Pattern Recognition 94 (2019) 96–109

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/patcog

A cross-domain collaborative filtering algorithm with expanding user and item features via the latent factor space of auxiliary domains

Xu Yu[a], Feng Jiang[a], Junwei Du[a], Dunwei Gong[a,b,*]

[a] School of Information Science and Technology, Qingdao University of Science and Technology, Qingdao, Shandong 266061, China
[b] School of Information and Control Engineering, China University of Mining and Technology, Xuzhou, Jiangsu 221116, China

ARTICLE INFO

Article history:
Received 6 October 2018
Revised 25 January 2019
Accepted 18 May 2019
Available online 18 May 2019

Keywords:
Cross-domain collaborative filtering
Feature expansion
Funk-SVD decomposition
Classification
Latent factor space

ABSTRACT

Cross-domain collaborative filtering, which transfers rating knowledge across multiple domains, has become a new way to effectively alleviate the sparsity problem in recommender systems. Different auxiliary domains are generally different in the importance to the target domain, which is hard to evaluate using previous approaches. Besides, most recommender systems only take advantage of information from user- or item-side auxiliary domains. To overcome these drawbacks, we propose a cross-domain collaborative filtering algorithm with expanding user and item features via the latent factor space of auxiliary domains in this paper. In the proposed algorithm, the recommendation problem is first formulated as a classification problem in the target domain, which takes user and item location as the feature vector, their rating as the label. Then, Funk-SVD decomposition is employed to extract extra user and item features from user- and item-side auxiliary domains, respectively, with the purpose of expanding the two-dimensional location feature vector. Finally, a classifier is trained using the C4.5 decision tree algorithm for predicting missing ratings. The proposed algorithm can make full use of user- and item-side information. We conduct extensive experiments and compare the proposed algorithm with various state-of-the-art single- and cross-domain collaborative filtering algorithms. The experimental results show that the proposed algorithm has advantages in terms of four different evaluation metrics.

1. Introduction

In recent years, recommender systems [1–3] are widely used in e-commerce and online social media, and their majority offer recommendations for items belonging to a single domain. The collaborative filtering (CF) algorithm [4–7], which is boiled down to analyzing tabular data in a user-item rating matrix, is one of the most widely used methods in recommender systems.

Users are, however, generally unwilling to rating items in real-world recommender systems, which results in a sparse rating matrix. The sparsity problem has become a major bottleneck for most CF algorithms. To address the problem, researchers have recently proposed a variety of cross-domain CF (CDCF) algorithms [8], which exploit knowledge from auxiliary domains to improve recommendations on a target domain. Previous CDCF algorithms can be generally classified into the following five categories, i.e., only user-side transfer [9–11], only item-side transfer [12], two-side (simultaneous) transfer [13], two-side (respective) transfer [14], and neither user-side nor item-side transfer [15–18].

Only user- or item-side transfer assumes that the target and the auxiliary domains share the same users or items, which commonly appears in real-world applications. However, previous work in this scenario either treats different auxiliary domains equally, or has difficulty in computing an optimal weights assignment for the auxiliary domains. Most importantly, all the above studies only make use of one-side shared information. Their performances will be further improved if shared information from the other side is fully exploited.

Two-side (simultaneous) transfer requires that the target and the auxiliary domains simultaneously share the same users and items. The two-side (simultaneous) transfer can provide precise information on the mapping between auxiliary and target data, which is beneficial to a high performance. There are, however, scarce scenarios in real-world applications where the assumption is held.

Two-side (respective) transfer assumes that there exist two kinds of auxiliary domains, user- and item-side auxiliary domains, which share the same users and items with the target domain,

* Corresponding author.
E-mail address: dwgong@vip.163.com (D. Gong).

https://doi.org/10.1016/j.patcog.2019.05.030
0031-3203/

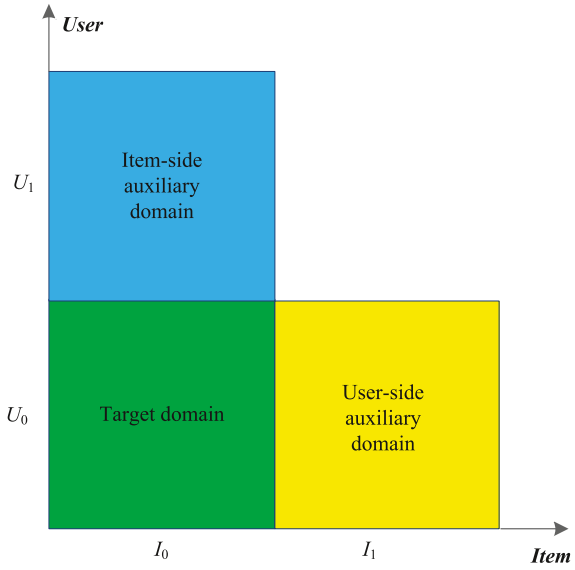


Fig. 1. A special case of the two-side (respectively) transfer scenario.

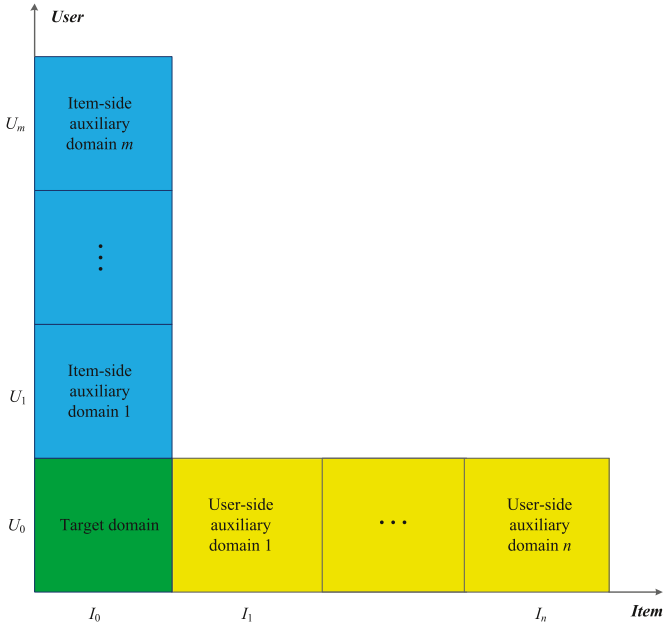


Fig. 2. The two-side (respectively) transfer setting.

respectively. Note that each kind of auxiliary domain may contain more than one domain. A special case of this scenario is illustrated in Fig. 1. There is only one auxiliary domain for each kind of auxiliary domain in this special case. For this special case, Pan et al. [14] proposed a coordinate system transfer (CST) algorithm. CST requires, however, that the target domain and the user-side auxiliary domain have homogeneous items such as movies, suggesting that it cannot address the case of heterogeneous items such as movie and music. Hence, the CST algorithm is not a real cross-domain algorithm.

To address this drawback, Yu et al. [19] first extract intrinsic user and item features from the auxiliary domain, and then transfer them to improve the recommendation performance of the target domain. However, since intrinsic features are usually domain independent, the representation ability of them for the target domain is limited, which leads to little improvement in recommendation performance. In addition, the above two algorithms can only deal with the case of one user-side and one item-side auxiliary domains.

Neither user-side nor item-side transfer does not require shared users or items. Along this line, previous studies [15–18] require two rating matrices to share the cluster-level rating patterns. In addition, they cannot make use of shared information from the user or item side.

Since the two-side (respectively) setting can exploit more shared information, it is a good idea for designing a cross-domain recommendation algorithm. To address the drawbacks of previous models in this setting, we propose a two-side cross-domain collaborative filtering algorithm with Expanding User and Item Features via the latent factor space of auxiliary domains (TSEUIF). To fulfill this task, the location of user-item in the target domain is regarded as the feature vector, and the corresponding rating as the label. In this way, the recommendation problem can be formulated as a classification problem. Given the fact that the two-dimensional feature vector has a difficulty in discriminating different ratings, i.e., 1, 2, 3, 4, and 5, more features are required when classifying.

Note that the first and the second dimensions of the feature vector are the locations of users and items, respectively, which represent the user and the item features in the recommender system. Inspired by this observation, we attempt to expand the feature vector in the target domain with important features of users or items. Due to privacy preservation, we have difficulties in extracting explicit user features, such as “Age” and “Occupation”. Given the fact that information retrieval is nontrivial, we do not extract explicit item features, such as “Film Genres”, “Main Actor/Actress” and “Director” for a movie. Instead we extract user and item features from the user and the item latent factor spaces, respectively.

We assume that there exist two kinds of auxiliary domains, i.e., user- and item-side auxiliary domains, where user- and item-side auxiliary domains share the same users and items with the target domain, respectively. Moreover, we assume that there are dense rating data in these auxiliary domains. Fig. 2 depicts the scenario corresponding to the above assumptions. For this scenario, it is possible to extract the user and item features from the user- and item-side auxiliary domains, respectively. Following that, we regard the user and the item latent vectors obtained by Funk-SVD decomposition [20] from the auxiliary domains as the extra features, and add them to the feature vector in the target domain. In this way, the feature vector can be effectively expanded, which guarantees a better performance in the following training of classification model. Details of the proposed model will be given in Section 3.

Unlike inferring intrinsic features, the proposed TSEUIF model extracts domain-dependent features from multiple auxiliary domains so as to expand the feature set. Though the new expanded feature set inevitably contains many redundant and irrelevant attributes, it indeed improves the representation ability of the feature set, which makes it possible to achieve a better recommendation performance by feature selection. Hence, the proposed model is not an incremental version of the previous work, but a novel recommendation model based on new feature extract and selection methods.

The proposed algorithm has the following tri-fold advantages: (a) it can effectively transfer knowledge from auxiliary domains, and evaluate the importance of auxiliary domains, (b) it can well alleviate the sparsity problem, and (c) it can well solve the cold-start problem.

The remainder of this paper is organized as follows. Section 2 reviews related work on CDCF algorithms. The proposed TSEUIF algorithm is detailed in Section 3. In Section 4, we conduct extensive experiments to evaluate the proposed algorithm.

Finally, Section 5 concludes the whole paper, and directs several topics to be further researched.

2. Related work

Some early studies on CDCF are carried out by Berkovsky et al. [9], who deploy a variety of mediation approaches for importing and aggregating user rating vectors from various domains. As mentioned before, previous CDCF algorithms can be generally classified into the following five categories, i.e., only user-side transfer [9–11], only item-side transfer [12], two-side (simultaneous) transfer [13], two-side (respective) transfer [14], and neither user-side nor item-side transfer [15–18].

Among previous work related to only user-side transfer and only item-side transfer, Berkovsky et al. [9] present their neighborhood-based CDCF (N-CDCF) versions, which is a cross-domain version of neighborhood-based CF (N-CF) algorithm [21]. Given the fact that neighborhood-based CF can generally be divided into the following two types, i.e., user-based nearest neighbor and item-based nearest neighbor, N-CDCF can also be further divided into two types, i.e., neighborhood-based CDCF from users (N-CDCF-U) and neighborhood-based CDCF from items (N-CDCF-I). The Pearson correlation is used to calculate the similarities, and then the unknown ratings are predicted according to a weighted average strategy [21].

