On both Cold-Start and Long-Tail Recommendation with Social Data

Jingjing Li, Ke Lu, Zi Huang and Heng Tao Shen

Abstract—The number of "hits" has been widely regarded as the lifeblood of many web systems, e.g., e-commerce systems, advertising systems and multimedia consumption systems. However, users would not hit an item if they cannot see it, or they are not interested in the item. Recommender system plays a critical role of discovering interesting items from near-infinite inventory and exhibiting them to potential users. Yet, two issues are crippling the recommender systems. One is "how to handle new users", and the other is "how to surprise users". The former is well-known as cold-start recommendation. In this paper, we show that the latter can be investigated as long-tail recommendation. We also exploit the benefits of jointly challenging both cold-start and long-tail recommendation, and propose a novel approach which can simultaneously handle both of them in a unified objective.
For the cold-start problem, we learn from side information, e.g., user attributes, user social relationships, etc. Then, we transfer the learned knowledge to new users. For the long-tail recommendation, we decompose the overall interesting items into two parts: a low-rank part for short-head items and a sparse part for long-tail items. The two parts are independently revealed in the training stage, and transfered into the final recommendation for new users. Furthermore, we effectively formulate the two problems into a unified objective and present an iterative optimization algorithm. A fast extension of the method is proposed to reduce the complexity, and extensive theoretical analysis are provided to proof the bounds of our approach. At last, experiments of social recommendation on various real-world datasets, e.g., images, blogs, videos and musics, verify the superiority of our approach compared with the state-of-the-art work.

Index Terms—Recommender system, cold-start recommendation, long-tail recommendation, transfer learning

1 Introduction

"The success of any kind of social epidemic is heavily dependent on the involvement of people with a particular and rare set of social gifts," writes Malcolm Gladwell in his bestseller *The Tipping Point* [1]. He further named these people as *connectors, mavens* and *salesmen*. The *connectors* know people, the *mavens* accumulate information of commodity, and the *salesman* persuade people to buy things. What's particularly intriguing is that in on-line retail stores, recommender system [2], [3], [4], [5], [6] plays the three roles at the same time. Specifically, a recommender system discovers the interesting items from a near-infinite inventory and exhibits them to potential users. In other words, a recommender system must know people to find out potential users, must accumulate information to pick up interesting items, and must persuade the potential users to buy the recommended items. As a result, the success of on-line retailing is heavily dependent on the recommender system. Actually, Amazon.com has received a 35% of their revenue from recommendations, and 75% of what people watch in Netflix is as a result of recommendations[1].

In the recent literatures reported in the community of recommender system [7], [8], [9], [10], [11], [12], [13], [14], [15], the most popular and effective methods are based on collaborative filtering (CF) [7], [8], [9], [10], [11], [15] and matrix factorization (MF) [12], [13], [14], [16]. CF methods are built on the past interactions between the user and the system. Thus, they generally cannot handle new users since there is no past interaction available. MF methods normally preserve only the principal components after the matrix factorization, so it is almost inevitable that they will lose sight of the niche items. In this paper, we investigate the two challenges and aim to jointly handle both of them in a unified objective.

Handling new users, or items, is well-known as the cold-start problem [17], [18], [19] in recommender systems. The user cold-start problem concerns recommendation for a new user who did not have any historical record in the target system. Cold-start problem has been regarded as a quite challenging problem in the community [18]. With no direct information available, previous work advocate just recommending the most popular items to new users, or resort to learning from side-information to facilitate the cold-start [13], [20], [21], [22]. The side-information can be user attributes (e.g. gender, age and occupation) [19], the content of items (e.g., CNN features of images) [23], [24] or user's social connections (e.g., user's friend groups) [25], [26]. Only recommending the most popular items, on the face of it, is a safe strategy. However, previous work [27], [28] have revealed that recommending popular items do not bring much benefits to either users or content providers. Leveraging side-information gives us an opportunity to learn more about the user and then recommend items based on the user's attributes. For instance, suppose that we found out married young males spend most of their money on beers and diapers, we should also recommend beers and diapers for a new user who is a married young male.

Furthermore, considering a real-world on-line content consumption system, e.g., youtube.com, there would be massive users with each one has only single-digit of available attributes. In other words, hundreds of thousands of users would share the same combination of attributes [29]. The recommender sys-

- *Jingjing Li, Ke Lu and Heng Tao Shen are with the School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu, China, 611731.*
  E-mail: lijin117@yeah.net
- *Zi Huang is with School of ITEE, The University of Queensland, QLD, Australia, 4067.*

1. www.mckinsey.com/industries/retail/our-insights/how-retailers-can-keep-up-with-consumers

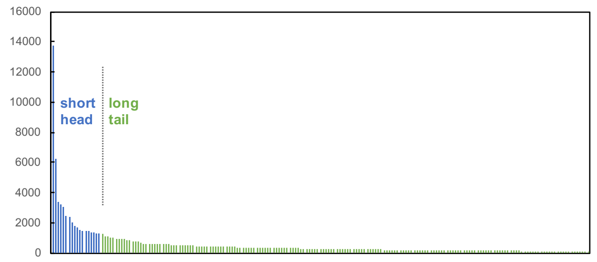


Fig. 1. The popularity of different items in which each item is presented in the Flickr dataset [29]. The horizontal axis denotes the index of items, and the vertical axis indicates the frequency of being interested.

tem, therefore, would treat these users with no difference. As a result, the system can recommend some interesting items to the new user, but there is much of a chance it would lose the *individuality*. In fact, losing the individuality is a pervasive problem in MF methods because previous work [12], [25], [30] generally preserve only principal components [27] after the matrix factorization. Individuality recommendations surprise users, while commonness recommendations end up boring users. Therefore, we should pay close attention to individuality recommendations. Individuality recommendation often involves niche items, which can be exploited by long-tail recommendation [27], [31]. Long-tail recommendation adds novelty and serendipity to the users.

The tipping point of long-tail is Chris Anderson's book *The Long Tail: Why the Future of Business is Selling Less of More* [28], which is the winner of the Gerald Loeb Award for best business book of the year. In the book, Anderson shows how the future of commerce and culture is not in hits, the high-volume head of a traditional demand curve, but in what used to be regarded as misses–the endlessly long tail of that same curve. As a real-world example, we show the long-tail in Fig. 1.

In recommender systems, long-tail recommendation focuses on items. From the perspective of items, the most popular items are well-known by users, they are willing to consume them with or without recommendation. However, a niche market needs more recommendations and it also brings much benefits to both users and content providers [28]. Compared with popular item recommendation, long-tail recommendation has at least three benefits: 1) The market of popular items are highly competitive, thus its profit is very limited. Long-tail items, however, embrace relatively large marginal profit [27]. 2) Long-tail items can further boost the sales for the reason of "one-stop shopping convenience" effect [28]. 3) As we stated before, recommending long-tail items will surprise the users, and increase customer loyalty and satisfaction. An intuitive verification of the effectiveness of the long-tail recommendation can be found in many user generated comments. Xiami[2], for instance, is a popular App of on-line music with a recommender system, and users can give their comments after each song. When one look through these comments, an interesting observation is that the comments after popular songs are usually related to the content of the song itself, e.g., "I really love this song", "I first heard this song when I am in primary school", and "this song was written by David Bowie when he was in Tibet". However, we can find very different comments after niche songs. These comments, for instance, go like "I cannot believe Xiami recommended this for me, this is fantastic" and "no one knows me better than Xiami". The key point of these comments is the recommender system rather than the song itself. That is how a fantastic recommender system works. In recommender systems, "how surprising the recommendations are" can be measured by serendipity [2], which is an important factor beyond accuracy to evaluate a recommender system.

In a nutshell, most of previous work, e.g., CF models [21], matrix factorization models [12] and probabilistic topic models [30], would fail either cold-start recommendation or long-tail recommendation for the reason that those models either depend on past interactions or preserve only principal components (short-head items) and ignore the niche factors (long-tail items). While both cold-start and long-tail recommendation are a matter of cardinal significance. This paper proposes a novel approach to challenge both of them. Technically, our approach is motivated by the following two considerations:

1) Since there is no direct information available for cold-start problem, we learn from side information. Specifically, we leverage the social data of users friend groups, special interest groups and page likes as side information in our experiments. At first, mapping functions which map side information to user-item relationship are learned on the training dataset. Then, we transfer the learned knowledge to new users.
2) Given a system of near-infinite inventory, popular items are only a small fraction of the total items. These items, however, have positive relationship with a majority of users. In a user-item matrix, as illustrated in Fig. 3, the popular items can be revealed by a low-rank function [32]. At the same time, long-tail items are a major portion of the inventory, but they have positive relationship with only a handful of users. Thus, their relationship can be represented by a sparse function. We, therefore, deploy two complementary parts, a low-rank part and a sparse part, in our formulation to handle both the principal components and the niche factors, respectively.

At last, we formulate the cold-start recommendation and long-tail recommendation into a unified objective, and propose an effective optimization strategy. The main ideas of this paper are illustrated in Fig. 2. One can also refer to our previous conference paper [33] for a brief understanding. The main contributions of this paper are listed as follows:

1) To the best of our knowledge, this is the first work in the community which challenges both cold-start and long-tail recommendation in a unified objective. An iterative optimization algorithm is further proposed to solve it.
2) Since scalability is a main challenge in recommender systems [34], we propose a fast extension of the method to reduce the computational complexity.
3) Extensive theoretical analysis, including estimation bound and generalization bound, are presented.
4) Experimental results on four real-world datasets, e.g., images, blogs, videos and musics, verify the superiority of our approach compared with the state-of-the-art work. It is worth noting that our recommendation is beyond accuracy, we also keep an eye on the diversity and serendipity of the recommendations.

The remainder of this paper is organized as follows, section 2 gives a brief review of related work and highlight the difference of our model. Section 3 is the problem definition, formulation and optimization. Experiments are presented in section 4. At last, section 5 is the conclusion.

2. www.xiami.com

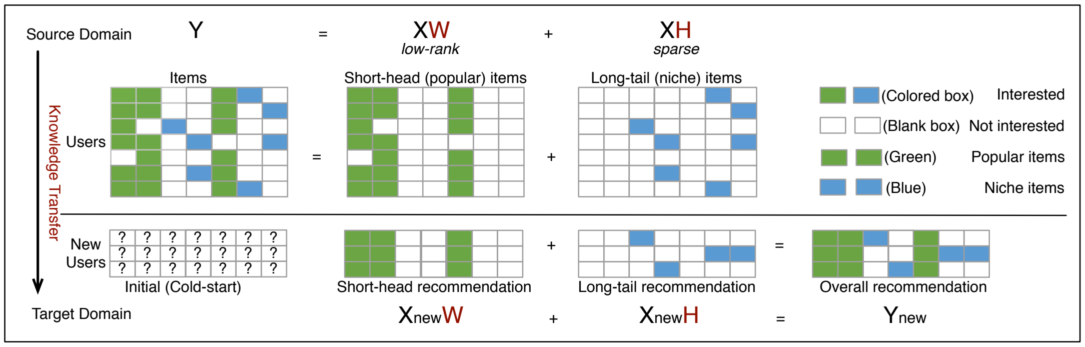


Fig. 2. Illustration of the proposed method. We handle cold-start recommendation by transfer learning, and we decompose the recommendations into two parts, a low-rank part to address short-head items and a sparse part to handle long-tail items.

