Transfer to Rank for Heterogeneous One-Class Collaborative Filtering

10

WEIKE PAN, College of Computer Science and Software Engineering and National Engineering Laboratory for Big Data System Computing Technology, Shenzhen University
QIANG YANG, Department of Computer Science and Engineering, Hong Kong University of Science and Technology
WANLING CAI, YAOFENG CHEN, QING ZHANG, XIAOGANG PENG, and ZHONG MING, Shenzhen University

Heterogeneous one-class collaborative filtering is an emerging and important problem in recommender systems, where two different types of one-class feedback, i.e., purchases and browses, are available as input data. The associated challenges include ambiguity of browses, scarcity of purchases, and heterogeneity arising from different feedback. In this article, we propose to model purchases and browses from a new perspective, i.e., users' roles of mixer, browser and purchaser. Specifically, we design a novel transfer learning solution termed *role-based transfer to rank* (RoToR), which contains two variants, i.e., integrative RoToR and sequential RoToR. In integrative RoToR, we leverage browses into the preference learning task of purchases, in which we take each user as a sophisticated customer (i.e., *mixer*) that is able to take different types of feedback into consideration. In sequential RoToR, we aim to simplify the integrative one by decomposing it into two dependent phases according to a typical shopping process. Furthermore, we instantiate both variants using different preference learning paradigms such as pointwise preference learning and pairwise preference learning. Finally, we conduct extensive empirical studies with various baseline methods on three large public datasets and find that our RoToR can perform significantly more accurate than the state-of-the-art methods.

CCS Concepts: • **Information systems → Personalization**; • **Human-centered computing → Collaborative filtering**;

Additional Key Words and Phrases: Heterogeneous one-class collaborative filtering, one-class feedback, transfer to rank, role-based recommendation

Qiang Yang, Xiaogang Peng, and Zhong Ming are co-corresponding authors for this work.
We thank the support of Natural Science Foundation of China (NSFC) Nos. 61502307 and 61672358, China National Fundamental Research (973 Program) No. 2014CB340304, Hong Kong CERG projects Nos. 16211214, 16209715, and 16244616, Hong Kong ITF ITS/391/15FX, and Natural Science Foundation of Guangdong Province No. 2016A030313038.
Authors' addresses: W. Pan, W. Cai, Y. Chen, Q. Zhang, X. Peng, and Z. Ming, College of Computer Science and Software Engineering and National Engineering Laboratory for Big Data System Computing Technology, Shenzhen University, 3688# Nanhai Avenue, Nanshan District, Shenzhen, China, 518060; emails: panweike@szu.edu.cn, wanlingcai@qq.com, chenyaofeng@email.szu.edu.cn, qingzhang1992@qq.com, {pengxg, mingz}@szu.edu.cn; Q. Yang, Department of Computer Science and Engineering, Hong Kong University of Science and Technology, Clearwater Bay, Hong Kong, China; email: qyang@cse.ust.hk.

1046-8188/2019/01-ART10 $15.00
https://doi.org/10.1145/3243652

ACM Reference format:
Weike Pan, Qiang Yang, Wanling Cai, Yaofeng Chen, Qing Zhang, Xiaogang Peng, Zhong Ming, 2019. Transfer to Rank for Heterogeneous One-Class Collaborative Filtering. *ACM Trans. Inf. Syst.* 37, 1, Article 10 (January 2019), 20 pages.
https://doi.org/10.1145/3243652

1 INTRODUCTION

In recommender systems, users' behaviors are one of the most important sources of information considering their value of learning and mining users' preferences for personalized delivery or advertisement. For a typical online system, users' behaviors are usually in different forms. For example, users may browse some products before purchasing one from an e-commerce site, go through some restaurants before selecting one in a group-buying app, or read some introduction before taking a course on a massive open online course (MOOC) site, and so on. Based on this observation, some recent recommendation works have switched from traditional problem settings such as rating prediction or item recommendation with homogeneous user feedback to item ranking with heterogeneous feedback. In particular, heterogeneous implicit feedback (HIF) [21] or heterogeneous one-class collaborative filtering (HOCCF) [19] is an emerging branch in the research community of recommender systems. To fully make use of such heterogeneous feedback of purchases and browses, there are some specific challenges arising from the characteristics of the data. For instance, the browse data are of some ambiguity regarding the users' preferences, the purchase data are usually few as compared with the browse data, and the feedback are heterogeneous as they are associated with different actions and intents.

