coordinate system transfer* (CST) for transferring both user and item knowledge from an auxiliary domain. In designing the CST algorithm, we

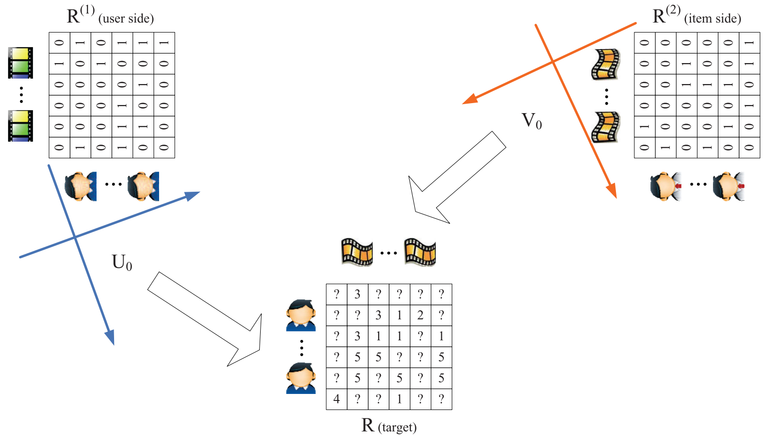


Figure 1: Illustration of the Two-Sided Transfer Learning Method (Coordinate System Transfer) in a 2-D plane.

have to overcome the aforementioned limitations of existing methods. First, we have to decide how to integrate the heterogeneous forms of user feedbacks, which are explicit ratings in the target domain and implicit feedbacks in the auxiliary domain. Second, we must incorporate both the user and item knowledge from the auxiliary domain in a flexible way. We observe that these two challenges are related to each other, and are similar to the two fundamental problems in transfer learning; that is, in deciding what to transfer and how to transfer in transfer learning (Pan and Yang 2009).

Our main idea is to discover the common latent information which is shared in both auxiliary and target domains. Although the user feedbacks in auxiliary and target domains may be heterogeneous, we find that for many users, their latent tastes which represent their intrinsic preference structure in some subspace are similar. For example, for movie renters, their preferences on drama, comedy, action, crime, adventure, documentary and romance expressed in their explicit rating records in the target domain, are similar to that in their implicit click records on other movies in an auxiliary domain. We assume that there is a finite set of tastes, referred to as principle coordinates, which characterize the domain independent preference structure of users and thus can be used to define a common coordinate system for representing users. On the other hand, we can have another coordinate system for representing items' main factors, i.e. director, actors, prices, 3ds Max techniques, etc. In the proposed solution CST, we first use sparse matrix tri-factorization on the auxiliary data to discover the principle coordinates for constructing the coordinate systems for users and items, which answers the question of what knowledge to transfer. We then use a novel regularization technique in order to adapt the coordinate systems for modeling target domain data, which addresses the problem of how to transfer the knowledge.

Our Solution: Coordinate System Transfer

Problem Formulation

We use boldface uppercase letters, such as $\mathbf{Y}$, to denote matrices, and $Y_{u:}$, $Y_{:i}$, y_{ui} to denote the uth row, ith column and the entry located at (u, i) of $\mathbf{Y}$, respectively. $\mathbf{I}$ denotes the identity matrix of appropriate dimension.

In our problem setting, we have a target domain where we wish to solve our CF problem. In addition, we also have an auxiliary domain which is similar to the target domain. The auxiliary domain can be partitioned into two parts: a user part and an item part, which share common users and items, respectively, with the target domain. We call them the user side and item side, respectively.

We use n as the number of users and m the number of items in the target domain, and we use $\mathbf{R} \in \mathbb{S}^{n\times m}$ as the observed sparse rating matrix, where $\mathbb{S}$ is the set of observed user feedbacks, i.e. $\mathbb{S} = \{1, 2, 3, 4, 5\}$. Here, $r_{ui} \in \mathbb{S}$ is the rating given by user u on item i. $\mathbf{Y} \in \{0, 1\}^{n\times m}$ is the corresponding indicator matrix, with $y_{ui} = 1$ if user u has rated item i, and $y_{ui} = 0$ otherwise.