2 Related Work

2.1 Recommender System

Recommender systems, also known as recommendation systems, basically are information filtering systems which aim to generate a list of recommendations for users [2], [3], [35]. In general, the recommendations are generated by two ways: collaborative filtering (CF) [15], [36], [37], [38] and content-based filtering (CBF) [23], [39]. CF methods place emphasis on users. They collect and analyze a large amount of user behavior and predict the potential items in which users may be interested based on their similarity to other users. CBF methods, on the other side, focus on items. They would recommend items to a user if the description of the items are matched with the user's preferences. Most of today's recommender systems are hybrid recommender systems [40], [41]. Since each recommender technology has its shortcomings, e.g., the well-know cold-start problem and the lack of item description. Hybrid recommender systems combine CF and CBF to strengthen each own strong points and learn from each other to overcome specific limitations.

No matter what recommendation techniques are deployed, it is easy to find that we have to know something about the user if we want to recommend the right item to the right person. User information can be collected by either explicit or implicit forms. The explicit data collection often involves active learning [42], [43]. For instances, asking a user to rate several items, asking a user to select some interesting topics, and asking a user to rank a list of items from favorite to not interested. The challenge of the explicit data collection is that users are not willing to rate the items or answer the questions. Implicit data collection can avoid this problem. There are several ways of implicit data collection, such as observing the items that a user views in a on-line store, analyzing the times of each item viewed by the user, and analyzing the user's social network to discover the user's preferences. In our experiments, we use the social network data to analyze user's interests. Specifically, in our experiments the user information is the "groups" subscribed by users in social network. If a user, for instance, subscribes (joins) the "political novel" group in social network, she would stand a good chance of liking George Orwell (user behavior, e.g., likes webpages about George Orwell and buys *1984*).

Recently, deep neural networks have achieved remarkable results in many communities. More and more researches in recommendation systems are proposed to take the advantage of deep models [15], [24], [44], [45], [46], [47]. For instance, NPR [44] extends the classic Bayesian personalized ranking [48] with multiple contextual preference clues by a deep model. Paul et al. [46] deploy deep neural networks for YouTube recommendations by taking advantage of video features. Compared with conventional methods, deep models are much powerful in feature representation. They are much popular in the fields such as vision, speech, language and games. Since most of existing recommendation systems focus on user-item interactions rather than the content of items. Thus, a majority of deep recommendation systems only deploy neural networks to leverage the content of items as side information. The user-item interactions are still processed by traditional techniques, e.g., matrix factorization. Recently, by replacing the inner product with a neural architecture that can learn an arbitrary function from data, NCF [15] implements collaborative filtering with neural network layers instead of matrix factorization. However, NCF considers neither the cold-start issue nor long-tail recommendation. The performance of NCF heavily depends on the past user-item interactions. It needs the content of items to facilitate cold-start.

Although neural network based recommendation models have demonstrated the state-of-the-art performance, different contents need different neural network feature representations, e.g., CNN for images and RNN for texts. Compared with deep methods which leverage the content of items, methods focusing on user-item interactions, therefore, are more generic. They do not need to change the models to adapt different features. They are robust to different application scenarios, e.g., image recommendation, news recommendation and music recommendation. At the same time, they are also more economical in terms of computation and storage. Therefore, one needs to balance the cost and performance to decide whether deploying deep models in real-world applications.

2.2 Cold-Start Recommendation

Among the models which address cold-start recommendation [18], [49], [50], we focus on the ones which exploit side-information to facilitate the cold-start problem. These models can be roughly grouped into three categories, e.g., the similarity based models [21], [22], [51], matrix factorization models [14], [16], [52] and feature mapping models [19], [53].

The similarity based models are straightforward. Their formulations can be expressed as

$$Y_t = WY_s, \tag{1}$$

where W is the similarity matrix. Generally, W is the similarity between the relationships Y_s and Y_t, the subscripts s and t denote "source" and "target", respectively. However, Y_t is unavailable in cold-start problems, so W represents the similarity between user side-information. W can be calculated by several ways, typically based on the cosine similarity.

Matrix factorization models generally factorize the relationship matrix into two latent representations by optimizing the following objective

$$\min_{U,V} \|Y - UV\|_F^2 + \Omega(U, V), \tag{2}$$

where Ω is the regularization used to avoiding over-fitting. For cold-start problems, one can learn a shared U or V from the side-information, and then use it to predict Y.

Feature mapping models normally learn a feature mapping between the side-information and one of the latent feature representations. The main difference between the matrix factorization models and the feature mapping models is that in matrix factorization models the shared U is jointly learned from Y and the side-information, while in feature mapping models, one needs to learn an additional feature mapping, and further learn different U_s and U_t by sharing the feature mapping. More details can be found in [19], [25].

Apart from the most related work discussed above, other techniques are also explored to address the cold-start problem. The most representative two categories are interview-based methods [16], [54] and deep methods [55], [56], [57]. Interview-based methods generally set up an interview for new users to infer user preference. Deep methods commonly leverage the content of items to facilitate cold-start.

similarity,MF,mapping models,
interview,deep methods

2.3 Long-Tail Recommendation

For the long-tail recommendation problems, Yin et al. [27] propose a novel suite of graph-based algorithms. They represent user-item information with undirected edge-weighted graph and apply hitting time algorithm for long-tail item recommendation. Park and Tuzhilin [31] handle the long-tail recommendation by clustering algorithm. They split the whole itemset into the head and the tail parts and clusters only the tail items. Then recommendations for the tail items are based on the ratings in these clusters and for the head items on the ratings of individual items. Szpektor et al. [58] focus on the long-tail problem in query. They extend the reach of query assistance techniques to long-tail queries by reasoning about rules between query templates rather than individual query transitions. Shi [59] proposes a graph-based recommendation approach which trade-off among multiple criteria of measurements, such as accuracy, similarity, diversity, and long-tail. Domingues et al. [60] study the long-tail recommendation in the application of one-line music recommender systems.

The existing long-tail recommendation approaches, however, still have some problems. Firstly, some of them, e.g., [27] and [60], are overdoing a bit by only recommending long-tail items and ignoring the popular items. Can you imagine a cellphone recommendation list without iPhone on it? I would not risk that. Both popular items and niche items should be considered in an "one-stop shopping" website. Popular items help bring users, and niche items increase customer loyalty. Secondly, none of them considers long-tail recommendation in cold-start scenarios. As we afore stated, long-tail items in cold-start recommendation will surprise the new users and attract them for staying.

2.4 Low-Rank Representation

Low-Rank Representation (LRR) [32] has been proven to be effective for many machine learning problems, such as image classification [61], subspace segmentation [62] and recommender systems [25], [34], [63]. A representative practical of LRR is the Robust PCA [64]. To discover the global subspace structures of data, LRR optimizes the following objective function:

$$\begin{aligned} \min_{Z,E}\ & \operatorname{rank}(Z) + \lambda\|E\|_\ell \\ s.t.\ & X = AZ + E, \end{aligned} \tag{3}$$

where X is the data matrix and A is a dictionary that linearly spans the data space. $\operatorname{rank}(Z)$ is the rank of coefficients matrix Z. $\lambda > 0$ is a balanced parameter and $\|\cdot\|_\ell$ indicates certain regularization strategy, such as the ℓ_1, ℓ_2-norms, used for modeling the noise. By choosing an appropriate dictionary A, LRR can recover the underlying row space so as to reveal the true segmentation of data.

LRR are widely used in computer vision (CV) tasks to decompose the observation into true segmentation and noises. Noises are generally obstructive in CV community. In recommender systems, however, novelty and diversity are key qualities to review the recommendations [65]. Users tend to be more satisfied with recommendations when there is a higher intra-list diversity [66]. However, the number of items sold on major e-commerce retailer is extremely large. The most active users will only have rated a small subset of the overall database. Thus, most items, even the most popular items, do not have enough ratings. Thus, the diversity, as well as the novelty, of recommendations heavily depends on niche items. In this paper, as a result, we not only investigate the low-rank part, but also pay close attention to the sparse part.

3 The Proposed Method

3.1 Notations

In this paper, we use lowercase and uppercase letters to represent vectors and matrices, respectively. A sample is denoted as a vector, e.g., x. Specifically, we use U to denote the user matrix, and I to denote the item matrix. $|U|$ and $|I|$ are the total number of users and items. Let $Y \in \{0,1\}^{|U|*|I|}$ be the relationship matrix between U and I, where $Y_{ui} = 1$ means there is a positive interaction between user u and item i. The positive interaction can be defined in several ways depending on specific application, e.g., for a e-commercial system, the interaction can be purchase, for a video system, the interaction can be watching and for a text system, the interaction can be reading.

3.2 Definitions

Definition 1. Side information $X \in \mathbb{R}^{|U|*|S_s|}$ is defined as users' attributes, preferences or social relationships, e.g., X_{us} can be used to measure how much user u prefers item s in the related, or auxiliary, domain S_s.

Definition 2. Cold-start recommendation addresses the problem of recommendation for a new user u where u has no prior interactions in the target system, i.e., $Y[u,:]$ is unknown.

Definition 3. Long-tail recommendation addresses the problem of recommendation item i for user u where (1) i is related to u and (2) i lies in the long tail.

Problem 1. Given a target system S_t and a related source system S_s, user u is new for S_t, but it has interactions in S_s, recommend top k items, i.e., $i_1, i_2, \cdots, i_k$ for u in S_t, where i_j $(j \in [1,k])$ are close enough to u and some of i_j lie in the long tail.

3.3 Problem Formulation

Most previous recommender systems preserve only principal components (short-head) and ignore the niche factors (long-tail). They would fail to surprise the users. Yin et al. [27] proposed a suite of graph-based algorithms for the long-tail recommendation. However, Yin et al. [27] focus on recommendations of only long-tail items. It will surprise the users, but it is overdoing a bit to make users confused. In this paper, We advocate that the best situation is to recommend items which are a mixture of popular items and niche items. Popular items to make users feel reasonable and niche items to make them feel surprised. In other words, an optimal recommender system should be out of expectation but within understanding. To achieve this, we propose to decompose the relationship matrix Y into two parts, one for short-head, and the other for long-tail. As a result, we have

$$Y = Y_{SH} + Y_{LT} \tag{4}$$

where Y_{SH} is used to formulate the relationship between users and short-head items, while Y_{LT} is used to specify the relationship between users and long-tail items.

Recently, Sedhain et al. [25] found out that various cold-start recommender systems can be boiled down to a linear model represented as

$$Y = XW \tag{5}$$

where W is used to map the user attributes or social relationship to the user-item matrix. For specific models, W can either be directly learned (W is explicit in the objective) or be derived (W is implicit in the objective), e.g., the objective function of [19] can be written as the form of Eq. (5) with $W = (X^\top X)^{-1} X^\top Y$. It is worth noting that although Eq. (5) is a linear assumption, the side information used in our model is users social relationship rather than the content of items, which makes the relation between the user preference and side information easy to be captured, even by a linear model. In our experiments, social data is the "groups" subscribed/joined by users. For instance, if both Joe and Jane are fans of FScott Fitzgerald and they joined the same reading group (social relationship, the side information), and Joe also likes Ernest Hemingway (user interaction of warm users), there is a good chance that Jane is also interested in Hemingway (the recommendation). In fact, users friend groups, special interest groups and page likes, which are the social data used in our paper, directly reflect users preference. Thus, it is safe to assume that users preferences and their social/interest groups can be formulated as a linear regression problem.