Some recent works have attempted to adapt some seminal recommendation algorithms for modeling homogeneous one-class feedback to the task of HOCCF. For instance, in *adaptive Bayesian personalized ranking* (ABPR) [21], an adaptive confidence learning algorithm based on TrAdaBoost [5] and BPR [24] is developed, where the learned confidence-weighted feedback are then modeled in the well-known pairwise preference learning framework. In *transfer via joint similarity learning* (TJSL) [19], a joint similarity learning algorithm based on FISM [12] is designed, where two different types of similarities are learned, one for that between candidate items and purchased items, and the other for that between candidate items and likely-to-prefer items as identified from the browsed ones. Both ABPR [21] and TJSL [19] are based on transfer learning methods [18, 37] for sharing knowledge between two different types of feedback, which are expected to address the scarcity challenge of the purchase data well. However, a major limitation of ABPR and TJSL is the efficiency, which is also observed in our empirical studies, especially on a large dataset. Very recently, a simple yet effective method called *role-based Bayesian personalized ranking* (RBPR) [22] is proposed to address the efficiency issue in ABPR and TJSL via a two-stage sequential framework. However, the sequential modeling technique in RBPR is lack of principled foundation for preference learning in two stages. Furthermore, the performance improvement obtained in RBPR is not very significant as compared with some strong baselines such as matrix factorization with logistic loss and item-oriented memory-based collaborative filtering, which can also be found in our empirical studies.

In this article, we design a novel transfer learning solution from the perspective of users' roles of mixer, browser, and purchaser, i.e., role-based transfer to rank (RoToR). First, we propose to digest the heterogeneous feedback from the perspective of a sophisticated customer (i.e., mixer) that is able to make well-informed decisions based on a mix of historical actions of purchases and browses, and then design an integrative algorithm called RoToR(int.). Second, we further simplify RoToR(int.) by decomposing it into two sequential phases from the perspective of first as a browser and then as a purchaser, which is expected not to sacrifice the accuracy. We denote the sequential

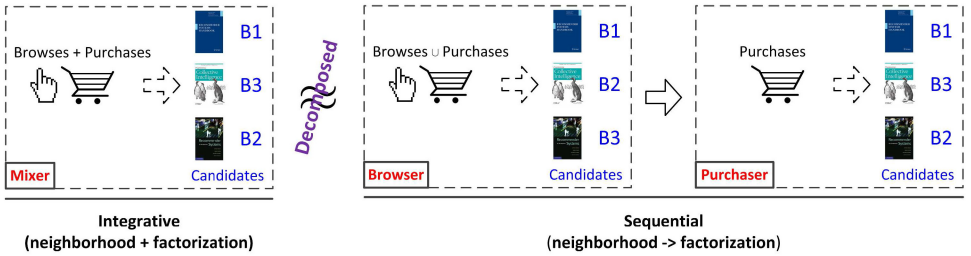

Fig. 1. Illustration of role-based transfer to rank (RoToR) for heterogeneous one-class collaborative filtering (HOCCF), including integrative RoToR and sequential RoToR.

variant as RoToR(seq.), which is easier for implementation, deployment, and maintenance. Furthermore, we adopt different preference learning paradigms in RoToR(int.) and RoToR(seq.), i.e., pointwise preference learning and pairwise preference learning, and thus have four specific algorithms, including RoToR(poi.,int.), RoToR(pai.,int.), RoToR(poi.,seq.) and RoToR(pai.,seq.). Notice that we do not include listwise preference learning because they are usually designed for numerical scores in recommendation and information retrieval [31, 32] instead of one-class data in our studied problem. And designing a novel listwise learning to rank algorithm is also vertical to our focus in this article. Finally, we conduct extensive empirical studies with various baseline methods and showcase the effectiveness of our RoToR in delivering accurate top-K items to each end user.

We summarize our main contributions as follows: (i) we study an important recommendation problem, i.e., heterogeneous one-class collaborative filtering (HOCCF), from a new perspective of users' roles in a principled way; (ii) we design a novel transfer learning solution, i.e., RoTor, for the studied problem; and (iii) we conduct extensive empirical studies on three large datasets with various baseline methods, and showcase the effectiveness of our proposed solution.

2 ROLE-BASED TRANSFER TO RANK

In this section, we first formally define the studied problem and then describe our proposed transfer learning solution, i.e., RoToR, in detail. Our RoToR, as shown in Figure 1, contains an integrative variant and a sequential variant denoted by RoToR(int.) and RoToR(seq.), respectively. The first variant RoToR(int.) interprets the heterogeneous one-class feedback as a whole from the perspective of a sophisticated customer (i.e., mixer) who is able to make judgements based on two different types of feedback systematically. The second variant RoToR(seq.) is a decomposed version of RoToR(int.) mainly for simplicity, which consists of two dependent learning phases w.r.t. that of two different roles of browser and purchaser. We will describe those two variants in detail in the sequel.