Hu et al. [10] propose a matrix factorization-based CDCF (MF-CDCF) algorithm, which is a cross-domain version of matrix factorization algorithms. In MF-CDCF, an augmented rating matrix is constructed by horizontally concatenating all matrices. As a result, the proposed algorithm can obtain the latent user and item factors for prediction. N-CDCF and MF-CDCF are developed straightforward from a single domain, which generally assumes the homogeneity of items. Items in different domains may, however, be heterogeneous, suggesting that N-CDCF and MF-CDCF have a difficulty in taking this fact into account.

Given the fact that different auxiliary domains have different correlations with the target domain, Yu et al. [11] present a cross-domain recommendation method using a linear decomposition model, which compute the local similarity in the target domain according to the total similarity and the local similarities in the auxiliary domains. Singh and Gordon [12] propose a collective matrix factorization (CMF) algorithm, which jointly factorizes multiple matrices with correspondences between rows and columns while sharing latent features of matching rows and columns in different matrices. Hu et al. [10] further present a Cross-Domain Tensor Factorization (CDTF) algorithm, which effectively exploits user preferences on items among different domains using the user-item-domain relation. A major problem of tensor factorization is, however, that the time complexity of this approach is $O(k^m)$, where k is the number of factors, and m is the number of domains. CMF and CDTF assign different weights to different auxiliary domains, which makes them superior to N-CDCF and MF-CDCF. However, CMF does not provide a mechanism for seeking optimal weights for auxiliary domains. Although CDTF assigns weights based on genetic algorithms (GAs), its performance is largely determined by the initial population of GAs.

Among previous studies of two-side (simultaneous) transfer, Pan et al. [13] present a transfer by collective factorization (TCF) model to transfer knowledge from auxiliary data of explicit binary ratings (like and dislike), which alleviates the data sparsity problem in numerical ratings. TCF requires that users and items of the target and the auxiliary matrices should be aligned. Hence, it is not applicable to the problem studied in this paper.

Among previous work of two-side (respective) transfer, Pan et al. [14] present a coordinate system transfer (CST) algorithm. In CST, the sparse SVD [22] on auxiliary data, $\mathbf{R}^{(i)}$, $i = 1, 2,$ is adopted in the following form

$$\min_{\mathbf{U}^{(i)},\mathbf{V}^{(i)},\mathbf{B}^{(i)}} \left\| \mathbf{Y}^{(i)} \odot (\mathbf{R}^{(i)} \sim \mathbf{U}^{(i)}\mathbf{B}^{(i)}{\mathbf{V}^{(i)}}^T) \right\|_F^2 \tag{1}$$

where $\mathbf{Y} \in \{0,\ 1\}^{n \times m}$ is an indicator matrix, with $y_{ui} = 1$ if u has rated i, and $y_{ui} = 0$ otherwise; $\odot$ means the entry-wise product; $\mathbf{B}^{(i)} = diag(\sigma_1^{(i)}, \ldots, \sigma_j^{(i)}, \ldots, \sigma_d^{(i)})$ is a diagonal matrix, $\sigma_1^{(i)} \geq \sigma_2^{(i)} \geq \ldots \geq \sigma_d^{(i)} \geq 0$ are eigenvalues, and ${\mathbf{U}^{(i)}}^T\mathbf{U}^{(i)} = \mathrm{I}$ and ${\mathbf{V}^{(i)}}^T\mathbf{V}^{(i)} = \mathrm{I}$ ensure that their columns are orthonormal.

Then, the coordinate systems (or the latent features) extracted from auxiliary data are incorporated into the target factorization system, i.e., $\mathbf{Y} \odot (\mathbf{R} \sim \mathbf{U}\mathbf{B}\mathbf{V}^T)$, via the following two biased penalty terms,

$$\frac{\rho_u}{2}\left\|\mathbf{U} - \mathbf{U}^{(1)}\right\|_F^2 + \frac{\rho_v}{2}\left\|\mathbf{V} - \mathbf{V}^{(2)}\right\|_F^2 \tag{2}$$

where ρ_u and ρ_v are trade-off parameters. The two terms in Eq. (2) are employed to constrain the latent feature matrices $\mathbf{U} \in \mathbb{R}^{n \times d}$ and $\mathbf{V} \in \mathbb{R}^{m \times d}$ to be similar to $\mathbf{U}^{(1)}$ and $\mathbf{V}^{(2)}$, respectively.

However, CST is not a real cross-domain model due to its incapability of addressing the case of heterogeneous items. In contrast, the one proposed in this paper is indeed a cross-domain model. That is a major difference between CST and the model proposed in this paper. We make an explanation in the following.

Let m and n be the number of users and items, respectively. In addition, let d be the dimensionality of latent factors. Then, SVD has the following form of factorization.

$$R = UBV^T = \begin{bmatrix} p_{11} & p_{12} & \cdots & p_{1d} \\ p_{21} & p_{22} & \cdots & p_{2d} \\ \vdots & \vdots & \cdots & \vdots \\ p_{m1} & p_{m2} & \cdots & p_{md} \end{bmatrix} \begin{bmatrix} \sigma_1 & & & \\ & \sigma_2 & & \\ & & \ddots & \\ & & & \sigma_d \end{bmatrix} \begin{bmatrix} q_{11} & q_{21} & \cdots & q_{n1} \\ q_{12} & q_{22} & \cdots & q_{n2} \\ \vdots & \vdots & \cdots & \vdots \\ q_{1d} & q_{2d} & \cdots & q_{nd} \end{bmatrix} \tag{3}$$

It is clear that r_{ij} can be calculated as follows.

$$r_{ij} = \sigma_1 p_{i1} q_{j1} + \sigma_2 p_{i2} q_{j2} + \cdots + \sigma_d p_{id} q_{jd} \tag{4}$$

If $\sigma_1 = \sigma_2 = \cdots = \sigma_d = 1$, SVD decomposition will be degraded to Funk-SVD decomposition.

According to Eq. (3), the latent factors in both U and V have the dimensionality of d. Moreover, B is a diagonal matrix. Hence, like Funk-SVD decomposition, users and items have been projected to the same latent factor space for SVD decomposition. In other words, U and V share the same latent factor space.

Taking a movie recommender system as an example, for SVD decomposition, each row of V measures the distribution of a movie on the latent factors, e.g., science fiction, comedy, action, and love, and each row of U represents a user taste on these latent factors. Since a user taste heavily depends on the kind of items, U is also dependent on the kind of items. Thus, CST requires that the target and the user-side auxiliary domains have homogeneous items, so as to transfer knowledge via the biased regularization term, $\left\|\mathbf{U} - \mathbf{U}^{(1)}\right\|_F^2$, in Eq. (2). If the target and the user-side auxiliary domains are movies and music, respectively, the transfer of $\mathbf{U}^{(1)}$ to the target domain will be incorrect. That is why CST cannot address the case of heterogeneous items. Pan stated that “CST

cannot well address the case of heterogeneous items" in Conclusion of [14].

Yu et al. [19] proposed a cross-domain recommendation model based on intrinsic feature extraction. Since intrinsic features are domain independent, they can be shared among different domains. Hence, they can be transferred from the auxiliary domain to the target domain so as to improve the recommendation performance. However, the representation ability of intrinsic features for the target domain is limited. The previous work just inferred intrinsic features, such as age information, from the user-side auxiliary domain, which makes it difficult to obtain more user features related to the target domain even if there are multiple auxiliary domains.

Among the studies related to neither user-side nor item-side transfer, Li et al. [15] propose a CBT algorithm. CBT achieves knowledge transfer with the assumption that both auxiliary and target data share the cluster-level rating patterns. Further, Li et al. [16] present a RMGM algorithm, which shares knowledge in the form of the latent cluster-level ratings. Parallel to CBT and RMGM, Gao et al. [17] present a cross-domain recommendation algorithm named CLFM for Cyber-physical systems (CPS). CLFM takes the diversity across domains into account. Zhang et al. [18] propose a cross-domain recommendation model with Consistent Information Transfer (CIT). In CIT, domain adaptation techniques are used to maintain consistency during the transfer learning process. All these models assume that the items in an auxiliary domain (e.g. books) are related to the target domain (e.g. movies). Hence, they are not applicable to the scenario studied in this paper.

3. The proposed algorithm

3.1. Problem formulation

Assume that D_1 is the target domain, and U_1 and I_1 are the sets of users and items, respectively, in this domain. We formulate a recommendation problem in D_1 by a function, y: $U_1 \times I_1 \rightarrow R$. Given the fact that each user-item interaction, $(u, i, r) \in U_1 \times I_1 \times \{1, 2, 3, 4, 5\}$, can be represented with a feature vector, (L_u, L_i), and a class label, r, where L_u and L_i are the locations of user u and item i, respectively, if we regard each user-item interaction as a training sample, the recommendation problem can be formulated as a classification problem, shown as Fig. 3.

3.2. Feature vector expansion

The training samples of the formulated classification problem have only two trivial location features, which is, however, inadequate to discriminate different ratings. Our experiments have demonstrated that a classification model with a two-dimensional feature vector has a difficulty in achieving satisfactory results, suggesting that more features are required for better classification. Since the user-side auxiliary domains, $A_1, A_2, \cdots, A_m$, contain dense rating data and share the same user set with the target domain, it is possible to extract relatively accurate user features from them. Similarly, we can also extract relatively accurate item features from the item-side auxiliary domains, $B_1, B_2, \cdots, B_n$.

Due to the problems of privacy preservation and information retrieval, it is difficult to acquire explicit user and item features. Hence, our solution is to extract user and item features from the latent factor space of the user-side and item-side auxiliary domains. We first employ Funk-SVD decomposition to obtain the user and the item latent vectors from the auxiliary domains. A detailed review on Funk-SVD decomposition model will be given in Section 3.3. Then, we expand the two-dimensional feature vector (L_u, L_i) in the target domain with the obtained latent vectors from the auxiliary domains. Given a user-item interaction, $(u, i, r) \in U_1 \times I_1 \times \{1, 2, 3, 4, 5\}$, in the target domain, we expand the location feature vector, (L_u, L_i), in the target domain using the latent vectors of user u and item i from the auxiliary domains. Thus, (L_u, L_i) can be expanded as $(L_u, L_i, p_u^1, \ldots, p_u^m, q_i^1, \ldots, q_i^n)$, where p_u^s, $s = 1, \ldots, m$, represents the latent vector of user u in the s-th user-side auxiliary domain; q_i^t, $t = 1, \ldots, n$, is the latent vector of item i in the t-th item-side auxiliary domain.