As shown in Eq. (5), let $Y_{SH} = XW$ and $Y_{LT} = XH$ respectively, our objective can be formulated as follows

$$\min_{W,H} L(Y, XW, XH) + \Omega(XW, XH), \tag{6}$$

where L is a loss function, and Ω is the regularization. For a real system with millions of users and near-infinite inventory, the popular items are only a tiny fraction (short-head) of total items. To capture this, we propose that XW should be low-rank. On the other hand, The frequency of long-tail items is essentially low. As a result, XH should be sparse. So, our objective can be written as

$$\min_{W,H} \|Y - XW - XH\|_F^2 + \alpha \mathrm{rank}(XW) + \beta \mathrm{sparse}(XH), \tag{7}$$

where $\|\cdot\|_F$ is the Frobenius norm, $\alpha > 0$ and $\beta > 0$ are two balancing parameters. Constraints $\mathrm{rank}(\cdot)$ and $\mathrm{sparse}(\cdot)$ enable a

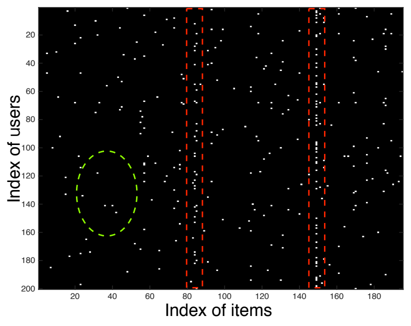
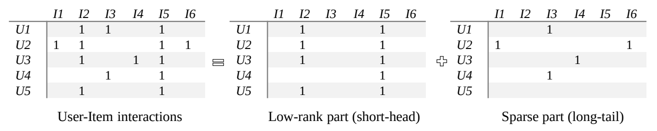


Fig. 3. Illustration of the user/item matrix. White dot means there is a positive relationship between corresponding user and item. Obviously, items in the red boxes are popular items, while others, e.g., the ones marked in the green circle, are long-tail items.

matrix to be low-rank and sparse, respectively. It is worth noting that this formulation is not trivial, it is sound and can be verified. For a better understanding, we randomly choose 200 users from the Flickr dataset [29], and illustrate their preferences in Fig. 3. It is obvious that popular items can be captured by a low-rank [32], [61], [67] matrix, and the long-tail items can be represented by a sparse [68], [69] matrix. As a toy example, we further explain our formulation in Fig. 4.



$$\begin{array}{c|cccccc} & I1 & I2 & I3 & I4 & I5 & I6 \\ \hline U1 & & 1 & 1 & & 1 & \\ U2 & 1 & 1 & & & 1 & 1 \\ U3 & & 1 & & 1 & 1 & \\ U4 & & & 1 & & 1 & \\ U5 & & 1 & & & 1 & \end{array} = \begin{array}{c|cccccc} & I1 & I2 & I3 & I4 & I5 & I6 \\ \hline U1 & & 1 & & & 1 & \\ U2 & & 1 & & & 1 & \\ U3 & & 1 & & & 1 & \\ U4 & & & & & 1 & \\ U5 & & 1 & & & 1 & \end{array} + \begin{array}{c|cccccc} & I1 & I2 & I3 & I4 & I5 & I6 \\ \hline U1 & & & 1 & & & \\ U2 & 1 & & & & & 1 \\ U3 & & & & 1 & & \\ U4 & & & 1 & & & \\ U5 & & & & & & \end{array}$$

User-Item interactions
Low-rank part (short-head)
Sparse part (long-tail)

Fig. 4. A toy example. In the user-item matrix, it is obvious that $I2$ and $I5$ are popular items, $I1$, $I4$ and $I6$ are niche (long-tail) items. Given the fact that long-tail is related with frequency of items, long-tail items naturally correspond to the sparse part of user-item matrix.

Eq. (7) encompasses several approaches with different choices of $\mathrm{rank}(\cdot)$ and $\mathrm{sparse}(\cdot)$. Since rank minimization in Eq. (7) is a NP-hard problem, previous work generally use the trace norm $\|\cdot\|_*$ as a surrogate of $\mathrm{rank}(\cdot)$. As a result, a straightforward approach of Eq. (7) can be written as:

$$\min_{W,H} \|Y - XW - XH\|_F^2 + \alpha \|XW\|_* + \beta \|XH\|_1, \tag{8}$$

where the trace norm $\|XW\|_*$ is used to encourage low-rankness on XW, and $\|XH\|_1$ is used to encourage sparsity on XH. Although $\|\cdot\|_*$ has been widely used as a surrogate of $\mathrm{rank}(\cdot)$ in existing work, $\|\cdot\|_*$ is an implicit form of the low-rank constraint. It controls the single values of XW, but the changes of the single values do not always lead to a change of the rank. Thus, we propose to use an explicit form of low-rank constraint as follows:

$$\min_{W,H} \|Y - XW - XH\|_F^2 + \alpha \sum_{i=r+1}^{d} (\sigma_i(XW))^2 + \beta \|XH\|_1, \tag{9}$$

where $\sigma_i(XW)$ is the $i-$th singular value of XW, d is the total number of singular values of XW. $\sum_{i=r+1}^{d} (\sigma_i(XW))^2$ explicitly solves the problem of minimizing the square sum of r-smallest singular value of XW. Note that

$$\textstyle\sum_{i=r+1}^{d} (\sigma_i(XW))^2 = \mathrm{tr}(V^\top (XW)(XW)^\top V), \tag{10}$$

where $\mathrm{tr}()$ is the trace operator of a matrix, and V are the singular vectors which corresponds to the $(d-r)$-smallest singular values of $(XW)(XW)^\top$. Thus, our final objective function can be written as:

$$\min_{W,H} \|Y - XW - XH\|_F^2 + \alpha \mathrm{tr}(V^\top XWW^\top X^\top V) + \beta\|XH\|_1 + \gamma_1\|W\|_F^2 + \gamma_2\|H\|_F^2 \tag{11}$$

where $\gamma_1 > 0$ and $\gamma_2 > 0$ are two penalty parameters. The F-norm on W and H are introduced to avoid over-fitting. In this paper, we set $\gamma_1 = 1$ and $\gamma_2 = 1$ for simplicity.

Now, we can learn a low-rank function W and a sparse function H by solving Eq. (11). Function W captures the commonality of different users sharing the same interests, and function H captures the personality of specific users. Thus, for a new user with its attribute matrix (or social relationship matrix) X_{new}, he will be mapped to the user-item matrix

$$Y_{new} = X_{new}W + X_{new}H. \tag{12}$$

Note that it is a transfer strategy. We learn W and H from Eq. (11), and transfer the learned knowledge to new users in Eq. (12). It is also worth noting that both short-head items and long-tail items are included in Y_{new}. Thus, our approach not only can handle cold-start problem, but also can handle long-tail recommendation.

3.4 Problem Optimization

Since Eq. (11) is not convex over all variables, we resort to an alternative optimization strategy. Specifically, we optimize only one variable at a time and keep the others fixed. Thus, Eq. (11) can be solved by alternatively solving the two subproblems: (1) solving the short-head recommendation W and (2) solving the long-tail recommendation H.

Solving the short-head recommendation W: When H is fixed, we can optimize W via:

$$\min_{W} \|Y-XW-XH\|_F^2+\alpha \mathrm{tr}(V^\top XWW^\top X^\top V)+\gamma_1\|W\|_F^2. \tag{13}$$

Note that the problem involves both V and W. We alternatively update them. At first, by treating V as a constant, we calculate the deviation with respect to W and set it to zero:

$$X^\top(Y - XW - XH) = \alpha X^\top VV^\top XW + \gamma_1 W. \tag{14}$$

Then, W has the following closed solution:

$$W^* = (\alpha X^\top VV^\top X + \gamma_1 I + X^\top X)^{-1}X^\top(Y - XH), \tag{15}$$

where I is the identity matrix with appropriate size. After getting W, we can update V by Eq. (10).

Solving the long-tail recommendation H: When W is fixed, we can optimize H via:

$$\min_{H} \|Y - XW - XH\|_F^2 + \beta\|XH\|_1 + \gamma_2\|H\|_F^2. \tag{16}$$

Since the ℓ_1−norm involves both X and H, and the constraint is not smooth. To handle this, we introduce an auxiliary variable $Z = XH$. Then, Eq. (16) can be rewritten as the following equivalent problem:

$$\min_{H,Z,E} \|Y - XW - Z\|_F^2 + \beta\|Z\|_1 + \gamma_2\|H\|_F^2 + \mu\|XH - Z + E\|_F^2, \tag{17}$$

where $\mu > 0$ is a penalty parameter, and E is the scaled dual variable.

For Z, we optimize it by fixing H and E. Then, Eq. (17) can be reduced as:

$$\min_{Z_j} \|XH_j - Z_j + E\|_F^2 + \tfrac{\beta}{\mu}\|Z_j\|_1. \tag{18}$$

Algorithm 1. *On both cold-start and long-tail recommendation*

```
Input: User-item matrix $Y$, user attributes $X$, parameters $\alpha$, $\beta$, and $\mu$.
Output: Recommended items for new users.
Warm-up:
  Repeat
    1. Optimize the short-head recommendation function $W$:
       $W^* = (\alpha X^\top VV^\top X + \gamma_1 I + X^\top X)^{-1}X^\top(Y - XH)$.
    2. Optimize the long-tail recommendation function $H$:
       $H_j^* = (\sum_{i=1}^{|U|} x_i x_i^\top + \gamma_2 I + \mu X^\top X)^{-1} * \Gamma$,
       where $\Gamma = (\sum_{i=1}^{|U|} x_i^\top(Y_{ij} - x_i W_j) + \mu X^\top(Z_j - E))$.
    3. Update the dual variable $E$:
       $E^* = E + HX - Z$.
  Until Convergence
Cold-start:
  $Y_{new} = X_{new}W + X_{new}H$.
Recommendation:
  Using the logistic regression function to predict the recommendation
  probability of items, and recommend the top-$k$ items.
```

Then, Z can be optimized via soft thresholding operation as:

$$Z_j^* = \mathrm{soft}(XH_j + E, \tfrac{\beta}{\mu}), \tag{19}$$

where

$$\mathrm{soft}(a, b) = \mathrm{sign}(a)\max(|a| - b, 0). \tag{20}$$

For H, by fixing Z and E, we have:

$$\min_{H_j} \sum_{i=1}^{|U|} \|Y_{ij} - x_i W_j - x_i H_j\|_F^2 + \gamma_2\|H\|_F^2 + \mu\|XH_j - Z_j + E\|_F^2. \tag{21}$$

Then, the gradient w.r.t. H_j can be calculated as:

$$\Delta_{H_j}\mathcal{J} = 2(\sum_{i=1}^{|U|} x_i x_i^\top + \gamma_2 I + \mu X^\top X)H_j - 2(\sum_{i=1}^{|U|} x_i^\top(Y_{ij} - x_i W_j) + \mu X^\top(Z_j - E)) \tag{22}$$

By setting Eq. (22) to zero, we have the closed form of H_j as:

$$H_j^* = (\sum_{i=1}^{|U|} x_i x_i^\top + \gamma_2 I + \mu X^\top X)^{-1}(\sum_{i=1}^{|U|} x_i^\top(Y_{ij} - x_i W_j) + \mu X^\top(Z_j - E)) \tag{23}$$

At last, the scaled dual variable E can be updated as:

$$E^* = E + HX - Z. \tag{24}$$

Thus, the optimal H can be achieved by alternatively updating Z, E and H. For clarity, we show the key steps of our algorithm in Algorithm 1.

4 Algorithm Analysis and Extension

4.1 Computational Complexity

Here we analyze the computational complexity of Algorithm 1 by big O notation. Suppose there are n users and m items, the optimization of W costs O(dn^2+d^3+dnm). The optimization of H costs O($dn^2+d^3+d^2m+dnm$). Thus, the overall computation costs is O(n^2).