For the auxiliary domain, we use $\mathbf{R}^{(1)}$, $\mathbf{R}^{(2)}$ to denote data matrices from auxiliary data sources that share common users and items with $\mathbf{R}$, respectively. The sets of observed user feedbacks, $\mathbb{S}^{(1)}$, $\mathbb{S}^{(2)}$, of the two data matrices are often not the same as $\mathbb{S}$ of the target domain, i.e. $\mathbb{S}^{(1)} = \mathbb{S}^{(2)} = \{0, 1\}$. Our goal is to make use of $\mathbf{R}^{(1)}$, $\mathbf{R}^{(2)}$ to help predict the missing values in $\mathbf{R}$, which is illustrated in Figure 1.

Coordinate System Transfer: Our Two-sided Transfer Learning Solution

In our solution, known as *coordinate system transfer* or CST, we first discover an auxiliary domain subspace where we can find some principle coordinates. These principle coordinates can be used to bridge two domains, and ensure knowledge transfer. Our algorithm is shown in Algorithm 1, which is described in two major steps, as described below.

Step 1: Coordinate System Construction

In step 1, we first find the principle coordinates of the auxiliary domain data. The principle coordinates in a CF system can be obtained via Singular Value Decomposition (SVD) on a full-rating matrix, if we ask every newly joined user to rate those system selected items. Typically, each principle coordinate represents a semantic concept related to the user's taste or item's factor. However, both our auxiliary and target domain data are represented by sparse matrices instead of full matrices, as many rating values are missing. Hence, we use the sparse SVD (Buono and Politi 2004) on auxiliary data $\mathbf{R}^{(i)}, i = 1, 2$,

$$\min_{\mathbf{U}^{(i)},\mathbf{V}^{(i)},\mathbf{B}^{(i)}} ||\mathbf{Y}^{(i)} \odot (\mathbf{R}^{(i)} - \mathbf{U}^{(i)}\mathbf{B}^{(i)}\mathbf{V}^{(i)T})||_F^2 \qquad (1)$$

where $\mathbf{B}^{(i)} = \text{diag}(\sigma_1^{(i)}, \ldots, \sigma_j^{(i)}, \ldots, \sigma_d^{(i)})$ is a diagonal matrix, $\sigma_1^{(i)} \geq \sigma_2^{(i)} \geq \ldots \geq \sigma_d^{(i)} \geq 0$ are eigenvalues, and $\mathbf{U}^{(i)T}\mathbf{U}^{(i)} = \mathbf{I}$, $\mathbf{V}^{(i)T}\mathbf{V}^{(i)} = \mathbf{I}$ ensures that their columns are othornormal. Note that each column of $\mathbf{U}^{(i)}$, $\mathbf{V}^{(i)}$ represents a semantic concept; i.e. user taste (Nathanson, Bitton, and Goldberg 2007) in collaborative filtering or document theme (Deerwester et al. 1990) in information retrieval. Those columns are the principle coordinates in the low-dimensional space, and for this reason, we call our approach the *coordinate system transfer*.

Definition (Coordinate System) A *coordinate system is a* ***matrix*** *with columns of orthonormal bases (principle coordinates), where the columns are located in descending order according to their corresponding eigenvalues.*

Figure 1 shows two coordinate systems in the auxiliary domain, one for users and the other for items. We represent these two coordinate systems using two matrices as,

$$\mathbf{U}_0 = \mathbf{U}^{(1)}, \mathbf{V}_0 = \mathbf{V}^{(2)} . \qquad (2)$$

where the matrices $\mathbf{U}^{(1)}$, $\mathbf{V}^{(2)}$ consist of top d principle coordinates from (1).

Step 2: Coordinate System Adaptation

In Step 2 of the CST algorithm (Algorithm 1), we adapt the principle coordinates discovered in the previous step to the target domain.