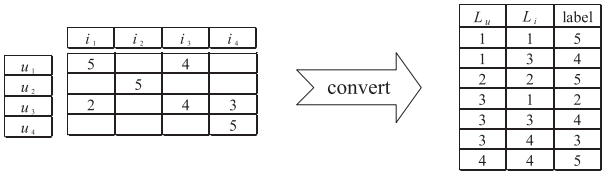


Fig. 3. Formulate the recommendation problem into a classification problem.

Note that different user IDs may correspond to different rating distributions, which indicates the rating bias of users. For example, user 1 has a rating distribution of (0.7,0.2,0,0,0.1) corresponding to ratings 1,2,3,4,5, respectively, and user 2 has a rating distribution of (0.1,0.2,0,0,0.7). The rating distributions indicate that user 1 tends to score low, while user 2 tends to score high. Thus, L_u is relevant to the label variable.

A more formal way to explain the relevance between L_u and the label variable is the "information gain". Please refer to [23] for details about how to compute the information gain. Generally, the information gain is in the range of [0,1], and a large value indicates a strong relevance. If the value equals 0, L_u will be uncorrelated to the label variable. Since the rating distributions of different users and the total rating distributions are usually different, the obtained information gain of L_u is usually larger than 0. Hence, L_u is relevant to the label variable. Similarly, L_i is also relevant to the label variable. Hence, both L_u and L_i are reserved in the feature subset in our model.

Except for the privacy preserve and information retrieval problems, the other reasons why we select the latent vectors to expand the original trivial features can be summarized as follows. Firstly, since we assume that the auxiliary domains contain sufficient rating data, the accuracy of the latent vectors obtained from Funk-SVD decomposition can be guaranteed. Secondly, given the fact that Funk-SVD decomposition can find a low-rank approximation for the rating matrix [20], the user and item latent vectors are a good low-dimensional representation of the user and item features, which will make the classification algorithm more efficient.

3.3. Funk-SVD decomposition model

Funk-SVD decomposition maps users and items to a joint latent factor space of dimensionality f. To learn the latent vectors, $p_u \in R^f$ and $q_i \in R^f$, corresponding to user u and item i, Funk-SVD minimizes the following regularized squared error on the set of known ratings

$$\min_{q*,p*} \sum_{(u,i)\in\kappa} (r_{ui} - q_i^T p_u)^2 + \lambda(\|q_i\|^2 + \|p_u\|^2) \tag{5}$$

where κ is the set of (u, i) pairs whose r_{ui} is known, λ controls the extent of regularization to avoid over-fitting, and is usually determined by cross-validation [24].

An effective approach to solving Eq. (5) is Stochastic Gradient Descent (SGD), which loops through all the ratings in the training set. For each given training case, the system predicts r_{ui} and computes the associated prediction error as follows

$$e_{ui} \stackrel{def}{=} r_{ui} - q_i^T p_u \tag{6}$$

Then, it modifies the parameters by a magnitude proportional to γ, i.e., the learning rate, in the opposite direction of the gradient,

yielding:

$$q_i \leftarrow q_i + \gamma(e_{ui}p_u - \lambda q_i)$$
$$p_u \leftarrow p_u + \gamma(e_{ui}q_i - \lambda p_u) \quad (7)$$

Typically, the learning rate, γ, is constant in the range of [0, 1]. If γ is too small, learning will occur very slowly. On the contrary, if the learning rate is too large, oscillation between inadequate solutions will occur.

In this study, we employ SGD to tackle Eq. (5). Due to its non-convex function, SGD can generally obtain a local optimal solution. To seek a better solution to Eq. (5), Funk-SVD decomposition is performed 10 times with respect to 10 initial solutions, and the optimal solution is chosen. The corresponding algorithm (Funk-SVD-SGD) is given in the form of Algorithm 1 as follows.

Algorithm 1 Funk-SVD-SGD.

Input: the rating matrix, $\mathbf{R}$, $\delta = 10^{-3}$, $T = 1000$, the user number m, the item number n, the rating number l, the dimensionality of latent factor space f.
Output: $p_u(u = 1, \cdots, m)$, $q_i(i = 1, \cdots, n)$.
1: $\tau = 1$; $\Delta J = 100$; $k = 0$; initialize $p_u(u = 1, \cdots, m)$; initialize $q_i(i = 1, \cdots, n)$;
2: $J_1 = \sum_{(u,i)\in\kappa} (r_{ui} - q_i^T p_u)^2 + \lambda(\|q_i\|^2 + \|p_u\|^2)$;
3: do $k \leftarrow (k+1) \bmod l$
4: $\gamma = \frac{1}{\tau}$; $e_{ui} = r_{ui} - q_i^T p_u$; $q_i^j = q_i^j + \gamma(e_{ui}p_u - \lambda q_i)$; $p_u^j = p_u^j + \gamma(e_{ui}q_i - \lambda p_u)$; $(j = 1, \cdots, f)$
5: $\tau = \tau + 1$;
6: $J_2 = \sum_{(u,i)\in\kappa} (r_{ui} - q_i^T p_u)^2 + \lambda(\|q_i\|^2 + \|p_u\|^2)$;
7: $\Delta J = |J_2 - J_1|$;
8: $J_1 = J_2$;
9: until($\Delta J < \delta = 10^{-3}$ or $\tau > T = 1000$);
10: return $p_u(u = 1, \cdots, m)$, $q_i(i = 1, \cdots, n)$;

3.4. Classification problem solving

3.4.1. Feature selection and classification

In our TSEUIF model, we infer a lot of extra user and item features from the auxiliary domains. However, the expanded feature set usually contains many irrelevant and redundant features, which may greatly degrade the performance of a classification algorithm. Hence, feature selection [25] is very necessary before classification. Generally, feature selection models can be divided into three categories, i.e., the filter models, the wrapper models, and the embedded models.

The filter models first carry out feature selection, and then train a classifier. Hence, the feature selection process is independent with the subsequent learning algorithm. Relief-F [26] is a well-known filter model. A "correlation statistic" vector is designed to measure the importance of features, in which each component corresponds to an original feature. Finally, the features corresponding to larger components than the threshold, τ_1, or the features corresponding to the largest k components are selected to compose the feature subset. Please refer to [26] for details about how to compute "correlation statistic".

Unlike the filter models which select important features without involving any learning algorithm, the wrapper models evaluate the selected feature subset according to the performance of a learning algorithm. In other words, the wrapper models aim to select the tailored feature subset that is most advantageous to the performance of a classifier. Sequential Forward Selection (SFS) [27] is a typical wrapper model. The procedure starts with an empty set of attributes as the feature subset. The best original attribute is determined and added to the feature subset. At the subsequent iteration, the best remaining original attribute is added to the set. When the performance on the new feature subset is worse than the current one, or the maximum size of the feature subset is greater than a specified value, SFS will stop.

Since the wrapper models conduct feature selection according to the performance of learning algorithms, they usually perform better than the filter models in terms of the classification performance. However, as the wrapper models train a particular learning algorithm for each subset, they are more computationally intensive than the filter models.

The embedded models combine feature selection and classification in a unified model, which automatically perform feature selection during training. They can usually achieve good feature subsets in short time. Considering both running time and classification performance, we choose an embedded model to solve the converted classification problem in our TSEUIF model.

The decision tree model [23] is a typical embedded model, which can automatically select significant features during training. Hence, it is assigned to solve the feature selection and classification task. In the decision tree model, attributes that do not appear in the final decision tree are irrelevant or redundant, and all the remaining attributes form a feature subset after reduction.

ID3 [23] and C4.5 [28] proposed by Quinlan are two well-known decision tree algorithms. ID3 algorithm can only handle discrete attributes. In addition, it prefers to select attributes having a large number of values. C4.5, a successor of ID3, makes improvements for the two problems. Firstly, it can handle both continuous and discrete attributes. Secondly, it can overcome the selection bias by using the gain ratio. Since the expanded feature set in our model comprises many continuous attributes and two location features having a large number of values, we choose the C4.5 algorithm to construct the decision tree.

3.4.2. Algorithm description and analysis

The flowchart of the proposed model is given in Fig. 4. We also summarize the proposed TSEUIF algorithm in the form of Algorithm 2 as follows.

Algorithm 2 TSEUIF.

Input: a rating matrix, M, in the target domain, m rating matrices, $M_1, M_2, \cdots, M_m$, in the user-side auxiliary domains, n rating matrices, $R_1, R_2, \cdots, R_n$, in the item-side auxiliary domains.
Output: the missing ratings in the target domain.
1: Formulate the recommendation problem in the target domain into a classification problem;
2: Perform Funk-SVD-SGD decomposition in each user-side auxiliary domain to obtain the user latent vectors;
3: Perform Funk-SVD-SGD decomposition in each item-side auxiliary domain to obtain the item latent vectors;
4: Expand the two-dimensional feature vectors using the corresponding user and item latent vectors;
5: Train a classifier based on the obtained training set with the C4.5 decision tree algorithm;
6: Predict the missing ratings.

Our classification-based model has three advantages. Firstly, it is difficult for traditional recommendation models to evaluate the importance of auxiliary domains, and to transfer important information from auxiliary domains to the target domain. However, in our classification-based recommendation model, by representing the auxiliary domains with latent factors, the knowledge transfer problem can be well solved by feature expansion, and the tough evaluation of the importance of auxiliary domains can be well solved by feature selection.

Secondly, the target domain contains a large number of users and items, but it is usually with a small rating density, which

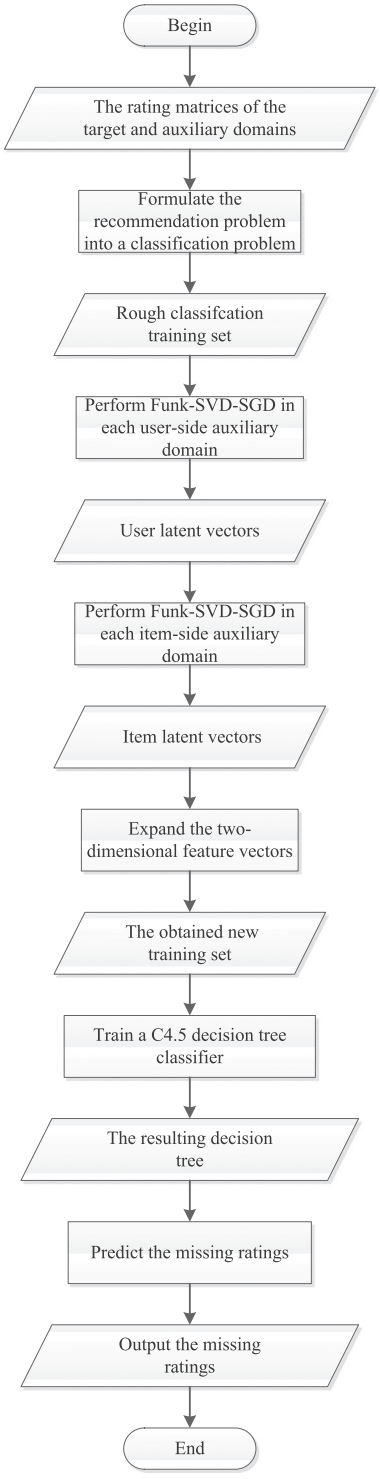


Fig. 4. The flowchart of TSEUIF.

results in a severe sparsity problem. It is difficult for traditional recommendation models to solve this problem. However, a sparse matrix may contain a large number of ratings due to the large number of users and items. For example, a sparse matrix of 10,000 rows and 100,000 columns with a density of 1% contains 10,000,000 trainings. In our classification-based recommendation model, numerous user-item pairs are characterized in a relatively low-dimensional feature space, and the corresponding ratings are treated as labels. Obviously, the number of features is much less than that of training samples, so the classification model can be well trained. Hence, our model can well alleviate the sparsity problem.