It is worth noting that the calculation of V involves eigen-decomposition of $XWW^\top X$, which would be time-consuming with large-scale dataset. As a surrogate, we can use the nuclear norm $\|\cdot\|_*$ instead of $\mathrm{rank}(\cdot)$, as shown in Eq. (8). Although it is an implicit constraint of low-rank, we can use it to make a tradeoff between the time complexity and the cost of some effectiveness. Specifically, from previous work [70], [71], we know that

$$\begin{aligned} \|M\|_* &= \min_{U,V:M=UV^\top} \|U\|_F\|V\|_F^2 \\ &= \min_{U,V:M=UV^\top} \tfrac{1}{2}(\|U\|_F^2+\|V\|_F^2) \end{aligned} \tag{25}$$

where if a matrix $M \in \mathbb{R}^{m*n}$, then $U \in \mathbb{R}^{m*r}$ and $V \in \mathbb{R}^{n*r}$. As a result, we can handle U and V instead of M [72]. The time complexity can be significantly reduced for the reason $r \ll \min(m,n)$. Based on the results, we propose a fast version of the algorithm.

4.2 Reduce the Complexity

Here we introduce Eq. (25) to reduce the complexity, and propose a fast extension of the algorithm. It is well-known that

$$\mathrm{rank}(AB) \leq \{\mathrm{rank}(A), \mathrm{rank}(B)\}. \tag{26}$$

As a result, if we encourage $\mathrm{rank}(W) \leq r$, the rank of XW would be automatically bounded. Thus, the new objective function of our algorithm can be written as

$$\begin{aligned} &\min_{H,U,V:W=UV^\top} \|Y-XUV^\top-XH\|_F^2 \\ &+\alpha(\|U\|_F^2+\|V\|_F^2)+\beta\|XH\|_1+\gamma\|H\|_F^2 \end{aligned} \tag{27}$$

Eq. (27) can be solved by alternatively optimizing the following three sub-problems:

Solving U: When H and V are fixed, U can be optimized via:

$$\min_U \|Y-XUV^\top-XH\|_F^2+\alpha\|U\|_F^2. \tag{28}$$

Eq. (28) is equivalent to the following problem [72]:

Since each column of U can be independently updated, we deploy

$$\min_u \sum_{i=1}^{|U|}\sum_{j=1}^{|I|} \|Y_{ij}-u^\top\widehat{X}_{ij}-(XH)_{ij}\|^2+\alpha\|U\|^2. \tag{29}$$

where $u = vec(U)$ and $\widehat{X}_{ij} = vec(x_i v_j^\top)$. By calculating the deviation to u and setting it to zero, we have the closed-form optimal value:

$$u^* = \Big(\sum_{i=1}^{|U|}\sum_{j=1}^{|I|}\widehat{X}_{ij}\widehat{X}_{ij}^\top+\alpha I\Big)^{-1}\Big(\sum_{i=1}^{|U|}\sum_{j=1}^{|I|}(Y_{ij}-(XH)_{ij})\widehat{X}_{ij}\Big). \tag{30}$$

Solving V: When H and U are fixed, V can be optimized via:

$$\min_V \|Y-XUV^\top-XH\|_F^2+\alpha\|V\|_F^2. \tag{31}$$

Given v_j is the $j-$th line of V, we can optimize V column by column as:

$$\min_{v_j} \sum_{i=1}^{|U|} \|Y_{ij}-x_iUv_j-(XH)_{ij}\|^2+\alpha\|v_j\|^2. \tag{32}$$

By calculating the deviation to v_j and set it to zero, we have the closed-form optimal value:

$$v_j{}^* = \Big(\sum_{i=1}^{|U|}(x_iU)^\top(x_iU)+\alpha I\Big)^{-1}\Big(\sum_{i=1}^{|U|}(Y_{ij}-(XH)_{ij})(x_iU)^\top\Big). \tag{33}$$

Solving H: When U and V are fixed, H can be optimized via:

$$\min_H \|Y-XUV^\top-XH\|_F^2+\beta\|XH\|_1+\gamma\|H\|_F^2. \tag{34}$$

Since U and V are fixed, the optimization of Eq. (34) is same with the optimization of Eq. (16). It is obvious that the cold-start and recommendation steps are same with the steps in Algorithm 1. In Algorithm 2, we show the warm-up step of the method proposed in this section.

Algorithm 2. *Fast warm-up of the proposed method*
Input: User-item matrix Y; side information X; α, β, γ and μ.
Output: The optimal value of U, V and H.

```
Repeat
  1. Optimize $U$ by:
     $u^* = \Big(\sum_{i=1}^{|U|}\sum_{j=1}^{|I|}\widehat{X}_{ij}\widehat{X}_{ij}^\top+\alpha I\Big)^{-1} * \Psi$,
     where $\Psi = \Big(\sum_{i=1}^{|U|}\sum_{j=1}^{|I|}(Y_{ij}-(XH)_{ij})\widehat{X}_{ij}\Big)$.
  2. Optimize $V$ by:
     $v_j{}^* = \Big(\sum_{i=1}^{|U|}(x_iU)^\top(x_iU)+\alpha I\Big)^{-1} * \Phi$,
     where $\Phi = \Big(\sum_{i=1}^{|U|}(Y_{ij}-(XH)_{ij})(x_iU)^\top\Big)$.
  3. Optimize $H$ by:
     $H_j^* = (\sum_{i=1}^{|U|}x_ix_i^\top+\gamma_2 I+\mu X^\top X)^{-1} * \Gamma$,
     where $\Gamma = (\sum_{i=1}^{|U|}x_i^\top(Y_{ij}-x_iW_j)+\mu X^\top(Z_j-E))$.
  4. Update the dual variable $E$:
     $E^* = E+HX-Z$.
Until Convergence
```

For extremely large dataset, we can deploy the divide-and-conquer strategy to further boost the algorithm. Specifically, in the divide step, the overall problem can be divided into several sub-problems according to the division of items. The total n items, for instance, can be divide into c parts, each part has n_i $(n = \sum_{i=1}^{c} n_i)$ items. The original problem, accordingly, can be divide into c sub-problems. Then, we solve each sub-problem in parallel. In the conquer step, the learned sub-projections W_i and H_i $(i \in [1,c])$ can be used to assemble W and H by projecting the sub-projections onto the column space of W and H, respectively.

4.3 Estimation Error

The estimation error measures the accuracy bound of our algorithm. Specifically, it shows the maximum error between the optimized result and the ideal result. It helps understand the theoretical performance of the algorithm. For a specific side information matrix X, we can calculate the recommendation matrix Y by Eq. (12). For the sake of estimation error analysis, suppose that W_0 and H_0 are the ground truth functions for short-head and long-tail recommendation, respectively. The ideal recommendation can be shown as:

$$Y = XW_0 + XH_0 + \epsilon \tag{35}$$

where ϵ is introduced to represent data-independent noise.

To estimate W_0, we deploy the empirical risk minimization as follows:

$$\widehat{W} = \arg\min_W \|Y-XH-XW\|_F^2+\alpha\|XW\|_*. \tag{36}$$

Since H is irrelevant with the optimization of W, we use $\widetilde{Y} = Y - XH$ for clarity. As a result, the estimation of W can be rewritten as:

$$\widehat{W} = \arg\min_W tr(W^\top X^\top XW) - 2tr(\widetilde{Y}^\top XW)+\alpha\|XW\|_*. \tag{37}$$

Since $X^\top X$ is positive semidefinite, if we use λ_s to denote the smallest eigenvalue of $X^\top X$, we have $\lambda_s \geq \varepsilon > 0$, where ε is the lower bound of λ_s. Thus, we have the following theorem.

Theorem 1. *The estimation error of W, i.e., $\|\widehat{W}-W_0\|_F$, has upper bound*

$$\tfrac{1}{\varepsilon}(2\|X^\top\widetilde{Y}-X^\top XW_0\|_* - \alpha\|X^{\mathrm{g}}\|_2^{-1}),$$

where X^{g} is a generalized inverse of X.

Proof. For clarity, we introduce $\Theta = \widehat{W} - W_0$. Then, the objective value in Eq. (37) can be rewritten as

$$\begin{aligned}\varpi = tr((W_0+\Theta)^\top X^\top X(W_0+\Theta))-\\ 2tr(\widetilde{Y}^\top X(W_0+\Theta)) + \alpha\|X(W_0+\Theta)\|_*.\end{aligned} \tag{38}$$

It is easy to know that

$$\varpi \le tr(W_0^\top X^\top XW_0) - 2tr(\widetilde{Y}^\top XW_0) + \alpha\|XW_0\|_*. \tag{39}$$

Combining Eq. (38) and Eq. (39), we have

$$\begin{aligned}tr(\Theta^\top X^\top X\Theta) \le\\ 2tr((X^\top\widetilde{Y}-X^\top XW_0)^\top\Theta) - \alpha\|X\Theta\|_*.\end{aligned} \tag{40}$$

Since $\lambda_s \ge \varepsilon$, it is obvious that

$$tr(\Theta^\top X^\top X\Theta) \ge \varepsilon\|\Theta\|_F^2. \tag{41}$$

Given $\|X\Theta\|_* \ge \|X\Theta\|_F$, we further have

$$\begin{aligned}2tr((X^\top\widetilde{Y}-X^\top XW_0)^\top\Theta) - \alpha\|X\Theta\|_* \le\\ 2\|X^\top\widetilde{Y}-X^\top XW_0\|_*\|\Theta\|_F - \alpha\|X\Theta\|_F.\end{aligned} \tag{42}$$

Combining Eq. (40), Eq. (41) and Eq. (42), we have

$$\varepsilon\|\Theta\|_F^2 \le 2\|X^\top\widetilde{Y}-X^\top XW_0\|_*\|\Theta\|_F - \alpha\|X^{\mathrm{g}}\|_2^{-1}\|\Theta\|_F. \tag{43}$$

As a result,

$$\|\Theta\|_F \le \tfrac{1}{\varepsilon}(2\|X^\top\widetilde{Y}-X^\top XW_0\|_* - \alpha\|X^{\mathrm{g}}\|_2^{-1}). \tag{44}$$

which completes the proof. □

4.4 Generalization Error

The generalization error indicates the capacity of adapting the learned model to unseen samples. As we have known, cold-start recommendation challenges the problem of recommending items to new users, or in other words, unseen users. On one hand, we are interested in analyzing the ability of generalizing to unobserved instances. On the other hand, we are also interested in comparing the proposed method with other approaches even when they deploy a totally different paradigm. We, therefore, introduce the Rademacher complexity to measure the generalization ability of our algorithm.

The Rademacher complexity captures the richness of a family of functions by measuring the degree to which a hypothesis set can fit random noise [73]. Given an arbitrary loss function $L : \mathcal{Y}\times\mathcal{Y}\to\mathbb{R}$. To each hypothesis $h : \mathcal{X}\to\mathcal{Y}$, we can associate a function f that maps $(x,y)\in\mathcal{X}\times\mathcal{Y}$ to $L(h(x),y)$ without explicitly describing the specific loss L used. As a result, we can not only analyze the generalization bound of our algorithm by Rademacher complexity, but also compare our method with other approaches regardless of what loss function is been used.