After obtaining the coordinate systems from the auxiliary data $\mathbf{R}^{(1)}, \mathbf{R}^{(2)}$, the latent user tastes and item factors are captured by the coordinate systems and can be transferred to the target domain $\mathbf{R}$. In the target domain, we denote the two coordinate systems as $\mathbf{U}, \mathbf{V}$ for users and items, respectively, which are also required to be orthonomal according to the definition of coordinate system, that is $\mathbf{U}^T\mathbf{U} = \mathbf{I}$, $\mathbf{V}^T\mathbf{V} = \mathbf{I}$. Instead of requiring the two coordinate systems from the auxiliary domain and target domain to be exactly the same, i.e. $\mathbf{U} = \mathbf{U}_0$, $\mathbf{V} = \mathbf{V}_0$, we relax this requirement and only require them to be similar. We believe that though two domains are related, the latent user tastes and item factors in two domains can still be a bit different due to the domain specific contexture, i.e. advertisements or promotions on the service provider's website. Hence, we replace the constraint $\mathbf{U} = \mathbf{U}_0$, $\mathbf{V} = \mathbf{V}_0$ with two additional regularization terms $||\mathbf{U} - \mathbf{U}_0||_F^2$, $||\mathbf{V} - \mathbf{V}_0||_F^2$.

Further, in order to allow more freedom of rotation and scaling, we adopt the tri-factorization method (Ding et al. 2006), (Long et al. 2007), and allow the rating matrix to be factorized into *three* parts, one for the user side coordinate system $\mathbf{U}$, a second part for the item side coordinate system $\mathbf{V}$, and the third part $\mathbf{B}$ to allow rotation and scaling between the two coordinate systems. Note that the problems in (Ding et al. 2006), (Long et al. 2007) are quite different from ours, as they require that $\mathbf{U}$, $\mathbf{V}$ are non-negative, $\mathbf{R}$ is full, and (Long et al. 2007) even requires $\mathbf{U} = \mathbf{V}$ in clustering.

Algorithm 1 CST: Coordinate System Transfer.

```
Input: The training data $\mathbf{R}$, the auxiliary data $\mathbf{R}^{(1)}$, $\mathbf{R}^{(2)}$
Output: $\mathbf{U}$, $\mathbf{V}$, $\mathbf{B}$.
  Step 1. Apply sparse SVD on auxiliary data $\mathbf{R}^{(1)}, \mathbf{R}^{(2)}$, and obtain two principle coordinate systems $\mathbf{U}_0 = \mathbf{U}^{(1)}$, $\mathbf{V}_0 = \mathbf{V}^{(2)}$. Initialize the target coordinate systems with $\mathbf{U} = \mathbf{U}_0$, $\mathbf{V} = \mathbf{V}_0$.
  repeat
    Step 2.1. Fix $\mathbf{U}$, $\mathbf{V}$, and estimate $\mathbf{B}$ from $||\mathbf{Y} \odot (\mathbf{R} - \mathbf{U}\mathbf{B}\mathbf{V}^T)|| = 0$ (Keshavan, Montanari, and Oh 2010).
    Step 2.2. Fix $\mathbf{B}$, and update $\mathbf{U}$, $\mathbf{V}$ via alternative gradient descent method on Grassmann manifold (Edelman, Arias, and Smith 1999), (Buono and Politi 2004).
  until Convergence
```