Thirdly, it is impossible for traditional models to make reliable recommendations due to an initial lack of ratings. This is the so-called cold-start problem, which can further be divided as new user cold-start and new item cold-start. Though a new user has no ratings or a small number of ratings in the target domain, she/he usually provides many ratings in the auxiliary domains which have more ratings than the target domain, as we have assumed. Hence, it is possible to extract her/his features from the auxiliary domains. Likewise, we can also extract the features for a new item. According to the obtained feature vector of a user-item pair, the proposed classification model can predict the corresponding rating. Hence, our model can well solve the cold-start problem.

3.5. Why TSEUIF proposed in this paper can address the case of heterogeneous items

For the case of heterogeneous items, the target domain and the user-side auxiliary domain contain different kinds of items. Since user latent factors or user features inferred from Funk-SVD depend on the kinds of items, not all the user features inferred from the user-side auxiliary domains are appropriate for classification in the target domain. There may be some features shared between the target domain and the user-side auxiliary ones, and acquiring appropriate features from all the constructed ones can be viewed as the problem of feature selection. Since the decision tree algorithm can automatically select significant features during the training, it is assigned to solve the feature selection task. Hence, the model proposed in this paper can address the case of heterogeneous items.

3.6. The computational complexity analysis

Since there are m user- and n item-side auxiliary domains, the proposed model consists of $(m+n)$ Funk-SVD-SGD decomposition and one C4.5 decision tree classification operation.

Let T, p, and f be the maximal number of iterations in one Funk-SVD-SGD decomposition, the number of non-zero entries in the rating matrix, and the number of latent factors, respectively. The worst-case scenario occurs when conducting T iterations. Hence, the computational complexity of one Funk-SVD-SGD decomposition is $O(T \cdot f)$. Specially, given the fact that the latent factors, f_1, $\cdots$, f_{m+n}, correspond to $(m+n)$ auxiliary domains, the total computational complexity is thus $O(T \cdot \sum_{i=1}^{m+n} f_i)$.

For C4.5 decision tree classification, the worst-case scenario occurs when adopting as many attributes as possible before each group of tuples can be classified. For a data set, D, its number of attributes is $t = \sum_{i=1}^{m+n} f_i + 2$, its number of training tuples is $|D| = p$, and the maximal depth of the tree is $\log_2 p$. At each level, we should compute the attribute selection measure $O(\sum_{i=1}^{m+n} f_i + 2)$ times (one per attribute). In addition, the total number of tuples at each level is p. Thus, the computational complexity per level of the tree is $O((\sum_{i=1}^{m+n} f_i + 2) \cdot p)$. Summarizing over all these levels, we can obtain the computational complexity of $O((\sum_{i=1}^{m+n} f_i + 2) \cdot p \cdot \log_2 p)$.

To sum up, the computational complexity of the proposed algorithm is $O(T \cdot \sum_{i=1}^{m+n} f_i + (\sum_{i=1}^{m+n} f_i + 2) \cdot p \cdot \log_2 p)$.

4. Experiments

In this section, we conduct extensive experiments to evaluate the proposed algorithm. We compare the proposed algorithm with seven state-of-the-art algorithms, namely, N-CF-U, Funk-SVD-SGD,

N-CDCF-U, MF-CDCF, CMF, CDTF, and CST, where N-CF-U and Funk-SVD-SGD are two single-domain CF algorithms, and N-CDCF-U, MF-CDCF, CMF, CDTF, and CST are five cross-domain algorithms. All the experiments are run on a PC with 2.20 GHz Intel (R) Core (TM) i5-5200U CPU, 8GB main memory, and window 7. All the algorithms are implemented with Matlab 2015B on top of an open source library for recommender systems: MyMediaLite [29], which implements most common CF approaches including Matrix Factorization.

4.1. Experiments on Amazon dataset

We conduct the first experiment on an Amazon product, co-purchasing network metadata [30], which consists of rating information of users in various domains. Among the product, 99% items belong to the following four main product groups, Book, Music CD, DVD, and VHS video tape. The dataset contains 7,593,243 ratings on the scale of 1–5 provided by 1,555,170 users over these products, including 393,561 books, 103,144 music CDs, 19,828 DVDs, and 26,132 VHS video tapes.

(1) The setting of the compared methods

N-CF-U: A user-based neighborhood CF algorithm. In the experiments, we set $k = 10$.

Funk-SVD-SGD: An algorithm which maps users and items to a joint latent factor space of dimensionality f. The cross-validation method is employed to determine the values of f and λ, where λ is the tradeoff parameter in Funk-SVD-SGD. To be specific, all the original rating data (u, i, r_{ui}) are randomly partitioned into 10 subsets with the same size. Among the 10 subsets, a single subset is retained as the validation data for testing the model, and the remaining 9 subsets are used as the training data. The cross-validation process is then repeated 10 times, with each of the 10 subsets being used exactly once as the validation data. The 10 results from the folds can then be averaged to produce a single estimation. In the experiments, different values of {5, 10, 15, 20} and {0.001, 0.01, 0.1, 1, 10, 100} are tried for f and λ, respectively.

To have a high learning rate and ensure the convergence of an algorithm, we set $\gamma = \frac{1}{\tau}$, where τ is the number of iterations of the training set. In this way, γ takes a larger value at the beginning of the iteration process and a smaller value along with the iteration. The termination rule is presented in the following

$$\Delta J < \delta = 10^{-3} \quad or \quad \tau > T = 1000 \tag{8}$$

where ΔJ is the absolute value of difference of objectives in adjacent iterations, τ means the number of iterations of the training set, and T represents the maximal number of iterations.

N-CDCF-U: A cross-domain version of N-CF-U. In the experiments, we set $k = 10$.

MF-CDCF: A cross-domain version of Funk-SVD-SGD. Here, the parameters, f, λ, and γ, are set in the same way as Funk-SVD-SGD.

CMF: Collective matrix factorization, which couples rating matrices for all the domains on the *User* dimension, so as to transfer knowledge through the common user-factor matrix.

CDTF: Cross-Domain Triadic Factorization, which takes one of the above domains as the target domain to perform prediction, and the others as the auxiliary domains to borrow knowledge. We adopt the same setting as [10]. More specifically, we take the following strategy to initialize individuals with exponential growth, where $a \in (0, 1]$ is a constant to scale the weight, β and γ represent integers to control the range of the weight, and $\mathbf{1}$ means an all-one vector with its length equal to the number of auxiliary domains.

$$\mathbf{w}_i^a = (a \times 10^i) \times \mathbf{1} \qquad i = \beta, \cdots, \gamma \tag{9}$$

In the experiments, we run a GA with its initial population being $\mathbf{w} = \{\mathbf{w^{0.33}}, \mathbf{w^{0.66}}, \mathbf{w^1}\}$, and $\beta = -2$, $\gamma = 2$ to seek an optimal weight assignment, which indicates that there are totally 15 initial individuals with various scales.

CST: The Coordinate System Transfer model. For CST, different latent dimensions {5, 10, 15, 20} are tried, and the tradeoff parameters ρ_u and ρ_v are all set to 5000.

CTSEUIF-C4.5: The proposed method employing C4.5 to solve the classification problem. In the experiment, the parameters f, λ, and γ are set in the same way as Funk-SVD-SGD.

TSEUIF-Relief-F: The proposed method using Relief-F for feature selection, and a naive Bayes classifier for classification. The threshold τ_1 is set to 0.1.

TSEUIF-SFS: The proposed method using SFS for feature selection, and also a naive Bayes classifier for classification. The maximum size of feature subset is set to 30.

(2) Evaluation metrics

We first adopt mean absolute error (MAE) and root mean square error (RMSE) as the evaluation metrics to test the rating prediction performance. Here, MAE is defined as

$$\left(\sum_{i \in T} |r_i - \tilde{r}_i|\right) / |T| \tag{10}$$

and RMSE has the following form

$$\sqrt{\sum_{i \in T} (r_i - \tilde{r}_i)^2 / |T|} \tag{11}$$

where T represents the set of test ratings, r_i is the ground truth, and $\tilde{r}_i$ is the predicted rating. A smaller value of MAE or RMSE means better prediction performance.

To further test the performance in TopN recommendations, we use two other metrics, precision and recall. Let $R(u)$ be a recommended list based on the behavior of the user on the training set, and $T(u)$ be a list of behaviors that the user has on the testing set. Then, the precision of the recommended results is computed by:

$$precision = \frac{\sum_{u \in U} |R(u) \cap T(u)|}{\sum_{u \in U} |R(u)|} \tag{12}$$

The recall of the recommended results is computed by:

$$recall = \frac{\sum_{u \in U} |R(u) \cap T(u)|}{\sum_{u \in U} |T(u)|} \tag{13}$$

(3) Data preparation for MAE and RMSE

We select Book and Music CD as the target domain, respectively. When Book is selected as the target domain, we choose users who have rated at least 30 music CDs, 30 DVDs, and 30 VHS video tapes, so that all the user-side auxiliary domains contain relatively dense rating data. According to this criterion, 496 users are selected. In addition, we retrieved all the items rated by these users in the four domains, and set aside the top h rated items for each domain. Table 1 lists the statistics of the selected data for the case of the Book target domain.

In order to construct the item-side auxiliary domains, we first randomly split the 496 users into four equal parts, P_1, P_2, P_3, and P_4. Then, we select P_1 and the selected items in the Book domain to form the rating matrix, $\mathbf{M}_0$, of the target domain, P_1 and the selected items in Music CD, DVD, and VHS video tape to form the rating matrices, $\mathbf{M}_1$, $\mathbf{M}_2$, and $\mathbf{M}_3$, of the user-side auxiliary domains, respectively. Following that, we select the selected items in Book and P_2, P_3, and P_4 to form the rating matrices, $\mathbf{M}_4$, $\mathbf{M}_5$, and $\mathbf{M}_6$, of the item-side auxiliary domains, respectively. Fig. 5 depicts the location relation of these matrices. In addition, Table 2 lists the statistics of the selected data in each matrix for the case of the Book target domain.