Definition 4. (Empirical Rademacher complexity) *Given a sample set $S=\{x_1,x_2,\cdots,x_n\}\in\mathcal{X}^n$ and a family of functions $\mathcal{F}$ defined on a space $\mathcal{X}$. The empirical Rademacher complexity of $\mathcal{F}$ with respect to samples S is defined as:*

$$\widehat{\mathcal{R}}_S(\mathcal{F}) = \mathbb{E}_{\boldsymbol{\sigma}}\left[\sup_{f\in\mathcal{F}}\frac{1}{n}\sum_{i=1}^{n}\sigma_i f(x_i)\right], \tag{45}$$

where $\boldsymbol{\sigma}=(\sigma_1,\sigma_2,\cdots,\sigma_n)^\top$, σ_i is the random Rademacher variables.

Definition 5. (Rademacher complexity) *Given the empirical Rademacher complexity of $\mathcal{F}$ as $\widehat{\mathcal{R}}_S(\mathcal{F})$, let D denote the distribution according to which samples are drawn. The Rademacher complexity of $\mathcal{F}$ is the expectation of $\widehat{\mathcal{R}}_S(\mathcal{F})$ over all samples of size m drawn according to D:*

$$\mathcal{R}_m(\mathcal{F}) = \mathop{\mathbb{E}}_{S\sim D^m}\left[\widehat{\mathcal{R}}_S(\mathcal{F})\right]. \tag{46}$$

Our model learns a short-head-items projection W and a long-tail-items projection H. Let $\widehat{\mathcal{L}}(W,H)$ denote the empirical risk of (W,H), the empirical risk minimization with respect to (W,H) can be shown as:

$$\inf_{\substack{\mathrm{rank}(XW)\le r\\ \mathrm{card}(XH)\le s}} \widehat{\mathcal{L}}(W,H) = \frac{1}{|U|}\sum_{i=1}^{|U|}\ell(y_i, f(x_i,W,H)). \tag{47}$$

As a result, the Rademacher complexity of our algorithm can be written as:

$$\mathcal{R}_{|U|}(\mathcal{F}) = \frac{1}{|U|}\mathop{\mathbb{E}}_{x,\boldsymbol{\sigma}}\left[\sup_{(W,H)\in\mathcal{F}}\sum_{i=1}^{|U|}\sigma_i\sum_{j=1}^{|I|}x_i(w_j+h_j)\right], \tag{48}$$

where the subscript j denotes the j-th column of the corresponding matrix. For clarity, Eq. (48) can be rewritten in the form of matrix:

$$\mathcal{R}_{|U|}(\mathcal{F}) = \frac{1}{|U|}\mathop{\mathbb{E}}_{x,\boldsymbol{\sigma}}\left[\sup_{(W,H)\in\mathcal{F}} tr((W+H)^\top\widehat{X})\right], \tag{49}$$

where $\widehat{X}$ denotes the samples weighted by random Rademacher variables, each element in $\widehat{X}$ can be calculated by $\sum_{i=1}^{|U|}\sigma_i x_i$.

Theorem 2. *Given $\mathrm{rank}(XW)\le r$, $\|XH\|_1\le\omega$, the Rademacher complexity of $\mathcal{F}$ has upper bound:*

$$(r+|I|^{3/2}\omega)/|U|.$$

Proof. Eq. (49) can be further written as:

$$\begin{aligned}\mathcal{R}_{|U|}(\mathcal{F}) &= \frac{1}{|U|}\mathop{\mathbb{E}}_{x,\boldsymbol{\sigma}}\sup_{(W,H)\in\mathcal{F}}\left[tr((W+H)^\top\widehat{X})\right]\\ &= \frac{1}{|U|}\mathop{\mathbb{E}}_{x,\boldsymbol{\sigma}}\sup_{(W,H)\in\mathcal{F}}\left[tr(W^\top\widehat{X})+tr(H^\top\widehat{X})\right]\\ &\le \frac{1}{|U|}\mathop{\mathbb{E}}_{x,\boldsymbol{\sigma}}\sup_{(W,H)\in\mathcal{F}}\left[\|\widehat{X}W\|_*)+\|\widehat{X}H\|_*\right]\\ &\le \frac{1}{|U|}\mathop{\mathbb{E}}_{x,\boldsymbol{\sigma}}\sup_{(W,H)\in\mathcal{F}}\left[r+\sqrt{|I|}\|\widehat{X}H\|_1\right].\end{aligned} \tag{50}$$

With the inequation

$$\mathop{\mathbb{E}}_{x,\boldsymbol{\sigma}}\|\widehat{X}H\|_1 \le \mathbb{E}_x|I|\|XH\|_1 \le |I|\omega \tag{51}$$

We have

$$\mathcal{R}_{|U|}(\mathcal{F}) \le (r+|I|^{3/2}\omega)/|U|. \tag{52}$$

□

5 Experiments

5.1 Data Description

- **Flickr** [29] is a dataset collected from *flickr.com*[3], which is a popular photos managing and sharing website. Users in flickr can tag photos and subscribe photos in terms of tags with which he is interested. For instance, a user

3. http://www.flickr.com

TABLE 1
Statistics of the tested dataset in CSR experiments.

Dataset	Relationship entity	Dimensionality	Number of observations	Average observations	Sparsity
Flickr	user-attribute	80,513 x 195	107,741	1.3	99.31%
	user-user	80,513 x 80,513	11,799,764	146.6	99.82%
BlogCatalog	user-attribute	10,312 x 39	14,476	1.4	96.40%
	user-user	10,312 x 10,312	667,966	64.7	99.37%
YouTube	user-attribute	1,138,499 x 47	50,691	0.04	99.60%
	user-user	1,138,499 x 1,138,499	5,980,886	5.3	99.99%
Hetrec11-LastFM	user-artists	1,892 x 17,632	92,834	49.06	99.72%
	user-user	1,892 x 1,892	12,717	6.72	99.64%

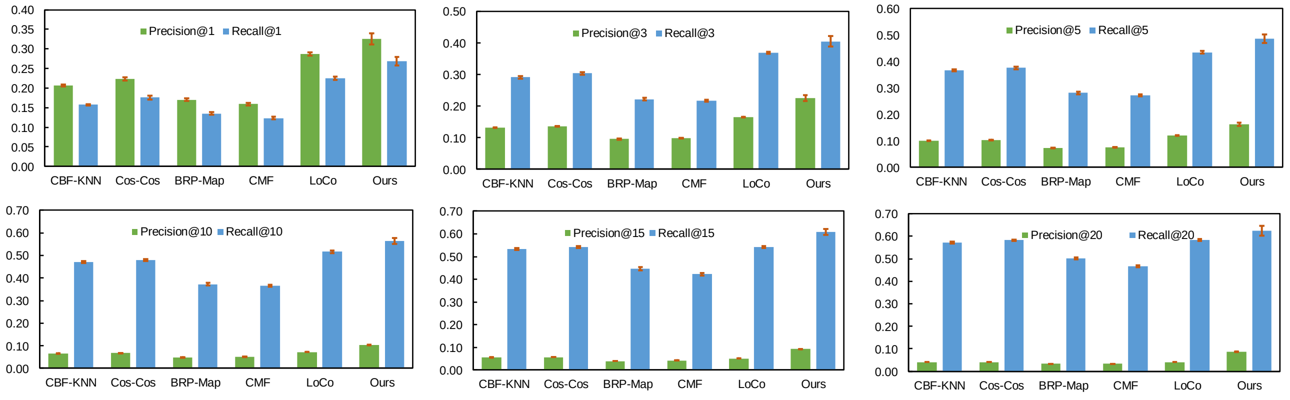


Fig. 5. Experimental results on Flickr dataset. The vertical axis denotes accuracy rate.

can subscribe photos with tag "baseball". The evaluated dataset consists of 80,513 users, 195 interest groups as the items, and a social network with 5,899,882 links.

- **BlogCatalog** [29] is a dataset collected from *blogcatalog.com*[4], which is a popular blog collaboration system. Any article published by a blogger in blogcatalog can be cataloged into some groups according to the topics, e.g., "sports", "business" and "technology". The tested dataset consists of 10,312 users, 39 topics as items, and a social network with 333,983 links.
- **YouTube** [29] is a dataset collected from *youtube.com*[5], which is a popular video watching and sharing website. Users in YouTube can also subscribe interesting topics. The evaluated dataset consists of 1,138,499 users, 47 categories as items and a network with 2,990,443 links.
- **Hetrec11-LastFM** [74] is collected from *last.fm*[6], which is an online music system. Hetrec11-LastFM contains social networking, tagging, and music artist listening information. The tested dataset consists of 1,892 users, 17,632 artists as items, and 186,479 tag assignments.

For clarity, we report the statistics of datasets in Table 1. As we stated in Section 2.1, personal information can be collected by either explicit or implicit forms. In our experiments, the user information is collected by analyzing user's social network data. It is necessary to explain why user's social information can be connected with user's behavior. We know that users' tastes or preferences are generally reflected by their behaviors, e.g., what does she read, listen and buy. User behavior (e.g., what does she buy) is connected with user information (e.g., age, gender and tastes). In our experiments, user information is extracted from user social data. Specifically, user information is the "groups" subscribed by users. If a user, for instance, subscribes the "folk song" group (user information), she would have a great chance of liking Joan Baez (user behavior, e.g., likes Joan Baezs webpage and buys her albums). In reality, we often say "friends make us who we are". The "friends" is our social information, and "who we are" is deeply connected with our behavior.

4. http://www.blogcatalog.com
5. http://www.youtube.com
6. http://www.last.fm

5.2 Compared Methods

We compare our method with six previous work listed as follows:

- **CBF-KNN** [19] is a straightforward recommender system based on **user similarity**. The user similarity is calculated from the low-dimensional projection of user-attributes.
- **Cos-Cos** [21] is a neighborhood-based methods for cold-start **collaborative filtering** in a generalized matrix algebra framework.
- **BPR-Map** [19] is a method that maps entity (e.g. user or item) attributes to the latent features of a matrix (or higher dimensional) factorization model. It is a **matrix factorization** optimized for **Bayesian personalized ranking** that can deal with the cold-start problem.
- **CMF** [14] is a **multi-relational matrix factorization** framework using Bayesian personalized ranking. It extends the Bayesian personalized ranking framework to the multi-relational case.
- **LoCo** [25] is a **low-rank linear regression** method for cold-start recommendation.
- **NCF** [15] is a collaborative filtering method based on **deep neural networks** for recommendation.

It is worth noting that the compared methods consist of very straightforward ones, collaborative filtering ones, matrix factorizations ones, multi-relational matrix factorization ones and deep models. The compared methods deploy various techniques for cold-start recommendation, so that the comparison would comprehensively verify the advantages of our method. Specifically,

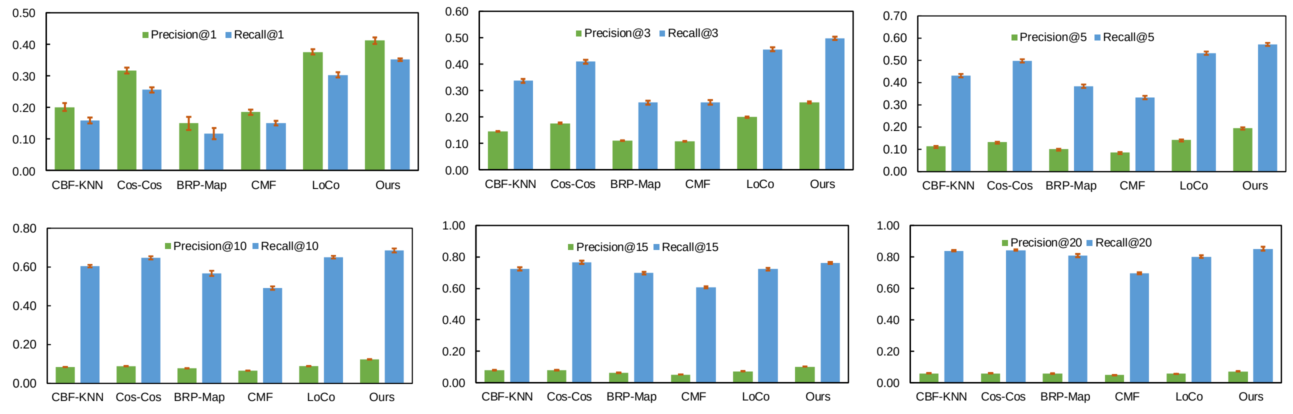


Fig. 6. Experimental results on BlogCatalog dataset. The vertical axis denotes accuracy rate.