We obtain the following optimization problem for CST formulation,

$$\begin{aligned} \min_{\mathbf{U},\mathbf{V},\mathbf{B}} \ & ||\mathbf{Y} \odot (\mathbf{R} - \mathbf{U}\mathbf{B}\mathbf{V}^T)|| \\ & + \frac{\rho_u}{2}||\mathbf{U} - \mathbf{U}_0||_F^2 + \frac{\rho_v}{2}||\mathbf{V} - \mathbf{V}_0||_F^2 \quad (3) \\ \text{s.t.} \quad & \mathbf{U}^T\mathbf{U} = \mathbf{I}, \mathbf{V}^T\mathbf{V} = \mathbf{I} \end{aligned}$$

where $\mathbf{B}$ is different from $\mathbf{B}^{(i)}$ in (1), as it is not required to be diagonal, but can be full, and the effect of $\mathbf{B}$ is not only scaling as that of $\mathbf{B}^{(i)}$, *but also rotation when fusing two coordinate systems via* $\mathbf{U}\mathbf{B}\mathbf{V}^T$*, and hence more flexible than two-part factorization*. After we learn $\mathbf{U}, \mathbf{V}, \mathbf{B}$, each missing entry r_{ui} in $\mathbf{R}$ can be predicted via $U_{u:}\mathbf{B}V_{i:}^T$, where $U_{u:}$, $V_{i:}$ are the user u's latent tastes and item i's latent factors, respectively. The tradeoff parameters ρ_u and ρ_v represent the confidence on the auxiliary data. When $\rho_u, \rho_v \rightarrow \infty$, $(\mathbf{U}, \mathbf{V}) = (\mathbf{U}_0, \mathbf{V}_0)$, it means that the two latent coordinate systems are exactly the same in two domains. When $\rho_u = \rho_v = 0$, (3) reduces to the spectral matrix completion method named OptSpace (Keshavan, Montanari, and Oh 2010), which means that we do not make use of any auxiliary knowledge; instead, we achieve matrix completion with the observed ratings of the target domain data only.

Finally, the optimization problem can be solved efficiently via an alternative method, by (a) fixing $\mathbf{U}, \mathbf{V}$, where the inner matrix $\mathbf{B}$ can then be solved analytically (Keshavan, Montanari, and Oh 2010), and (b) fixing $\mathbf{B}$, where $\mathbf{U}, \mathbf{V}$ can be alternatively solved on the Grassman manifold through a projected gradient descent method (Edelman, Arias, and Smith 1999), (Buono and Politi 2004). The complete two-sided transfer learning solution is given in Algorithm 1.

The alternative method used in Algorithm 1 monotonically decreases the objective function (3), and hence ensures convergence to local minimum. The time complexity of CST and other baseline methods are reported in Table 2, where k is the iteration number, $p(p > n, m)$ is the number of non-zeno entries in the rating matrix $\mathbf{R}$, and d is the number of of latent features. Note, d and k are usually quite small, i.e. $d < 20$, $k < 50$ in our experiments.

Experimental Results

Data Sets and Evaluation Metrics

We evaluate the proposed method using two movie rating data sets Netflix[1] and MovieLens[2]. The Netflix rating data contains more than 10^8 ratings with values in $\{1, 2, 3, 4, 5\}$, which are given by more than 4.8×10^5 users on around 1.8×10^4 movies. The MovieLens rating data contains more than 10^7 ratings with values in $\{1, 2, 3, 4, 5\}$, which are given by more than 7.1×10^4 users on around 1.1×10^4 movies. The data set used in the experiments is constructed as follows,

- we first randomly extract a $10^4 \times 10^4$ dense rating matrix $\mathfrak{R}$ from the Netflix data, and take the sub-matrices $\mathbf{R} = \mathfrak{R}_{1\sim 5000,1\sim 5000}$ as the target rating matrix, and $\mathbf{R}^{(1)} = \mathfrak{R}_{1\sim 5000,5001\sim 10000}$ as the user side auxiliary data, so that $\mathbf{R}$ and $\mathbf{R}^{(1)}$ share only common users but not common items;
- we then extract an item side auxiliary data $\mathbf{R}^{(2)}$ of size 5000×5000 from the MovieLens data by identifying the movies appearing both in MovieLens and Netflix. Clearly, $\mathbf{R}$ and $\mathbf{R}^{(2)}$ share only common items but no users;
- finally, to simulate heterogenous auxiliary and target domain data, we adopt the pre-processing approach (Sindhwani et al. 2009) on $\mathbf{R}^{(1)}$, $\mathbf{R}^{(2)}$, by relabeling $1, 2, 3$ ratings in $\mathbf{R}^{(1)}, \mathbf{R}^{(2)}$ as 0, and then $4, 5$ ratings as 1.