When Music CD is selected as the target domain, we choose users who have rated at least 90 books, 30 DVDs, and 30 VHS video tapes, so that each auxiliary domain contains relatively dense rating data. According to the same criterion, we select 435 users, retrieve all the items rated by these users in these four domains,

Table 1
Statistics of the selected data for the case of the Book target domain.

Domain	User size	h	Avg. # ratings for each item	Avg. # ratings for each user	Density (%)
Books	496	100	29.89	6.03	6.03
Music CDs	496	100	34.65	6.99	6.99
DVDs	496	100	61.77	12.45	12.45
VHS video tapes	496	100	59.43	11.98	11.98

Table 2
Statistics of data in each matrix for the case of the Book target domain.

Matrix	User size	Item size	Avg. # ratings for each item	Avg. # ratings for each user	Density (%)
$\mathbf{M}_0$	124	100	7.8	6.29	6.29
$\mathbf{M}_1$	124	100	7.1	5.73	5.73
$\mathbf{M}_2$	124	100	15.47	12.48	12.48
$\mathbf{M}_3$	124	100	14.91	12.02	12.02
$\mathbf{M}_4$	124	100	8.27	6.67	6.67
$\mathbf{M}_5$	124	100	4.96	4	4
$\mathbf{M}_6$	124	100	8.86	7.15	7.15

Table 3
Statistics of the selected data for the case of the Music CD target domain.

Domain	User size	h	Avg. # ratings for each item	Avg. # ratings for each user	Density (%)
Music CDs	435	100	18.7	4.30	4.30
Books	435	100	50.03	11.5	11.50
DVDs	435	100	62.74	14.42	14.42
VHS video tapes	435	100	59.12	13.59	13.59

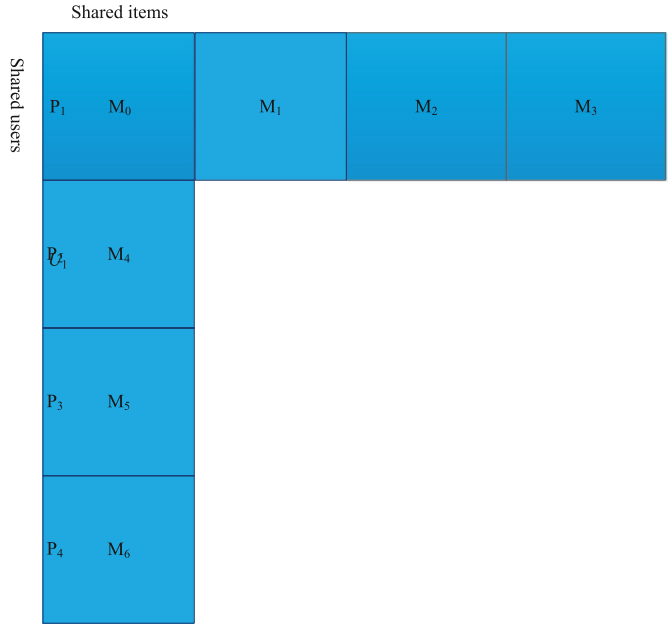


Fig. 5. The location relation of matrices.

and set aside the top h rated items for each domain. Table 3 lists the statistics of the selected data for the case of the Music CD target domain. In addition, we construct the item-side auxiliary domains using the same method. Table 4 lists the statistics of the selected data in each matrix for the case of the Music CD target domain.

To simulate the sparse data problem, we construct five sparse training sets, TR_{50}, TR_{40}, TR_{30}, TR_{20}, and TR_{10}, by holding out 50%, 60%, 70%, 80%, and 90% rating data from $\mathbf{M}_0$, respectively, i.e., the ratio of the remaining data for training is 50%, 40%, 30%, 20%, and 10%, respectively. The hold-out data serve as the ground truth for testing. Likewise, we also construct the other five training sets, TR_{50}, TR_{40}, TR_{30}, TR_{20}, and TR_{10}, when choosing Music CD as the target domain. Since N-CF-U and Funk-SVD-SGD are the single domain CF algorithms, we test their performance on $\mathbf{M}_0$. Since N-CDCF-U, MF-CDCF, and CDTF are the user-side transfer algorithms, and can deal with the case of multiple auxiliary domains, we conduct experiments on $\mathbf{M}_0$, $\mathbf{M}_1$, $\mathbf{M}_2$, and $\mathbf{M}_3$. Since CMF is an item-side transfer algorithm, we test it on $\mathbf{M}_0$, $\mathbf{M}_4$, $\mathbf{M}_5$, and $\mathbf{M}_6$. We conduct an experiment on $\mathbf{M}_0$, $\mathbf{M}_1$, $\mathbf{M}_2$, $\mathbf{M}_3$, $\mathbf{M}_4$, $\mathbf{M}_5$, and $\mathbf{M}_6$ for the proposed algorithm.

(4) Data preparation for precision and recall

We train different algorithms using the same training set constructed for MAE and RMSE previously. However, for simplicity, we only test them on the Book target domain. To compute precision and recall, we need to construct a new testing set in the following way.

We first choose users who have rated more than 10 books from the set composed of hold-out data. Then we select 10 books randomly from each selected user to form the testing set. To compute precision and recall, we map the five classes of the original ratings, {1,2,3,4,5}, into 2 classes, "liked" and "disliked", for the testing set. Generally, an item with a score greater than or equal to 3 is defined as "liked"; otherwise, it is defined as "disliked".

We define the size of the recommendation list, $N = 2, 4, 6,$ and the set of all the books liked by the user from the 10 books as the "liked" list. We sort the predictive ratings of the 10 books for each user in the testing set, and choose the top N books for recommendation. The books in the recommendation list are labeled as "liked". Hence, we can compute the precision and recall.

(5) Experimental results and analysis

Table 5 reports the best values of f, λ for Funk-SVD-SGD model on six user- or item-side auxiliary domains (for the case of the Book target domain). Fig. 6 shows the iteration process on these domains with the maximal number of iterations being 1000. Table 6 reports the comparison results in terms of MAE among TSEUIF-C4.5, TSEUIF-relief-F, and TSEUIF-SFS for the case of the Book target domain. Considering both running time and the recommendation performance, we choose the C4.5 classifier in our TSEUIF model. The values of MAE on Amazon for TSEUIF with and

Table 4
Statistics of data in each matrix for the case of the Music CD target domain.

Matrix	User size	Item size	Avg. # ratings for each item	Avg. # ratings for each user	Density (%)
$\mathbf{M}_0$	108	100	6.04	4.87	4.87
$\mathbf{M}_1$	108	100	11.17	9.01	9.01
$\mathbf{M}_2$	108	100	15.91	12.83	12.83
$\mathbf{M}_3$	108	100	15.33	12.36	12.36
$\mathbf{M}_4$	109	100	3.43	2.52	2.52
$\mathbf{M}_5$	109	100	4.13	3.30	3.30
$\mathbf{M}_6$	109	100	5.45	4.40	4.40

Table 5
The best values of f and λ.

auxiliary domain	f	λ
$\mathbf{M}_1$	10	0.1
$\mathbf{M}_2$	15	0.01
$\mathbf{M}_3$	10	1
$\mathbf{M}_4$	5	0.1
$\mathbf{M}_5$	10	0.01
$\mathbf{M}_6$	15	0.1

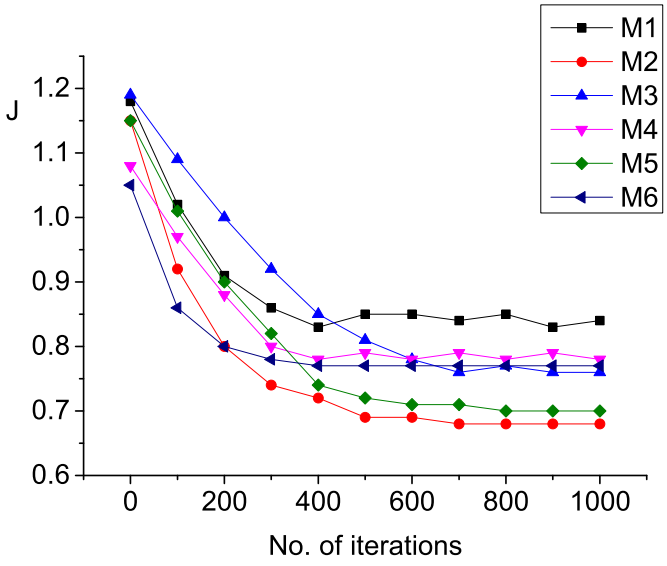


Fig. 6. The iteration process of the stochastic gradient descent algorithm.

without location features are reported in Table 7, which further confirms the conclusion that TSEUIF with location features will perform better than that without location features.

Tables 8 and 9 list the comparison results on MAE and RMSE, respectively. Fig. 7 reports the precision and recall results. The corresponding running time is given in Tables 10 and 11. In addition, Table 12 lists an example of feature importance obtained by C4.5. Since it is inconvenient to list all the feature importance for a large f, we set $f = 5$ for each auxiliary domain, and only provide the results corresponding to the case of the Book target domain in Table 12, where F_{31} represents the user location in the target domain, F_{32} denotes the item location in the target domain, F_1, $\cdots$, F_5 are the five features in $\mathbf{M}_1$, F_6, $\cdots$, F_{10} refer to the five features in $\mathbf{M}_2$, and so on. Note that the results of the proposed algorithm listed in Tables 8 and 9 are obtained with the best values of f and λ in Table 5.

Since different domains are represented with different feature sets in our method, their weights can be computed by adding their feature importance together. According to Table 12, the weights of different domains can be computed easily. Table 13 reports the corresponding results.

We have the following observations from Tables 8 and 9, and Fig. 7. (1) All the six CDCF algorithms, N-CDCF-U, MF-CDCF, CMF, CDTF, CST, and TSEUIF-C4.5, perform better than Funk-SVD-SGD and N-CF-U, because Funk-SVD-SGD and N-CF-U are single-domain CF algorithms which have a difficulty in dealing with the sparsity problem. (2) N-CDCF-U and MF-CDCF achieve almost the same performance. Since they fail to consider the differences among domains, as expected, they perform worse than the other three algorithms, CMF, CDTF, and TSEUIF, which take the differences into account. (3) Though CST is a two-side cross-domain model, it performs even worse than the one-side cross-domain models, i.e., CMF, and CDTF, due to the heterogeneous items between the target and the user-side domains. (4) It is worth noting that N-CDCF-U achieves acceptable performance for dense data, i.e. TR_{50}. Its performance, however, deteriorate rapidly when data become sparse. The reason is that the total similarity used in N-CDCF-U can hardly reflect the local counterpart in the target domain for sparse data.

Table 12 tells that the feature importance can be automatically computed during training C4.5. CMF has no mechanism to seek the optimal weights for the auxiliary domains. Although CDTF assigns the weights based on a genetic algorithm, its performance is susceptible to the initial population. As shown in Table 13, the weight of the target domain becomes smaller with the decrease of the size of the training set. The weight of the target domain possesses the largest value on TR_{50}, which has relatively sufficient training samples. These results demonstrate an obvious truth that the target domain needs only a little information transferred from auxiliary domains (relative small weights on auxiliary domains) if there are sufficient data on it, but it needs to leverage more and more information from auxiliary domains (relative large weights on auxiliary domains) when the data become sparser.