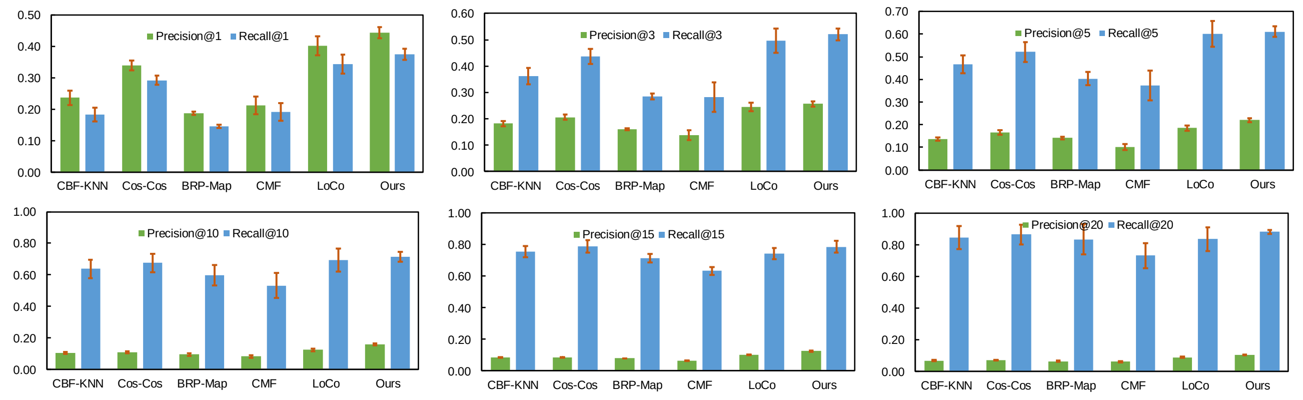


Fig. 7. Experimental results on YouTube dataset. The vertical axis denotes accuracy rate.

TABLE 2
Results of mAP@100 on different dataset.

	Flickr	BlogCatalog	YouTube	LastFM
CBF-KNN	0.2805	0.3271	0.3421	0.1712
Cos-Cos	0.3142	0.4106	0.4667	0.1226
BPR-Map	0.2159	0.2822	0.3035	0.0885
CMF	0.2141	0.2776	0.2816	0.0813
LoCo	0.3357	0.4535	0.4879	0.1809
Ours	**0.3711**	**0.4962**	**0.5306**	**0.2113**

LoCo deploys low-rank constraint on the weight matrix, which is similar with our formulation of learning short-head items. By comparing with LoCo, we can see that introducing long-tail items into cold-start recommendation is a rewarding serendipity.

5.3 Experimental Protocols

For the evaluated datasets, we split each of them into two subsets, one includes 10% of the users as new users (test dataset) for cold-start, and the remainder of 90% users are collected as training data for warm-up. The new users are randomly selected, so we build 10 training-test folds and report the average results.

Following previous work [25], we deploy the widely used precision@k, recall@k and mean average precision (mAP@100) as the measurements. All the hyper-parameters in the objective are tuned by cross-validation. One can also directly set $\lambda = 1$, $\beta = 0.1$ and $\mu = 1$ for simplicity.

5.4 Experimental Results and Discussions

Fig. 5 shows the experimental results on Flickr dataset. We report both precision and recall (@1, @3, @5, @10, @15 and @20). It can be seen that our approach always performs the best with respect to either precision or recall. The neighbor relationship based methods, e.g., CBP-KNN and Cos-Cos, generally perform better than the learning based approaches, e.g., BPR-Map and CMF. A possible explanation is that using social information is effective when we know nothing about the user's preference. Users have similar attributes tend to have similar tastes. Although the learning based approaches usually perform better on regular recommendation, the compared methods are not sophisticated enough in cold-start recommendation.

Our approach deploys the ideas of transfer learning [75]. We first learn from the side information, and then transfer the learned knowledge to the new users. The user relationship are embedded in the learned knowledge. Thus, our approach takes the advantage of neighbor relationship based approaches. On the other hand, the compared baselines commonly preserve the principal components (short-head items) and ignore the long-tail items, which can degenerate the personal recommendation. Our approach independently handles the short-head items and long-tail items, and combines them for overall recommendation.

The same observations can also be drawn from the experimental results on BolgCatalog dataset reported in Fig. 6. The precision is relatively high when k is small, while the recall is relatively high when k is large. The latter is easy to be explained for the reason that more relevant items will be included with larger k. The former is, however, not straightforward. It is relevant with the ground truth. Since the relationship matrices of both Flickr and BlogCatalog are very sparse, e.g., the sparsity of Flickr is 99.31%, and the average observations per user is around 1.4, the numerator of the precision@k tends to be fixed with the increase of the denominator.

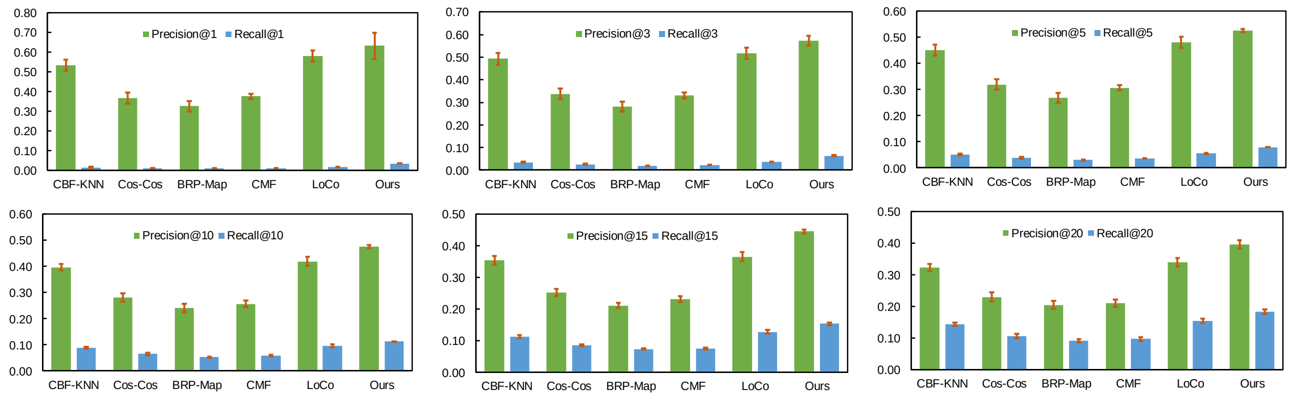


Fig. 8. Experimental results on Hetrec11-LastFM dataset. The vertical axis denotes accuracy rate.

TABLE 3
Comparing with deep model NCF on three datasets.

Settings		Precision						Recall					
		@1	@3	@5	@10	@15	@20	@1	@3	@5	@10	@15	@20
Flickr	NCF	0.1519	0.0800	0.0621	0.0392	0.0336	0.0274	0.1495	0.2285	0.2978	0.3769	0.4845	0.5277
	Ours	0.3256	0.2256	0.1623	0.1026	0.0921	0.0875	0.2687	0.4053	0.4862	0.5633	0.6075	0.6243
BlogCatalog	NCF	0.1581	0.1117	0.0975	0.0649	0.0525	0.0452	0.1266	0.2930	0.4224	0.5599	0.6745	0.7744
	Ours	0.4122	0.2563	0.1955	0.1236	0.1011	0.0712	0.3526	0.4986	0.5721	0.6858	0.7619	0.8521
YouTube	NCF	0.3409	0.1854	0.1381	0.0938	0.0684	0.0526	0.3149	0.5006	0.6162	0.8356	0.8933	0.9044
	Ours	0.4431	0.2563	0.2201	0.1557	0.1253	0.1026	0.3738	0.5201	0.6113	0.7156	0.7861	0.8835

Furthermore, we show the experimental results on two additional multimedia systems, e.g., YouTube and LastFM, in Fig. 7 and Fig. 8, respectively. Fig. 7 shows the similar observations with Fig. 6. However, the performance on YouTube is generally better than on BlogCatalog. A possible explanation is that YouTube dataset has more information than BlogCatalog dataset. Both the average observations per user and per label in YouTube dataset are larger than in BlogCatalog dataset. In other words, the side information is richer in YouTube dataset.

Fig. 8 shows different patterns with the former ones. Experimental results reported in Fig. 8 commonly have higher precision than recall. This is related with the ground truth of the dataset. Different from the other three datasets, the interesting items of each user are much more in Hetrec11-LastFM. As a result, the recall@k is small with a small k and a big relevant number.

All in all, experiments on different real-world datasets verify the superiority of our algorithm, no matter which measurement is used. Our formulation, therefore, is effective for cold-start recommendation. In addition, we report the mAP@100 on different datasets in Table 2. It can be seen that our approach achieves a significant improvements against the compared methods, which further verifies the effectiveness of our formulation.

5.5 Comparing with Deep Method

In the past few years, deep learning has swept the machine intelligence community. Many of state-of-the-art results in recommender systems are also achieved by deep models. For instance, neural personalized ranking (NPR) [44] implements and extends the Bayesian personalized ranking with neural networks and achieves state-of-the-art result. However, although several work have explored taking advantage of deep learning for recommendation, they mainly use the deep model to handle auxiliary information, e.g., CNN features of images and RNN features of texts. In our method, we learn the recommendations by user-item interactions, which are still resorted to matrix factorization in most of previous deep models. As a result, it is not necessary to specifically compare our method with them. Recently, neural collaborative

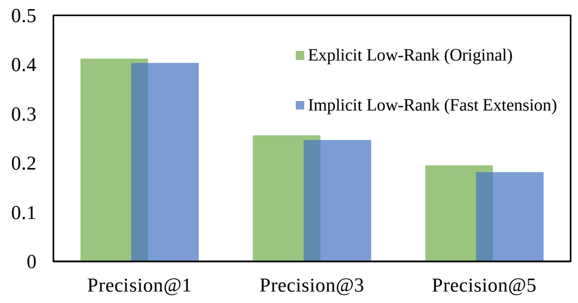


Fig. 9. The precision@k on BlogCatalog dataset.

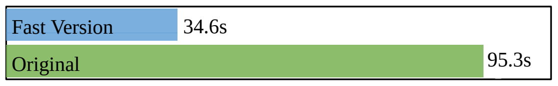


Fig. 10. The running time of original version and fast extension on Flickr dataset.

filtering (NCF) [15] deploys a multi-layer perceptron to learn the useritem interaction function instead of only exploring auxiliary information, which has attracted much attention in the community. Thus, we take NCF as a representative deep model and compare it our method. The results are reported in Table 3.

It is worth noting that NCF itself is not applicable for user cold-start. In order to make NCF work, the reported results of NCF in Table 3 are achieved according to the protocols of NCF. Therefore, the results of NCF are not strictly cold-start results. We randomly select 90% user-item interactions, instead of users, for training and the remainder 10% for test. With respect to the test users, our setting is purely cold-start, while NCF is partially cold-start.