In all of our experiments, the target domain rating set from $\mathbf{R}$ is randomly split into training and test sets, T_R, T_E, with 50% ratings, respectively. $T_R, T_E \subset \{(u, i, r_{ui}) \in \mathbb{N} \times \mathbb{N} \times \{1, 2, 3, 4, 5\} | 1 \leq u \leq n, 1 \leq i \leq m\}$. T_E is kept unchanged, while different number of observed ratings for each user, 10, 20, 30, 40, are randomly picked from T_R for training, with different sparsity levels of 0.2%, 0.4%, 0.6%, 0.8% correspondingly. The final data set used in the experiments is summarized in Table 1.

Table 1: Description of target and auxiliary data (all matrices are of size 5000×5000).

Data set		Form	Sparsity
$\mathbf{R}$	target (training)	1-5	<1.0%
	target (test)	1-5	11.3%
$\mathbf{R}^{(1)}$	auxiliary (user side)	0/1	10.0%
$\mathbf{R}^{(2)}$	auxiliary (item side)	0/1	9.5%

We adopt two evaluation metrics: Mean Absolute Error (MAE) and Root Mean Square Error (RMSE),

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

where r_{ui} and $\hat{r}_{ui}$ are the true and predicted ratings, respectively, and $|T_E|$ is the number of test ratings. In all experiments, we run 10 random trials when generating the required number of observed ratings for each user from the target training rating set T_R, and averaged results are reported.

Baselines and Parameter Settings

We compare our CST method with two non-transfer learning methods: the average filling method (AF), LFM (Bell and Koren 2007), and the transfer learning method CMF (Singh and Gordon 2008). Note, the codebook in CBT (Li, Yang, and Xue 2009a) and RMGM (Li, Yang, and Xue 2009b) constructed from a $0/1$ matrix of implicit feedbacks is always a full matrix of 1s only, and does not reflect any cluster-level rating patterns, hence both CBT and RMGM are not applicable to our problem. As mentioned before, CST reduces to the spectral matrix completion method, OptSpace (Keshavan, Montanari, and Oh 2010) when no auxiliary data exist. Thus, we also report the performance of OptSpace when studying the effect of using the auxiliary data.

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

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

where $\bar{r}_{u\cdot} = \sum_i y_{ui} r_{ui} / \sum_i y_{ui}$ is the average rating of user u, $\bar{r}_{\cdot i} = \sum_u y_{ui} r_{ui} / \sum_u y_{ui}$ is the average rating of item i, $b_{u\cdot} = \sum_i y_{ui}(r_{ui} - \bar{r}_{\cdot i}) / \sum_i y_{ui}$ is the bias of user u, $b_{\cdot i} = \sum_u y_{ui}(r_{ui} - \bar{r}_{u\cdot}) / \sum_u y_{ui}$ is the bias of item i, and $\bar{r} = \sum_{u,i} y_{ui} r_{ui} / \sum_{u,i} y_{ui}$ is the global average rating. We use $\hat{r}_{ui} = \bar{r} + b_{u\cdot} + b_{\cdot i}$ as it performs best in our experiments.

For LFM, CMF, CST and OptSpace, different latent dimensions $\{5, 10, 15\}$ are tried; for LFM and CMF, different tradeoff parameters $\{0.001, 0.01, 0.1, 1, 10, 100\}$ and $\{0.1, 0.5, 1, 5, 10\}$ are tried, respectively, and best results are reported; for CST, the performance are quite stable using different tradeoff parameters, and the result using $\rho_u/n = \rho_v/m = 1$ are reported.