Table 6
The comparison results among TSEUIF-C4.5, TSEUIF-relief-F, and TSEUIF-SFS.

Methods	TR_{50}		TR_{40}		TR_{30}		TR_{20}		TR_{10}	
	MAE	Running time (s)	MAE	Running time (s)	MAE	Running time (s)	MAE	Running time (s)	MAE	Running time (s)
TSEUIF-C4.5	0.552	60.52	0.621	56.03	0.643	53.27	0.729	49.51	0.796	46.52
TSEUIF-relief-F	0.631	57.28	0.663	53.15	0.690	49.42	0.761	46.23	0.825	44.15
TSEUIF-SFS	0.549	1160	0.628	1080	0.649	1050	0.736	1010	0.788	960

Table 7
The values of MAE on Amazon for TSEUIF with and without location features.

Methods	Target Domain: Book					Target Domain: Music CDs				
	TR_{50}	TR_{40}	TR_{30}	TR_{20}	TR_{10}	TR_{50}	TR_{40}	TR_{30}	TR_{20}	TR_{10}
with location features	0.552	0.621	0.643	0.729	0.796	0.536	0.603	0.622	0.706	0.773
without location features	0.602	0.691	0.713	0.809	0.851	0.626	0.643	0.701	0.772	0.826

Table 8
The values of MAE on Amazon.

Methods	Training data set	Target Domain: Book					Target Domain: Music CDs				
		TR_{50}	TR_{40}	TR_{30}	TR_{20}	TR_{10}	TR_{50}	TR_{40}	TR_{30}	TR_{20}	TR_{10}
N-CF-U	$\mathbf{M}_0$	0.709	0.775	0.843	0.892	0.953	0.682	0.749	0.808	0.861	0.921
Funk-SVD-SGD	$\mathbf{M}_0$	0.689	0.768	0.841	0.899	0.957	0.674	0.738	0.812	0.865	0.926
N-CDCF-U	$\mathbf{M}_0$, $\mathbf{M}_1$, $\mathbf{M}_2$, $\mathbf{M}_3$	0.680	0.714	0.767	0.849	0.896	0.649	0.679	0.734	0.819	0.869
MF-CDCF	$\mathbf{M}_0$, $\mathbf{M}_1$, $\mathbf{M}_2$, $\mathbf{M}_3$	0.688	0.723	0.756	0.835	0.892	0.655	0.690	0.728	0.806	0.861
CST	$\mathbf{M}_0$, $\mathbf{M}_1$, $\mathbf{M}_4$	0.686	0.710	0.762	0.853	0.893	0.661	0.695	0.721	0.822	0.863
CMF	$\mathbf{M}_0$, $\mathbf{M}_4$, $\mathbf{M}_5$, $\mathbf{M}_6$	0.676	0.707	0.742	0.821	0.875	0.643	0.667	0.713	0.796	0.852
CDTF	$\mathbf{M}_0$, $\mathbf{M}_1$, $\mathbf{M}_2$, $\mathbf{M}_3$	0.671	0.700	0.729	0.813	0.862	0.640	0.659	0.702	0.781	0.843
CTSEUIF	$\mathbf{M}_0$, $\mathbf{M}_1$, $\mathbf{M}_2$, $\mathbf{M}_3$, $\mathbf{M}_4$, $\mathbf{M}_5$, $\mathbf{M}_6$	0.552	0.621	0.643	0.729	0.796	0.536	0.603	0.622	0.706	0.773

Table 9
The values of RMSE on Amazon.

Methods	Training data set	Target Domain: Book					Target Domain: Music CDs				
		TR_{50}	TR_{40}	TR_{30}	TR_{20}	TR_{10}	TR_{50}	TR_{40}	TR_{30}	TR_{20}	TR_{10}
N-CF-U	$\mathbf{M}_0$	0.894	0.975	1.030	1.103	1.142	0.886	0.941	0.991	1.073	1.110
Funk-SVD-SGD	$\mathbf{M}_0$	0.917	0.961	1.028	1.097	1.149	0.881	0.938	1.004	1.082	1.140
N-CDCF-U	$\mathbf{M}_0$, $\mathbf{M}_1$, $\mathbf{M}_2$, $\mathbf{M}_3$	0.902	0.916	0.995	1.055	1.097	0.871	0.864	0.949	1.018	1.059
MF-CDCF	$\mathbf{M}_0$, $\mathbf{M}_1$, $\mathbf{M}_2$, $\mathbf{M}_3$	0.907	0.925	0.990	1.046	1.089	0.879	0.877	0.943	1.011	1.050
CST	$\mathbf{M}_0$, $\mathbf{M}_1$, $\mathbf{M}_4$	0.903	0.912	0.993	1.060	1.090	0.882	0.880	0.939	1.021	1.052
CMF	$\mathbf{M}_0$, $\mathbf{M}_4$, $\mathbf{M}_5$, $\mathbf{M}_6$	0.897	0.903	0.949	1.005	1.100	0.869	0.849	0.894	0.993	1.029
CDTF	$\mathbf{M}_0$, $\mathbf{M}_1$, $\mathbf{M}_2$, $\mathbf{M}_3$	0.894	0.889	0.916	0.995	1.089	0.859	0.841	0.882	0.976	1.002
CTSEUIF	$\mathbf{M}_0$, $\mathbf{M}_1$, $\mathbf{M}_2$, $\mathbf{M}_3$, $\mathbf{M}_4$, $\mathbf{M}_5$, $\mathbf{M}_6$	0.736	0.814	0.866	0.915	1.001	0.735	0.812	0.810	0.923	0.957

Table 10
The running time (s) for MAE and RMSE.

Methods	Target Domain: Book					Target Domain: Music CDs				
	TR_{50}	TR_{40}	TR_{30}	TR_{20}	TR_{10}	TR_{50}	TR_{40}	TR_{30}	TR_{20}	TR_{10}
N-CF-U	5.12	5.08	5.15	5.22	5.13	4.68	4.75	4.62	4.55	4.61
Funk-SVD_SGD	5.61	4.92	4.32	3.95	3.57	4.92	4.63	4.28	4.05	3.81
N-CDCF-U	10.45	10.13	10.30	10.44	10.12	9.26	9.35	9.41	9.37	9.19
MF-CDCF	10.26	9.69	8.92	7.65	7.32	8.98	8.65	8.22	8.01	7.82
CST	22.53	21.09	20.02	19.52	19.33	20.11	20.26	19.82	19.36	18.92
CMF	20.12	21.68	19.32	19.17	18.63	18.36	18.10	17.28	17.11	16.94
CDTF	151.60	133.92	121.83	100.72	99.28	128.35	116.66	105.27	89.68	82.12
TSEUIF_C4.5	60.52	56.03	53.27	49.51	46.52	47.28	44.12	40.24	38.94	37.26

Table 11
The running time (s) for precision and recall.

Methods	Target Domain: Book				
	TR_{50}	TR_{40}	TR_{30}	TR_{20}	TR_{10}
N-CF-U	6.26	6.17	6.22	6.26	6.20
Funk-SVD_SGD	6.71	6.18	5.62	5.21	4.82
N-CDCF-U	11.92	11.16	11.22	10.76	10.91
MF-CDCF	11.61	10.92	10.35	9.12	8.98
CST	24.12	22.69	21.87	21.93	20.99
CMF	21.89	22.91	20.68	20.71	20.11
CDTF	152.92	135.18	124.26	102.61	100.87
TSEUIF_C4.5	62.27	58.12	55.32	51.22	47.97

4.2. Experiments on Netflix

We further compare the performance of CST and TSEUIF-C4.5 on the case of homogeneous items between the user-side auxiliary domain and the target domain. In the experiments, we construct homogeneous auxiliary and target domain rating data for comparisons. We conduct the experiments on Netflix, which contains more than 10^8 ratings with their values in the range of {1, 2, 3, 4, 5}. The ratings are provided by more than 4.8×10^5 users on around 1.8×10^4 movies.

(1) Data preparation for MAE and RMSE

Since CST requires the user-side auxiliary domain and the target domain have homogeneous items, both the user-side auxiliary domain and the target domain data are selected from Netflix data set. The detailed data set construction method for CST and TSEUIF is provided as follows.

We first randomly extract a $10^4 \times 10^4$ dense rating matrix, $\mathbf{M}$, from Netflix, and take the resulted sub-matrix, $\mathbf{M}_0 = \mathbf{M}_{1:2500,1:2500}$, as the one in the target domain. Then, we take the sub-matrices, $\mathbf{M}_1 = \mathbf{M}_{1:2500,2501:5000}$, $\mathbf{M}_2 = \mathbf{M}_{1:2500,5001:7500}$, and $\mathbf{M}_3 = \mathbf{M}_{1:2500,7501:10000}$, as the user-side auxiliary rating matrices, so that $\mathbf{M}_0$, $\mathbf{M}_1$, $\mathbf{M}_2$, and $\mathbf{M}_3$ only share their common users. Finally, we take the sub-matrices, $\mathbf{M}_4 = \mathbf{M}_{2501:5000,1:2500}$, $\mathbf{M}_5 = \mathbf{M}_{5001:7500,1:2500}$, and $\mathbf{M}_6 = \mathbf{M}_{7501:10000,1:2500}$, as the item-side