It can be seen that our method performs much better than NCF on the cold-start recommendation. Although NCF is a deep model, its core is still collaborative filtering (CF). The recommendations of CF methods, however, heavily rely on previous user-item interactions. They are vulnerable to cold-start recommendation. To address this issue, CF methods generally leverage the content

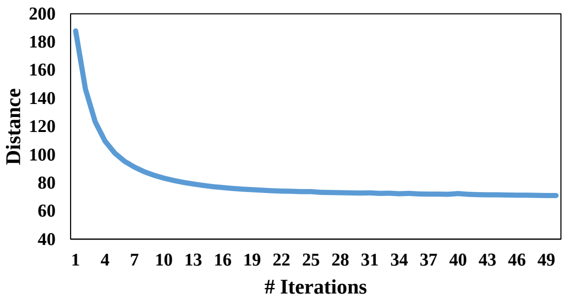


Fig. 11. The values of $\|Y - XW - XH\|_F^2$ with different iterations.

of recommendations, e.g., convolutional features for image recommendation and acoustic features for music recommendation. On the other hand, NCF does not consider the long-tail recommendation, which has been proven to be important in previous work [27].

5.6 Tradeoff between Speed and Performance

In section 4.2, we presented an fast extension of the proposed method. We also reported that the trace norm is an implicit surrogate of the rank norm, so that the fast extension may perform not as well as the explicit low-rank optimization. In this section, we discuss the tradeoff between speed and performance with corresponding experimental results. Fig. 9 shows the precision@k of the two methods on BlogCatalog dataset. It can be seen that the performance of the fast version is slightly worse than the original one. At the same time, Fig. 10 reports the running time of the two methods on Flickr dataset. We can observe that the fast extension does speed up the training process, which makes the proposed method being practical for large-scale datasets.

5.7 Assumption Verification

In our experiments, we leverage the social data, specifically, the groups subscribed/joined by users, as the side information to address cold-start problem. Since users friend groups, special interest groups and page likes directly reflect users preference, we formulate our model as a liner regression problem. To verify the formulation, we report the distance between the true preferences and learned ones. Specifically, we report the value of $\|Y - XW - XH\|_F^2$ with different iterations in Fig. 11. From the result, we can see that the learned W and H can be fitted with Y in several iterations and achieve a stable result.

5.8 Parameters Sensitivity

To show the parameters sensitivity of our model, we report the experimental results with different values of λ, β and μ in Fig. 12(a), Fig. 12(b) and Fig. 12(c), respectively. Each of the parameters are tested from a wide range from 0 to 100.

It can be seen that our approach is not sensitive to β and μ, but the value of α should not be too small. In practice, we suggest setting α from 1 to 10, and finding the optimal value by cross-validation.

5.9 Effectiveness of Long-tail Recommendation

Since short-head and long-tail are a pair of relative concepts, it is hard to precisely quantify the long-tail recommendation. To evaluate the effectiveness of long-tail recommendation in the cold-start process, we build two series of experiments. In the first series, we set $H = 0$ to verify the contribution of long-tail recommendations w.r.t overall performance. The experimental results show that the overall performance degenerates 2%-5% on different evaluations. In the second series, we study the popularity of the recommended items to verify whether long-tail items are included in the final recommendation. The measurement of popularity is defined as the rating/clicking/liking frequency of items [27] (please note that long-tail is defined based on the frequency of items). We randomly select 1,000 users from BlogCatalog and report the average popularity of recommended items in Fig. 13. It can be seen that our approach generally recommend more niche items (long-tail items) than LoCo [25]. Since our approach considers both popular items and niche items, it tends to recommend the popular item when the number of recommendation is only 1 (If it can recommend only 1 cellphone to users today, surely it would be iPhone). LoCo continues recommending popular items with increasing number of recommendations. Our approach, however, prefers recommending more niche items.

6 Conclusion

Both cold-start recommendation and long-tail recommendation are challenging problems in the community. To the best of our knowledge, this work is the first one which challenges both cold-start recommendation and long-tail recommendation in a unified optimization problem. Extensive experiments on four real-world datasets verify the effectiveness of the proposed method. This paper shows that one can use side information to warm-up the recommender system when there is no available historical recorders. We also find that considering long-tail items in the process of cold-start can be beneficial. In fact, our ideas of independently handling the short-head items and long-tail items can also be used in regular recommendations (relative to cold-start recommendation). It is one of the work we will study in the future.

In our experiments, the side formation is mainly from social data. However, side information can be collected from many other sources, e.g., cross domain platforms, questionnaires by active learning, and each source can reinforce the others. As a result, in our future work, we are going to study the multi-source side information assisted cold-start recommendation.

7 Acknowledgements

We sincerely thank the reviewers for valuable suggestions and constructive comments. This work was supported in part by the National Natural Science Foundation of China under Grant 61806039 and 61832001, in part by the ARC under Grant FT130101530, in part by the National Postdoctoral Program for Innovative Talents under Grant BX201700045, and in part by the China Postdoctoral Science Foundation under Grant 2017M623006.

References

[1] M. Gladwell, *The tipping point*. Boston: Little, Brown, 2000.
[2] F. Ricci, L. Rokach, and B. Shapira, *Introduction to recommender systems handbook*. Springer, 2011.
[3] G. Adomavicius and A. Tuzhilin, "Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions," *IEEE TKDE*, vol. 17, no. 6, pp. 734–749, 2005.
[4] S. M. McNee, J. Riedl, and J. A. Konstan, "Being accurate is not enough: how accuracy metrics have hurt recommender systems," in *CHI'06 extended abstracts on Human factors in computing systems*. ACM, 2006, pp. 1097–1101.

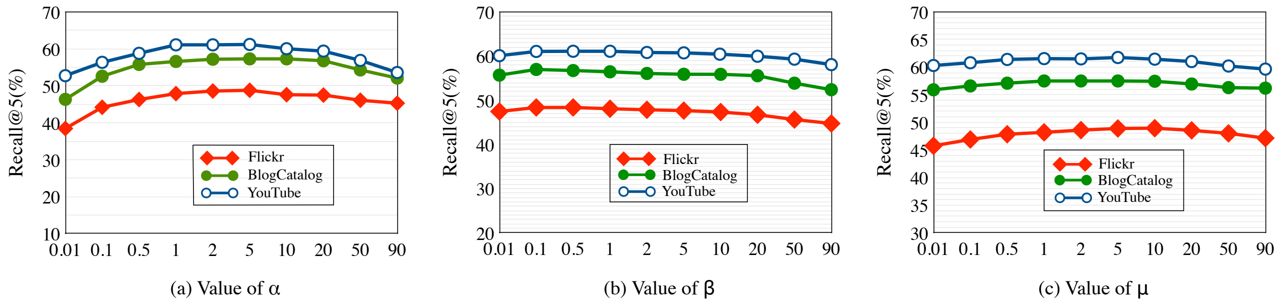


Fig. 12. Parameters sensitivity of the proposed method.

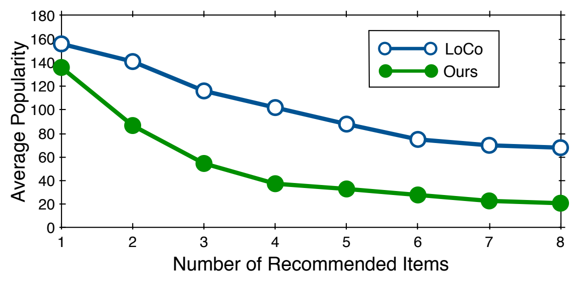


Fig. 13. Average popularity of recommendations.

[5] Z. Cheng and J. Shen, "Just-for-me: An adaptive personalization system for location-aware social music recommendation," in *ICMR*, 2014, p. 185.
[6] C. Zhiyong and S. Jialie, "On effective location-aware music recommendation," *ACM TIST*, vol. 34, no. 2, pp. 13:1–13:32, 2016.
[7] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Item-based collaborative filtering recommendation algorithms," in *WWW*. ACM, 2001, pp. 285–295.
[8] G. Linden, B. Smith, and J. York, "Amazon. com recommendations: Item-to-item collaborative filtering," *IEEE Internet computing*, vol. 7, no. 1, pp. 76–80, 2003.
[9] D. Goldberg, D. Nichols, B. M. Oki, and D. Terry, "Using collaborative filtering to weave an information tapestry," *Communications of the ACM*, vol. 35, no. 12, pp. 61–70, 1992.
[10] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl, "Grouplens: an open architecture for collaborative filtering of netnews," in *CSCW*. ACM, 1994, pp. 175–186.
[11] Y. Hu, X. Yi, and L. S. Davis, "Collaborative fashion recommendation: a functional tensor factorization approach," in *ACM MM*. ACM, 2015, pp. 129–138.
[12] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, no. 8, 2009.
[13] M. Jamali and M. Ester, "A matrix factorization technique with trust propagation for recommendation in social networks," in *ACM RecSys*. ACM, 2010, pp. 135–142.
[14] A. Krohn-Grimberghe, L. Drumond, C. Freudenthaler, and L. Schmidt-Thieme, "Multi-relational matrix factorization using bayesian personalized ranking for social network data," in *ACM WSDM*. ACM, 2012, pp. 173–182.
[15] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, "Neural collaborative filtering," in *WWW*. WWW Steering Committee, 2017, pp. 173–182.
[16] L. Shi, W. X. Zhao, and Y.-D. Shen, "Local representative-based matrix factorization for cold-start recommendation," *ACM TOIS*, vol. 36, no. 2, p. 22, 2017.
[17] X. N. Lam, T. Vu, T. D. Le, and A. D. Duong, "Addressing cold-start problem in recommendation systems," in *ACM IMCOM*. ACM, 2008, pp. 208–211.
[18] A. I. Schein, A. Popescul, L. H. Ungar, and D. M. Pennock, "Methods and metrics for cold-start recommendations," in *ACM SIGIR*. ACM, 2002, pp. 253–260.
[19] Z. Gantner, L. Drumond, C. Freudenthaler, S. Rendle, and L. Schmidt-Thieme, "Learning attribute-to-feature mappings for cold-start recommendations," in *ICDM*. IEEE, 2010, pp. 176–185.
[20] Z.-K. Zhang, C. Liu, Y.-C. Zhang, and T. Zhou, "Solving the cold-start problem in recommender systems with social tags," *EPL*, vol. 92, no. 2, p. 28002, 2010.
[21] S. Sedhain, S. Sanner, D. Braziunas, L. Xie, and J. Christensen, "Social collaborative filtering for cold-start recommendations," in *ACM RecSys*. ACM, 2014, pp. 345–348.
[22] V. A. Rohani, Z. M. Kasirun, S. Kumar, and S. Shamshirband, "An effective recommender algorithm for cold-start problem in academic social networks," *Mathematical Problems in Engineering*, 2014.
[23] A. Van den Oord, S. Dieleman, and B. Schrauwen, "Deep content-based music recommendation," in *NIPS*, 2013, pp. 2643–2651.
[24] S. Zhang, L. Yao, and A. Sun, "Deep learning based recommender system: A survey and new perspectives," *arXiv preprint arXiv:1707.07435*, 2017.
[25] S. Sedhain, A. K. Menon, S. Sanner, L. Xie, and D. BraziunasS, "Low-rank linear cold-start recommendation from social data," in *AAAI*, 2017.
[26] P. Cui, Z. Wang, and Z. Su, "What videos are similar with you?: Learning a common attributed representation for video recommendation," in *ACM MM*. ACM, 2014, pp. 597–606.
[27] H. Yin, B. Cui, J. Li, J. Yao, and C. Chen, "Challenging the long tail recommendation," vol. 5, no. 9. VLDB Endowment, 2012, pp. 896–907.
[28] C. Anderson, *The long tail: Why the future of business is selling less of more*. Hachette Books, 2006.
[29] L. Tang and H. Liu, "Relational learning via latent social dimensions," in *ACM SIGKDD*. ACM, 2009, pp. 817–826.
[30] W.-Y. Chen, J.-C. Chu, J. Luan, H. Bai, Y. Wang, and E. Y. Chang, "Collaborative filtering for orkut communities: discovery of user latent behavior," in *WWW*. ACM, 2009, pp. 681–690.
[31] Y.-J. Park and A. Tuzhilin, "The long tail of recommender systems and how to leverage it," in *ACM RecSys*. ACM, 2008, pp. 11–18.
[32] G. Liu, Z. Lin, S. Yan, J. Sun, Y. Yu, and Y. Ma, "Robust recovery of subspace structures by low-rank representation," *IEEE TPAMI*, vol. 35, no. 1, pp. 171–184, 2013.
[33] J. Li, K. Lu, Z. Huang, and H. T. Shen, "Two birds one stone: On both cold-start and long-tail recommendation," in *ACM MM*, ser. MM '17. ACM, 2017, pp. 898–906.
[34] G. Takács, I. Pilászy, B. Németh, and D. Tikk, "Scalable collaborative filtering approaches for large recommender systems," *JMLR*, vol. 10, no. Mar, pp. 623–656, 2009.
[35] B. Smith and G. Linden, "Two decades of recommender systems at amazon. com," *IEEE Internet Computing*, vol. 21, no. 3, pp. 12–18, 2017.
[36] Y. Koren and R. Bell, "Advances in collaborative filtering," in *Recommender systems handbook*. Springer, 2015, pp. 77–118.
[37] Y. Shi, M. Larson, and A. Hanjalic, "Collaborative filtering beyond the user-item matrix: A survey of the state of the art and future challenges," *CSUR*, vol. 47, no. 1, p. 3, 2014.
[38] Y. Cai, H.-f. Leung, Q. Li, H. Min, J. Tang, and J. Li, "Typicality-based collaborative filtering recommendation," *IEEE TKDE*, vol. 26, no. 3, pp. 766–779, 2014.
[39] M. J. Pazzani and D. Billsus, "Content-based recommendation systems," in *The adaptive web*. Springer, 2007, pp. 325–341.
[40] J. Lian, F. Zhang, X. Xie, and G. Sun, "Cccfnet: A content-boosted collaborative filtering neural network for cross domain recommender systems," in *WWW*. WWW Steering Committee, 2017, pp. 817–818.
[41] L. Yao, Q. Z. Sheng, A. Segev, and J. Yu, "Recommending web services via combining collaborative filtering with content-based features," in *ICWS*. IEEE, 2013, pp. 42–49.
[42] N. Rubens, M. Elahi, M. Sugiyama, and D. Kaplan, "Active learning in recommender systems," in *Recommender systems handbook*. Springer, 2015, pp. 809–846.
[43] N. Houlsby, J. M. Hernández-Lobato, and Z. Ghahramani, "Cold-start active learning with robust ordinal matrix factorization," in *ICML*, 2014, pp. 766–774.