Results

The best results of using different parameters as described in the previous section are reported in Table 2. We can make the following observations:

- CST performs significantly better than all other baselines in all sparsity levels;
- for the non-transfer learning methods of AF and LFM, we can see that AF always performs better than LFM, which shows the usefulness of smoothing for sparse data;
- for the transfer learning method, CMF beats two non-transfer learning baselines, AF, LFM, in all sparsity levels, which demonstrates the usefulness of transfer learning methods for sparse data; however, we can see that CMF is still worse than CST, which can be explained

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

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

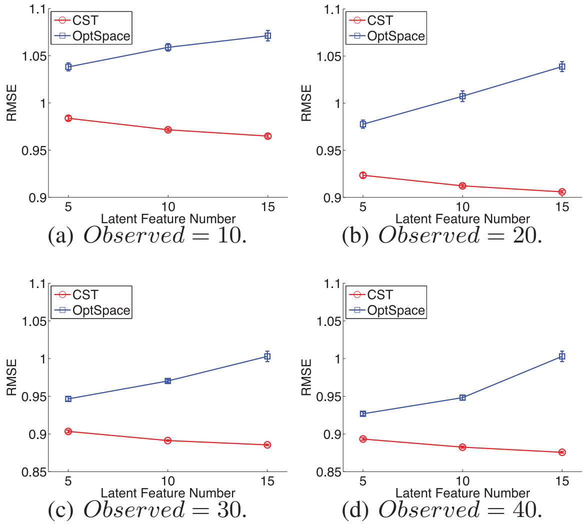
Table 2: Prediction performance of average filling (AF), latent factorization model (LFM), collective matrix factorization (CMF), and coordinate system transfer (CST). Numbers in boldface (i.e. **0.7481**) are the best results among all methods.

	Observed (sparsity)	Without Transfer AF	Without Transfer LFM	With Transfer CMF	With Transfer CST
MAE	10 (0.2%)	0.7764 ± 0.0008	0.8934 ± 0.0005	0.7642 ± 0.0024	**0.7481** ± 0.0014
	20 (0.4%)	0.7430 ± 0.0006	0.8243 ± 0.0019	0.7238 ± 0.0012	**0.7056** ± 0.0008
	30 (0.6%)	0.7311 ± 0.0005	0.7626 ± 0.0008	0.7064 ± 0.0008	**0.6907** ± 0.0006
	40 (0.8%)	0.7248 ± 0.0004	0.7359 ± 0.0008	0.6972 ± 0.0007	**0.6835** ± 0.0008
RMSE	10 (0.2%)	0.9853 ± 0.0011	1.0830 ± 0.0000	0.9749 ± 0.0033	**0.9649** ± 0.0019
	20 (0.4%)	0.9430 ± 0.0006	1.0554 ± 0.0016	0.9261 ± 0.0014	**0.9059** ± 0.0013
	30 (0.6%)	0.9280 ± 0.0005	0.9748 ± 0.0012	0.9058 ± 0.0009	**0.8855** ± 0.0010
	40 (0.8%)	0.9202 ± 0.0003	0.9381 ± 0.0010	0.8955 ± 0.0007	**0.8757** ± 0.0011
Time Complexity		$O(p)$	$O(kpd^2 + k\max(n,m)d^3)$	$O(kpd^2 + k\max(n,m)d^3)$	$O(kpd^3 + kd^6)$

by the more flexibility of tri-factorization method used in CST in (3).

Note that by setting the tradeoff parameter ρ_u and ρ_v in (3) to 0, the CST model become equivalent to the OPTSpace model (Keshavan, Montanari, and Oh 2010), which considers no auxiliary domain information, neither initialization nor regularization. To gain a deeper understanding of CST and more carefully assess the benefit from the auxiliary domain, we compared the performance of CST and OPTSpace at different data sparsity levels when the parameter d, the number of latent features, is increased from 5 to 15. The results in RMSE are shown in Figure 2 (the results in MAE are quite similar). We can see that:

- the performance of OPTSpace consistently deteriorates as d increases, which is due to that more flexible models are more likely to suffer from overfitting given sparse data;
- in contrast to OPTSpace, CST consistently improves as d increases which demonstrates how the auxiliary domain knowledge based initialization and regularization techniques can help avoid overfitting even for highly flexible models;
- the relative improvement is more significant when fewer ratings are observed, which confirms the effectiveness of CST in alleviating data sparsity.