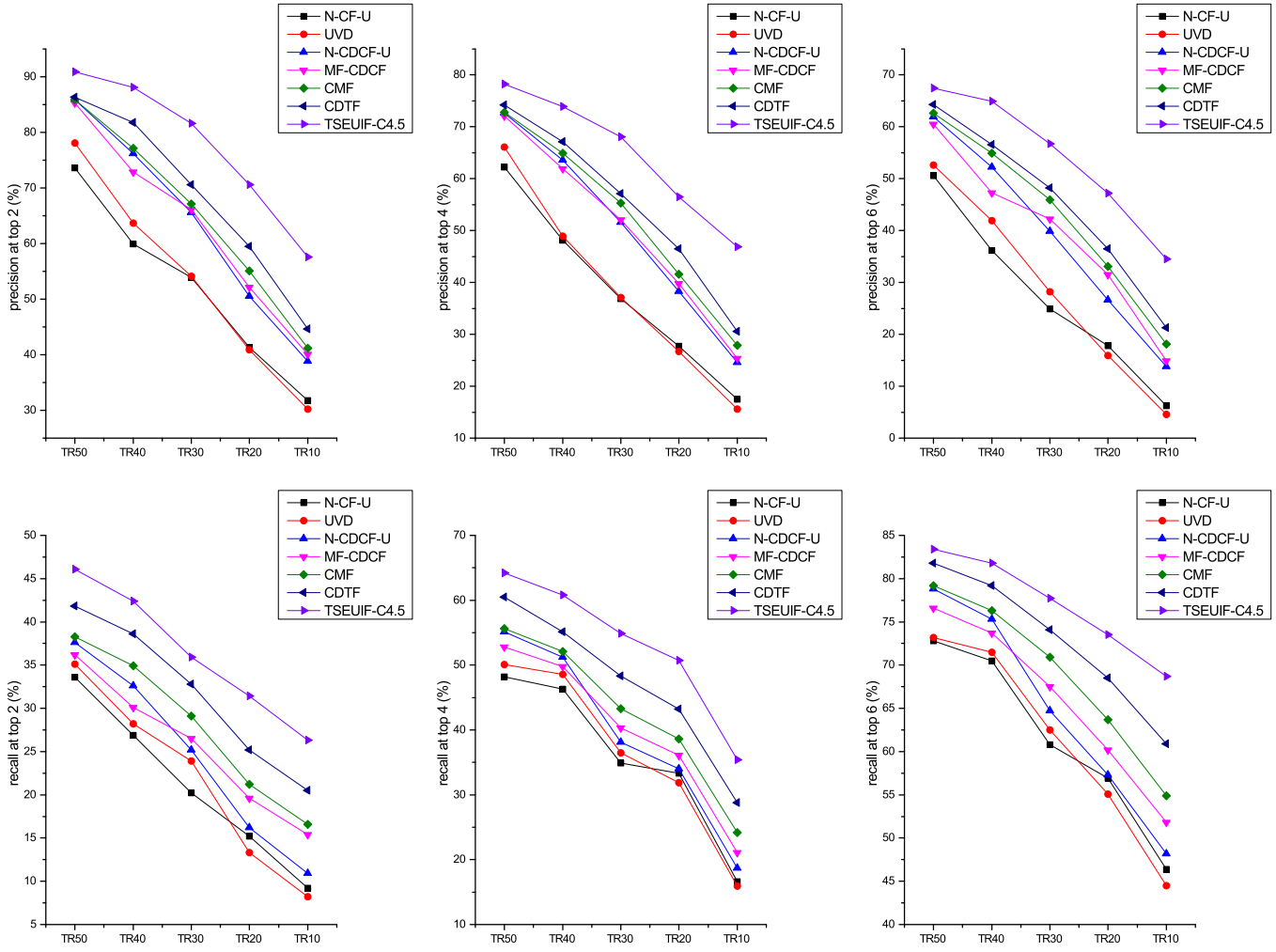


Fig. 7. The precision and recall results on Amazon.

Table 12
The feature importance obtained by C4.5 on the case of the Book target domain.

Feature	Feature importance					Feature	Feature importance				
	TR_{50}	TR_{40}	TR_{30}	TR_{20}	TR_{10}		TR_{50}	TR_{40}	TR_{30}	TR_{20}	TR_{10}
F_1	0.044	0.011	0.003	0.006	0.014	F_{17}	0.030	0.040	0.030	0.019	0.022
F_2	0.012	0.016	0.017	0.000	0.018	F_{18}	0.035	0.050	0.032	0.053	0.073
F_3	0.009	0.000	0.102	0.024	0.013	F_{19}	0.026	0.034	0.022	0.022	0.120
F_4	0.027	0.039	0.002	0.000	0.040	F_{20}	0.040	0.032	0.055	0.043	0.062
F_5	0.005	0.007	0.005	0.002	0.000	F_{21}	0.047	0.024	0.022	0.042	0.012
F_6	0.045	0.018	0.011	0.072	0.007	F_{22}	0.040	0.036	0.044	0.073	0.042
F_7	0.013	0.049	0.012	0.000	0.034	F_{23}	0.029	0.049	0.033	0.065	0.022
F_8	0.002	0.013	0.025	0.007	0.028	F_{24}	0.021	0.039	0.008	0.054	0.035
F_9	0.022	0.003	0.028	0.097	0.007	F_{25}	0.048	0.061	0.054	0.070	0.034
F_{10}	0.004	0.005	0.005	0.011	0.048	F_{26}	0.028	0.038	0.009	0.031	0.022
F_{11}	0.062	0.019	0.027	0.015	0.000	F_{27}	0.030	0.022	0.031	0.021	0.019
F_{12}	0.005	0.033	0.036	0.012	0.013	F_{28}	0.039	0.026	0.030	0.033	0.000
F_{13}	0.021	0.026	0.029	0.037	0.011	F_{29}	0.016	0.042	0.052	0.024	0.073
F_{14}	0.009	0.003	0.021	0.025	0.000	F_{30}	0.018	0.042	0.037	0.039	0.014
F_{15}	0.004	0.088	0.000	0.012	0.113	F_{31}	0.017	0.003	0.065	0.007	0.000
F_{16}	0.039	0.033	0.067	0.012	0.085	F_{32}	0.212	0.099	0.082	0.070	0.018

auxiliary rating matrices. Clearly, $\mathbf{M}_0$, $\mathbf{M}_4$, $\mathbf{M}_5$, and $\mathbf{M}_6$ only share their common items.

Since CST just transfers knowledge from one user-side auxiliary domain and one item-side auxiliary domain, we utilize $\mathbf{M}_0$, $\mathbf{M}_1$, and $\mathbf{M}_4$ for evaluations. Given the fact that the proposed algorithm can transfer knowledge from multiple user- and item-side auxiliary domains, we employ $\mathbf{M}_0$, $\mathbf{M}_1$, $\mathbf{M}_2$, $\mathbf{M}_3$, $\mathbf{M}_4$, $\mathbf{M}_5$, and $\mathbf{M}_6$ for evaluations. In order to form the sparsity problem, the rating set, $\mathbf{M}_0$, is randomly split into the training and the test sets, T_R and T_E, with the ratio of 50% ratings, respectively. $T_R, T_E \subset \{(u, i, r_{ui} \in \mathbb{N} \times \mathbb{N} \times \{1, 2, 3, 4, 5\} \mid 1 \leq u \leq 25,001 \leq i \leq 2500\}$. T_E remains unchanged, whereas different numbers of observed ratings for each

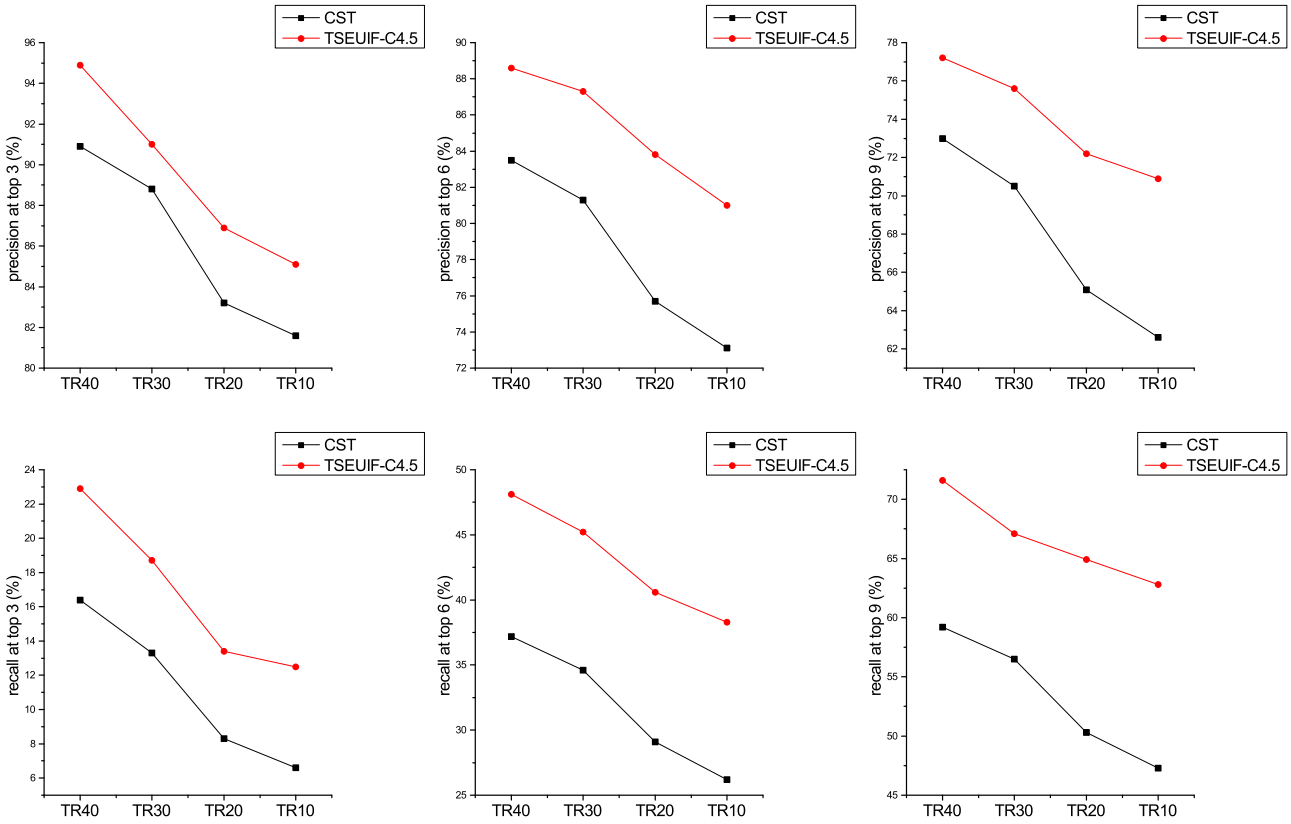


Fig. 8. The precision and recall results on Netflix.

Table 13
Domain weights obtained by C4.5 on the Book target domain.

Domains	Weights				
	TR_{50}	TR_{40}	TR_{30}	TR_{20}	TR_{10}
$\mathbf{M}_0$	0.229	0.102	0.147	0.077	0.018
$\mathbf{M}_1$	0.097	0.073	0.129	0.032	0.085
$\mathbf{M}_2$	0.086	0.088	0.081	0.187	0.124
$\mathbf{M}_3$	0.101	0.169	0.113	0.101	0.137
$\mathbf{M}_4$	0.17	0.189	0.206	0.149	0.362
$\mathbf{M}_5$	0.185	0.209	0.161	0.304	0.145
$\mathbf{M}_6$	0.131	0.17	0.159	0.148	0.128

Table 14
Statistics of data in the target and auxiliary domains for evaluations.

Data set	Avg. # ratings for each user	Sparsity (%)
T_R (target training data)	40, 30, 20, 10	1.6, 1.2, 0.8, 0.4
T_E (target testing data)	271.5	10.86
$\mathbf{M}_1$ (the first user-side auxiliary rating matrix)	280.25	11.21
$\mathbf{M}_2$ (the second user-side auxiliary rating matrix)	308.75	12.35
$\mathbf{M}_3$ (the third user-side auxiliary rating matrix)	267.25	10.69
$\mathbf{M}_4$ (the first item-side auxiliary rating matrix)	304	12.16
$\mathbf{M}_5$ (the second item-side auxiliary rating matrix)	252.75	10.11
$\mathbf{M}_6$ (the third item-side auxiliary rating matrix)	240.75	9.63

user, 10, 20, 30, and 40, are randomly picked from T_R for training, with the sparsity level of 0.4%, 0.8%, 1.2%, and 1.6% correspondingly. Table 14 lists the statistics of data in the experiments.