[44] W. Niu, J. Caverlee, and H. Lu, "Neural personalized ranking for image recommendation," in *WSDM*. ACM, 2018, pp. 423–431.

[45] H. Wang, N. Wang, and D.-Y. Yeung, "Collaborative deep learning for recommender systems," in *SIGKDD*. ACM, 2015, pp. 1235–1244.

[46] P. Covington, J. Adams, and E. Sargin, "Deep neural networks for youtube recommendations," in *RecSys*. ACM, 2016, pp. 191–198.

[47] L. Liao, X. He, H. Zhang, and T.-S. Chua, "Attributed social network embedding," *IEEE TKDE*, vol. 30, no. 12, pp. 2257–2270, 2018.

[48] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, "Bpr: Bayesian personalized ranking from implicit feedback," in *UAI*. AUAI Press, 2009, pp. 452–461.

[49] J. Xu, Y. Yao, H. Tong, X. Tao, and J. Lu, "Rapare: A generic strategy for cold-start rating prediction problem," *IEEE TKDE*, vol. 29, no. 6, pp. 1296–1309, 2017.

[50] J. Lin, K. Sugiyama, M.-Y. Kan, and T.-S. Chua, "Addressing cold-start in app recommendation: latent user models constructed from twitter followers," in *ACM SIGIR*. ACM, 2013, pp. 283–292.

[51] L. Xu, X. Wei, J. Cao, and P. S. Yu, "On learning mixed community-specific similarity metrics for cold-start link prediction," in *WWW*. WWW Steering Committee, 2017, pp. 861–862.

[52] J. Noel, S. Sanner, K.-N. Tran, P. Christen, L. Xie, E. V. Bonilla, E. Abbasnejad, and N. Della Penna, "New objective functions for social collaborative filtering," in *WWW*. ACM, 2012, pp. 859–868.

[53] D. Cohen, M. Aharon, Y. Koren, O. Somekh, and R. Nissim, "Expediting exploration by attribute-to-feature mapping for cold-start recommendations," in *ACM RecSys*. ACM, 2017, pp. 184–192.

[54] N. N. Liu, X. Meng, C. Liu, and Q. Yang, "Wisdom of the better few: cold start recommendation via representative based rating elicitation," in *ACM RecSys*. ACM, 2011, pp. 37–44.

[55] X. Dong, L. Yu, Z. Wu, Y. Sun, L. Yuan, and F. Zhang, "A hybrid collaborative filtering model with deep structure for recommender systems." in *AAAI*, 2017, pp. 1309–1315.

[56] M. Vartak, A. Thiagarajan, C. Miranda, J. Bratman, and H. Larochelle, "A meta-learning perspective on cold-start recommendations for items," in *NIPS*, 2017, pp. 6904–6914.

[57] J. Wei, J. He, K. Chen, Y. Zhou, and Z. Tang, "Collaborative filtering and deep learning based recommendation system for cold start items," *Expert Systems with Applications*, vol. 69, pp. 29–39, 2017.

[58] I. Szpektor, A. Gionis, and Y. Maarek, "Improving recommendation for long-tail queries via templates," in *WWW*. ACM, 2011, pp. 47–56.

[59] L. Shi, "Trading-off among accuracy, similarity, diversity, and long-tail: A graph-based recommendation approach," in *ACM RecSys*. ACM, 2013, pp. 57–64.

[60] M. A. Domingues, F. Gouyon, A. M. Jorge, J. P. Leal, J. Vinagre, L. Lemos, and M. Sordo, "Combining usage and content in an online recommendation system for music in the long tail," *IJMIR*, vol. 2, no. 1, pp. 3–13, 2013.

[61] J. Li, Y. Wu, J. Zhao, and K. Lu, "Low-rank discriminant embedding for multiview learning," *IEEE TCYB*, 2016.

[62] G. Liu and S. Yan, "Latent low-rank representation for subspace segmentation and feature extraction," in *ICCV*. IEEE, 2011, pp. 1615–1622.

[63] J. D. Rennie and N. Srebro, "Fast maximum margin matrix factorization for collaborative prediction," in *ICML*. ACM, 2005, pp. 713–719.

[64] E. J. Candès, X. Li, Y. Ma, and J. Wright, "Robust principal component analysis," *JACM*, vol. 58, no. 3, p. 11, 2011.

[65] S. Vargas and P. Castells, "Rank and relevance in novelty and diversity metrics for recommender systems," in *RecSys*. ACM, 2011, pp. 109–116.

[66] C.-N. Ziegler, S. M. McNee, J. A. Konstan, and G. Lausen, "Improving recommendation lists through topic diversification," in *WWW*, 2005, pp. 22–32.

[67] Z. Ding, M. Shao, and Y. Fu, "Missing modality transfer learning via latent low-rank constraint," *IEEE TIP*, vol. 24, no. 11, pp. 4322–4334, 2015.

[68] J. Li, J. Zhao, and K. Lu, "Joint feature selection and structure preservation for domain adaptation." in *IJCAI*, 2016, pp. 1697–1703.

[69] L. Zhu, J. Shen, L. Xie, and Z. Cheng, "Unsupervised visual hashing with semantic assistant for content-based image retrieval," *IEEE TKDE*, vol. 29, no. 2, pp. 472–486, 2017.

[70] R. Mazumder, T. Hastie, and R. Tibshirani, "Spectral regularization algorithms for learning large incomplete matrices," *JMLR*, vol. 11, no. Aug, pp. 2287–2322, 2010.

[71] A. Srebro, Nathanand Shraibman, "Rank, trace-norm and max-norm," in *COLT*, 2005, pp. 545–560.

[72] C. Xu, D. Tao, and C. Xu, "Robust extreme multi-label learning," in *ACM SIGKDD*, 2016, pp. 13–17.

[73] M. Mohri, A. Rostamizadeh, and A. Talwalkar, *Foundations of machine learning*, 2012.

[74] I. Cantador, P. Brusilovsky, and T. Kuflik, "2nd workshop on information heterogeneity and fusion in recommender systems (hetrec 2011)," in *ACM RecSys*. New York, NY, USA: ACM, 2011.

[75] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE TKDE*, vol. 22, no. 10, pp. 1345–1359, 2010.

Jingjing Li received his M.Sc. degree and Ph.D. degree in Computer Science from University of Electronic Science and Technology of China in 2013 and 2017, respectively. Now he is a postdoc researcher of national postdoctoral program for innovative talents in School of Computer Science and Engineering, University of Electronic Science and Technology of China. He has great interest in computer vision, transfer learning and recommender systems.

Ke Lu received his B.S. degree in thermal power engineering from Chongqing University, China in 1996. He obtained his M.Sc. and Ph.D. degrees in Computer Application Technology from the University of Electronic Science and Technology of China, in 2003 and 2006, respectively. He is currently a professor in School of Computer Science and Engineering, University of Electronic Science and Technology of China. His research interests include pattern recognition, multimedia and computer vision.

Zi Huang received the B.Sc. degree in computer science from Tsinghua University, China, and the Ph.D. degree in computer science from the School of Information Technology and Electrical Engineering, The University of Queensland, Australia. She is an ARC Future Fellow with the School of Information Technology and Electrical Engineering, The University of Queensland. Her research interests include multimedia indexing and search, social data analysis, and knowledge discovery.

Heng Tao Shen iis currently a Professor of National "Thousand Talents Plan", the Dean of School of Computer Science and Engineering, and the Director of Center for Future Media at the University of Electronic Science and Technology of China. He is also an Honorary Professor at the University of Queensland. He obtained his BSc with 1st class Honours and PhD from Department of Computer Science, National University of Singapore in 2000 and 2004 respectively. He then joined the University of Queensland as a Lecturer, Senior Lecturer, Reader, and became a Professor in late 2011. His research interests mainly include Multimedia Search, Computer Vision, Artificial Intelligence, and Big Data Management. He has published 200+ peer-reviewed papers, most of which appeared in top ranked publication venues, such as ACM Multimedia, CVPR, ICCV, AAAI, IJCAI, SIGMOD, VLDB, ICDE, TOIS, TIP, TPAMI, TKDE, VLDB Journal, etc. He has received 6 Best Paper Awards from international conferences, including the Best Paper Award from ACM Multimedia 2017 and Best Paper Award - Honorable Mention from ACM SIGIR 2017. He has served as a PC Co-Chair for ACM Multimedia 2015 and currently is an Associate Editor of IEEE Transactions on Knowledge and Data Engineering.