(a) $Observed = 10$. (b) $Observed = 20$.

(c) $Observed = 30$. (d) $Observed = 40$.

Figure 2: Comparison of CST and OPTSpace at different sparsity level with increasing d .

Related Work

Latent factorization model (LFM) (Bell and Koren 2007) is a widely used method in collaborative filtering, which seeks an appropriate low-rank approximation of the rating matrix $\mathbf{R}$ with two latent feature matrices, one for users and one for items. For any missing entry in $\mathbf{R}$, it can be predicted by the production of those two latent feature matrices.

Collective Matrix Factorization (CMF) (Singh and Gordon 2008) is a multi-task learning (MTL) (Caruana 1997) version of LFM, which jointly factorizes multiple matrices with correspondences between rows and columns while sharing latent features of matching rows and columns in different matrices. Note, there are at least three differences compared to the CST method, (a) CMF is an MTL style algorithm, which does not distinguish auxiliary domain from target domain, whereas CST is an adaptation style algorithm, which focuses on improving performance in the target domain by transferring knowledge from but not to the auxiliary domain. Hence CST is more efficient especially when the auxiliary data is dense, and more secure for privacy considerations, (b) CMF is less flexible in that it requries $\mathbf{R}, \mathbf{R}^{(1)}$ and $\mathbf{R}, \mathbf{R}^{(2)}$ to share exactly the same user and item latent features, respectively, while CST relaxes this assumption and only requires two corresponding coordinate systems to be similar, (c) CMF is a bi-factorization method (i.e. $\mathbf{R} = \mathbf{U}\mathbf{V}^T$), so the latent features $\mathbf{U}, \mathbf{V}$ have to capture both domain dependent effects and independent shared knowledge, while CST inherits the flexibility of tri-factorization (i.e., $\mathbf{R} = \mathbf{U}\mathbf{B}\mathbf{V}^T$) method (Ding et al. 2006) by absorbing domain dependent effects such as advertisements and promotions into the inner matrix $\mathbf{B}$, while $\mathbf{U}, \mathbf{V}$ are principle coordinates of users and items that are more consistent across different domains.

Codebook Transfer (CBT) (Li, Yang, and Xue 2009a) is a recently developed heterogenous transfer learning method for collaborative filtering, which contains two steps of codebook construction and codebook expansion, and achieves knowledge transfer with the assumption that both auxiliary and target data share the cluster-level rating patterns (codebook). Rating-Matrix Generative Model (RMGM) (Li, Yang, and Xue 2009b) is derived and extended from the FMM generative model (Si and Jin 2003), and we can consider RMGM as an MTL version of CBT with the same assumption. Note, both CBT and RMGM are limited to explicit rating matrices only, and can not achieve knowledge transfer from an implicit rating matrix with values of 0/1 to an explicit one with values of 1-5, as it requires two rating matrices to share the cluster-level rating patterns. Also, CBT and RMGM can neither make use of user side nor item side shared information, and only take a general explicit rating matrix as its auxiliary input. Hence, both CBT and RMGM are not applicable to the problem studied in this paper.

Table 3: Summary of some related work.

		Algorithm Style	
		Adaptation	Multi-Task
Knowledge	Codebook	CBT	RMGM
	Latent Features	***CST***	CMF

We summarize the above related work in Table 3 from the perspective of transferred knowledge (what to transfer) and algorithm style (how to transfer). CST can also be considered as a two-sided extension in matrix form of one-sided domain adaptation methods in vector form (Kienzle and Chellapilla 2006), which is proposed not for collaborative filtering but classification problems and achieve knowledge transfer via incorporating the model parameters learned from the auxiliary domain as prior knowledge.