(2) Data preparation for precision and recall

Following the preparation method on Amazon dataset, we map the five ratings into 2 classes, “liked” and “disliked”. We also train all the models on the previous training set constructed for MAE and RMSE. To construct the testing set for precision and recall, we first choose users who have more than 10 “liked” books from the testing set constructed for MAE and RMSE. Then we select 10 “liked” movies and 20 “disliked” movies to form the testing set for each selected user. Hence, the size of the “liked” list for each user is 10. We define the size of the recommendation list, $N = 3, 6, 9$.

(3) Experimental results and analysis

Tables 15, 16, and Fig. 8 reports the best results of using different parameters as described in the previous subsection. The corresponding running time is given in Tables 17 and 18. We can see that the proposed algorithm outperforms CST even when the user-side auxiliary domain and the target domain have homogeneous items. In the part of related work and the first experiment, we have analyzed and validated that CST cannot effectively deal with the case that the user-side auxiliary domain and the target domain have heterogeneous items. Hence, the proposed algorithm is superior to CST for both cases.

Table 15
The values of MAE on Netflix.

Methods	TR_{40}	TR_{30}	TR_{20}	TR_{10}
CST	0.732	0.749	0.771	0.798
CTSEUIF	0.703	0.711	0.742	0.761

Table 16
The values of RMSE on Netflix.

Methods	TR_{40}	TR_{30}	TR_{20}	TR_{10}
CST	0.926	0.961	0.957	1.019
CTSEUIF	0.910	0.896	0.968	0.986

Table 17
The running time (s) for MAE and RMSE on Netflix.

Methods	TR_{40}	TR_{30}	TR_{20}	TR_{10}
CST	1193.85	1122.63	1082.36	1064.23
TSEUIF_C4.5	2623.12	2411.65	2271.88	2112.19

Table 18
The running time (s) for precision and recall on Netflix.

Methods	TR_{40}	TR_{30}	TR_{20}	TR_{10}
CST	1201.18	1147.32	1132.92	1106.28
TSEUIF_C4.5	2663.81	2433.17	2280.26	2138.72

5. Conclusion and further study

We have proposed a two-side CDCF algorithm with expanding user and item features via the latent factor space of auxiliary domains in this paper from the perspective of classification. In the proposed algorithm, knowledge can be transferred from user- and item-side auxiliary domains to the target domain by expanding the original feature vector. Compared with previous one-side CDCF algorithms, the proposed algorithm can transfer more valuable information to the target domain. In addition, the constructed extra features from auxiliary domains are employed to represent various domains, so calculating the weights of these domains can be converted into the calculation of the weights of the corresponding features. Since C4.5 can automatically select important features during training, the proposed algorithm avoids computing the weights of these domains, which is tough for previous CDCF algorithms. The experimental results have shown that the proposed algorithm significantly outperforms all the state-of-the-art baseline algorithms at various sparsity levels.

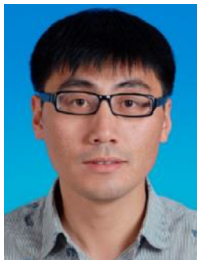
In this study, we just extract domain dependent features, which may have a negative influence on the classification performance. In our future work, we will try to infer domain independent features as well, from auxiliary domain(s), and combine them with domain dependent features to improve the classification performance in the target domain. Besides, we may also study how to apply our model in some real-world applications, such as e-Government and web-page recommendation.

Acknowledgments

This work is jointly sponsored by National Natural Science Foundation of China (Nos. 61402246, 61273180, 61375067, 61773384, 61602133), Natural Science Foundation of Shandong Province (Nos. ZR2019MF014, ZR2018MF007), and key research and development program of Shandong Province (No. 2018GGX101052).

References

[1] J. Bobadilla, F. Ortega, A. Hernando, Recommender systems survey, Knowl.-Based Syst. 46 (1) (2013) 109–132.
[2] D. Goldberg, B.M. Oki, B.M. Oki, D. Terry, Using collaborative filtering to weave an information tapestry, Commun. ACM 35 (12) (1992) 61–70.
[3] P. Bai, Y. Ge, F. Liu, H. Lu, Joint interaction with context operation for collaborative filtering, Pattern Recogn. 88 (2019) 729–738.
[4] D.S. Lee, G.Y. Kim, H.I. Choi, A web-based collaborative filtering system, Pattern Recogn. 36 (2) (2003) 519–526.
[5] H. Liu, Z. Hu, A. Mian, H. Tian, X. Zhu, A new user similarity model to improve the accuracy of collaborative filtering, Knowl.-Based Syst. 56 (3) (2014) 156–166.
[6] Y. Cao, W. Li, D. Zheng, An improved neighborhood-aware unified probabilistic matrix factorization recommendation, Wireless Person. Commun. (4) (2018) 1–20.
[7] A.M. Elkahky, Y. Song, X. He, A multi-view deep learning approach for cross domain user modeling in recommendation systems, in: International Conference on World Wide Web, 2015, pp. 278–288.
[8] P. Cremonesi, A. Tripodi, R. Turrin, Cross-domain recommender systems, in: IEEE International Conference on Data Mining Workshops, 2012, pp. 496–503.
[9] S. Berkovsky, T. Kuflik, F. Ricci, Cross-domain mediation in collaborative filtering, in: 11th International on User Modeling Conference, 2007, pp. 355–359.
[10] L. Hu, J. Cao, G. Xu, L. Cao, Z. Gu, C. Zhu, Personalized recommendation via cross-domain triadic factorization, in: 22nd International Conference on World Wide Web, 2013, pp. 595–606.
[11] X. Yu, F. Jiang, J. Du, D. Gong, A user-based cross domain collaborative filtering algorithm based on a linear decomposition model, IEEE Access 5 (1) (2017) 27582–27589.
[12] A.P. Singh, G.J. Gordon, Relational learning via collective matrix factorization, in: 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2008, pp. 650–658.
[13] W. Pan, N.N. Liu, E.W. Xiang, Q. Yang, Transfer learning to predict missing ratings via heterogeneous user feedbacks, in: 22nd International Joint Conference on Artificial Intelligence, 2011, pp. 2318–2323.
[14] W. Pan, E.W. Xiang, N.N. Liu, Q. Yang, Transfer learning in collaborative filtering for sparsity reduction, in: the 24th AAAI Conference on Artificial Intelligence, 2010, pp. 230–235.
[15] B. Li, Q. Yang, X. Xue, Can movies and books collaborate? cross-domain collaborative filtering for sparsity reduction, in: the International Joint Conference on Artificial Intelligence, 2009, pp. 2052–2057.
[16] B. Li, Q. Yang, X. Xue, Transfer learning for collaborative filtering via a rating–matrix generative model, in: International Conference on Machine Learning, 2009, pp. 617–624.
[17] S. Gao, H. Luo, D. Chen, S. Li, P. Gallinari, Z. Ma, J. Guo, A cross-domain recommendation model for cyber-physical systems, IEEE Trans. Emerg. Topics Comp. 1 (2) (2014) 384–393.
[18] Q. Zhang, D. Wu, J. Lu, F. Liu, G. Zhang, Q. Zhang, D. Wu, J. Lu, F. Liu, G. Zhang, A cross-domain recommender system with consistent information transfer, Decision Support Syst. 104 (2017).
[19] X. Yu, Y. Chu, F. Jiang, Y. Guo, D. Gong, Svms classification based two-side cross domain collaborative filtering by inferring intrinsic user and item features, Knowl.-Based Syst. 141 (2018) 80–91.
[20] Y. Koren, R. Bell, C. Volinsky, Matrix factorization techniques for recommender systems, Computer 42 (8) (2009) 30–37.
[21] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, J. Riedl, Grouplens: an open architecture for collaborative filtering of netnews, in: the ACM Conference on Computer Supported Cooperative Work, 1994, pp. 175–186.
[22] N.D. Buono, T. Politi, A continuous technique for the weighted low-rank approximation problem, in: International Conference on Computational Science and its Applications, 2004, pp. 988–997.
[23] J.R. Quinlan, Induction of decision trees, Mach. Learn. 1 (1) (1986) 81–106.
[24] R. Kohavi, A study of cross-validation and bootstrap for accuracy estimation and model selection, in: the International Joint Conference on Artificial Intelligence, 1995, pp. 1137–1143.
[25] J. Han, Z. Sun, H. Hao, L0-norm based structural sparse least square regression for feature selection, Pattern Recogn. 48 (12) (2015) 3927–3940.
[26] I. Kononenko, Estimating attributes: Analysis and extensions of relief, in: European Conference on Machine Learning, 1994, pp. 171–182.
[27] J. Han, M. Kamber, J. Pei, Data Mining: Concepts and Techniques, Morgan Kaufmann Publishers, San Francisco, California, 2011.
[28] J.R. Quinlan, C4.5: programs for machine learning, Morgan Kaufmann, 1993.
[29] Z. Gantner, S. Rendle, C. Freudenthaler, L. Schmidt-Thieme, Mymedialite:a free recommender system library, in: the 5th ACM Conference on Recommender Systems, 2011, pp. 305–308.
[30] J. Leskovec, L.A. Adamic, B.A. Huberman, The dynamics of viral marketing, ACM Trans. Web 1 (1) (2005) 5.



Xu Yu received the Ph.D. degree in Computer Science and Technology from Harbin Engineering University, China, in 2012. He is currently an associate professor of computer science in the School of Information Science and Technology, Qingdao University of Science and Technology, Qingdao, China. His current research interests include recommender system and transfer learning.

Feng Jiang received the Ph.D. degree in computer software from the Institute of Computing Technology, Chinese Academy of Sciences, China, in 2007. He is currently an associate professor of computer science in the School of Information Science and Technology, Qingdao University of Science and Technology, Qingdao, China. He has been a program committee member of the IEEE International Conference on Granular Computing. His research interests include data mining, rough set theory and machine learning.

Junwei Du received the Ph.D. degree in Computer Software and Theory from TongJi University, ShangHai, China, in 2009. He is a Professor of computer science in the School of Information Science and Technology, Qingdao University of Science and Technology, Qingdao, China. His current research interests include recommender system and intelligent software analysis.

Dunwei Gong received Ph.D. degree in control theory and control engineering from China University of Mining and Technology, Xuzhou, China, in 1999. He is a Professor of China University of Mining and Technology, Xuzhou, China, and is also a Guest Professor of Qingdao University of Science and Technology, Qingdao, China. His current research interests include intelligence optimization and control.