Conclusions and Future Work

In this paper, we presented a novel transfer learning method, CST, for alleviating the data sparsity problem in collaborative filtering. Our method first finds a subspace where coordinate systems are used for knowledge transfer, then uses the transferred knowledge to adapt to the target domain data. The novelty of our algorithm includes using both the user and item side information in an integrated way. Experimental results show that CST performs significantly better than several state-of-the-art methods at various sparsity levels. Our experimental study clearly demonstrates (a) the usefulness of transferring two coordinate systems from the auxiliary data (what to transfer), and (b) the effectiveness of incorporating two-sided auxiliary knowledge via a regularized tri-factorization method, thus addressing the how to transfer question for CF. For future works, we will study on how to extend CST in other heterogeneous settings, e.g., for transferring knowledge between movies, music, books, etc.

Acknowledgement

We thank the support of RGC-NSFC Joint Research Grant N_HKUST624/09.

References

Bell, R. M., and Koren, Y. 2007. Scalable collaborative filtering with jointly derived neighborhood interpolation weights. In *ICDM*, 43–52.

Buono, N. D., and Politi, T. 2004. A continuous technique for the weighted low-rank approximation problem. In *ICCSA (2)*, 988–997.

Caruana, R. 1997. Multitask learning. *Machine Learning* 28(1):41–75.

Deerwester, S.; Dumais, S. T.; Furnas, G. W.; Landauer, T. K.; and Harshman, R. 1990. Indexing by latent semantic analysis. *Journal of the American Society for Information Science* 41:391–407.

Ding, C. H. Q.; Li, T.; Peng, W.; and Park, H. 2006. Orthogonal nonnegative matrix tri-factorizations for clustering. In *KDD*, 126–135.

Edelman, A.; Arias, T. A.; and Smith, S. T. 1999. The geometry of algorithms with orthogonality constraints. *SIAM Journal on Matrix Analysis and Applications* 20(2):303–353.

Hu, Y.; Koren, Y.; and Volinsky, C. 2008. Collaborative filtering for implicit feedback datasets. In *ICDM*, 263–272.

Keshavan, R.; Montanari, A.; and Oh, S. 2010. Matrix completion from noisy entries. In *NIPS*, 952–960.

Kienzle, W., and Chellapilla, K. 2006. Personalized handwriting recognition via biased regularization. In *ICML*, 457–464.

Li, B.; Yang, Q.; and Xue, X. 2009a. Can movies and books collaborate? cross-domain collaborative filtering for sparsity reduction. In *IJCAI*, 2052–2057.

Li, B.; Yang, Q.; and Xue, X. 2009b. Transfer learning for collaborative filtering via a rating-matrix generative model. In *ICML*, 617–624.

Long, B.; Zhang, Z. M.; Wu, X.; and Yu, P. S. 2007. Relational clustering by symmetric convex coding. In *ICML*, 569–576.

Nathanson, T.; Bitton, E.; and Goldberg, K. 2007. Eigentaste 5.0: constant-time adaptability in a recommender system using item clustering. In *RecSys*, 149–152.

Pan, S. J., and Yang, Q. 2009. A survey on transfer learning. *IEEE TKDE*. IEEE Computer Society, http://doi.ieeecomputersociety.org/10.1109/TKDE.2009.191.

Resnick, P.; Iacovou, N.; Suchak, M.; Bergstrom, P.; and Riedl, J. 1994. Grouplens: An open architecture for collaborative filtering of netnews. In *CSCW*, 175–186.

Si, L., and Jin, R. 2003. Flexible mixture model for collaborative filtering. In *ICML*, 704–711.

Sindhwani, V.; Bucak, S.; Hu, J.; and Mojsilovic, A. 2009. A family of non-negative matrix factorizations for one-class collaborative filtering. In *RecSys '09: Recommender based Industrial Applications Workshop*.

Singh, A. P., and Gordon, G. J. 2008. Relational learning via collective matrix factorization. In *KDD*, 650–658.