2.1 Problem Definition

In the studied HOCCF problem, we have two different types of one-class feedback, i.e., browses $\mathcal{B} = \{(u, i)\}$ and purchases $\mathcal{P} = \{(u, i')\}$, representing the recorded online behaviors of a set of users $\mathcal{U} = \{u\}$ to a set of items $\mathcal{I} = \{i\}$. We use $\mathcal{B}_u = \{i|(u, i) \in \mathcal{B}\}$ and $\mathcal{P}_u = \{i'|(u, i') \in \mathcal{P}\}$ to denote the set of browsed items and the set of purchased items by user u, respectively. The goal of HOCCF is then to rank the not-yet purchased items, i.e., $\mathcal{I} \backslash \mathcal{P}_u$, for each end user $u \in \mathcal{U}$.

Notice that the main characteristic of HOCCF as compared with the well-studied one-class collaborative filtering (OCCF) problem is that the input data of HOCCF is heterogeneous one-class feedback such as purchases and browses instead of homogeneous one-class feedback such as purchases only.

We list some commonly used notations and their explanations in Table 1 for quick reference.

Table 1. Some Notations and Explanations

$n = \|\mathcal{U}\|$	user number
$m = \|\mathcal{I}\|$	item number
$u \in \mathcal{U}$	user ID
$i, i' \in \mathcal{I}$	item ID
$\mathcal{R} = \{(u, i)\}$	universe of all possible (user, item) pairs
$\mathcal{P} = \{(u, i)\}$	(user, item) pairs denoting purchases
$\mathcal{P}_u$	set of items purchased by user u
$\mathcal{B} = \{(u, i')\}$	(user, item) pairs denoting browses
$\mathcal{B}_u$	set of items browsed by user u
$\mathcal{A}$	sampled negative feedback from $\mathcal{R} \backslash \mathcal{P}$
r_{ui}	$r_{ui} = 1$ if $(u, i) \in \mathcal{P}$ and $r_{ui} = -1$ if $(u, i) \in \mathcal{A}$
$d \in \mathbb{R}$	number of latent dimensions
$b_u \in \mathbb{R}$	user bias
$b_i \in \mathbb{R}$	item bias
$U_{u\cdot} \in \mathbb{R}^{1\times d}$	user-specific latent feature vector
$V_{i\cdot} \in \mathbb{R}^{1\times d}$	item-specific latent feature vector
$W_{i'\cdot} \in \mathbb{R}^{1\times d}$	item-specific latent feature vector
$\hat{r}_{ui}$	predicted preference of user u to item i
$\hat{r}_{uij}, \hat{r}^{(F)}_{uij}$,	preference difference in pairwise preference learning
$\hat{r}^{(F)}_{ui}$	predicted preference of user u to item i in a factorization-based method
$\hat{r}^{(N)}_{ui}$	predicted preference of user u to item i in a neighborhood-based method
$s^{(\ell)}_{i'i}$	learned similarity between item i' and item i
$\hat{r}^{(N')}_{ui}$	predicted preference of user u to item i in item-oriented CF
$s^{(p)}_{i'i}$	predefined similarity (Jaccard index) between item i' and item i
$\mathcal{N}_i$	A nearest set of items of item i
T	iteration number in the algorithm

2.2 Integrative Role-Based Transfer to Rank

In our integrative variant of RoToR, i.e., RoToR(int.), we assume that a user acts as a mixer and is able to integrate two different types of one-class feedback in his/her decision-making process. Specifically, we follow the seminal work on integrating heterogeneous feedback of ratings and examinations [13], and have the estimated preference of user u to item i as follows:

$$\hat{r}_{ui} = \hat{r}^{(F)}_{ui} + \hat{r}^{(N)}_{ui}, \tag{1}$$

where $\hat{r}^{(F)}_{ui} = b_u + b_i + U_{u\cdot}V^T_{i\cdot}$ and $\hat{r}^{(N)}_{ui} = \frac{1}{\sqrt{|\mathcal{B}_u|}} \sum_{i' \in \mathcal{B}_u} s^{(\ell)}_{i'i} = \frac{1}{\sqrt{|\mathcal{B}_u|}} \sum_{i' \in \mathcal{B}_u} W_{i'\cdot}V^T_{i\cdot}$ are the prediction rules of the classical factorization-based method and the neighborhood-based method, respectively. The variables in those two prediction rules are the user bias $b_u \in \mathbb{R}$, the item bias $b_i \in \mathbb{R}$, the user-specific latent feature vector $U_{u\cdot} \in \mathbb{R}^{1\times d}$, the item-specific latent feature vector $V_{i\cdot} \in \mathbb{R}^{1\times d}$, and the item-specific latent feature vector $W_{i'\cdot} \in \mathbb{R}^{1\times d}$ in the learned similarity $s^{(\ell)}_{i'i}$. Notice that the terms $\hat{r}^{(F)}_{ui}$ and $\hat{r}^{(N)}_{ui}$ in Equation (1) are used to model the purchases and browses, respectively. Such integrative modeling will usually improve the recommendation accuracy. The reason is that the

knowledge of users' preferences beneath browses are transferred to the learning task of purchases in a way that ensures users with similar browsing history will have similar purchasing behaviors in the future. The differences between our work and SVD++ [13] are mainly threefold: (i) we define the prediction rule on purchases and browses, instead of on ratings and examinations in Reference [13]; (ii) we use the prediction rule for item ranking, instead of for rating prediction in Reference [13]; and (iii) we embed the prediction rule in pointwise logistic loss and pairwise loss, instead of pointwise square loss in Reference [13]. Notice that when more types of user behaviors are available, we may further expand the prediction rule in Equation (1) with additional terms defined on other types of behaviors.

With the expanded prediction rule that is able to fuse two different types of one-class feedback as shown in Equation (1), we adopt two different preference learning paradigms, i.e., pointwise preference learning and pairwise preference learning, in two specific algorithms, which will be described in the subsequent two subsections.

2.2.1 Pointwise Preference Learning. In pointwise preference learning, we model each (user, item) pair of the behavior data independently. To address the problem of lack of negative feedback, we sample some (user, item) pairs that are not observed in the purchase data, i.e., $\mathcal{A}$ from $\mathcal{R}\backslash\mathcal{P}$, where $|\mathcal{A}| = \rho|\mathcal{P}|$ [12]. With the positive feedback in $\mathcal{P}$ and the sampled negative feedback in $\mathcal{A}$, we have the objective function as follows:

$$\min_{\Theta} \sum_{(u,i)\in\mathcal{P}\cup\mathcal{A}} f_{ui}, \tag{2}$$

where $\Theta = \{U_{u\cdot}, b_u, u = 1, 2, \ldots, n; W_{i\cdot}, V_{i\cdot}, b_i, i = 1, 2, \ldots, m\}$ are model parameters to be learned, and $f_{ui} = \log(1 + \exp(-r_{ui}\hat{r}_{ui})) + \frac{\alpha_u}{2}||U_{u\cdot}||_F^2 + \frac{\alpha_v}{2}||V_{i\cdot}||_F^2 + \frac{\alpha_w}{2}\sum_{i'\in\mathcal{B}_u}||W_{i'\cdot}||_F^2 + \frac{\beta_u}{2}b_u^2 + \frac{\beta_v}{2}b_i^2$ is the tentative objective function for the (u, i) pair. Notice that the prediction rule is $\hat{r}_{ui} = \hat{r}_{ui}^{(F)} + \hat{r}_{ui}^{(N)} = b_u + b_i + U_{u\cdot}V_{i\cdot}^T + \frac{1}{\sqrt{|\mathcal{B}_u|}}\sum_{i'\in\mathcal{B}_u} W_{i'\cdot}V_{i\cdot}^T$, and $r_{ui} = 1$ if $(u, i) \in \mathcal{P}$ and $r_{ui} = -1$ if $(u, i) \in \mathcal{A}$, denoting a positive and negative preference, respectively.

We can see that our integrative RoToR with pointwise preference learning, i.e., RoToR(poi.,int.), is mainly based on the prediction rule of SVD++ [13] and the pointwise preference learning paradigm in LogisticMF [11]. We design RoToR(poi.,int.) in this way, aiming to inherit the merits of SVD++ in modeling heterogeneous feedback and that of LogisticMF in modeling one-class feedback.

To solve the optimization problem in Equation (2), we adopt the commonly used stochastic gradient descent (SGD) algorithm. Specifically, for each $(u, i) \in \mathcal{P} \cup \mathcal{A}$, we have the gradients of $\frac{\partial f_{ui}}{\partial U_{u\cdot}}$, $\frac{\partial f_{ui}}{\partial V_{i\cdot}}$, $\frac{\partial f_{ui}}{\partial b_u}$, $\frac{\partial f_{ui}}{\partial b_i}$ and $\frac{\partial f_{ui}}{\partial W_{i'\cdot}}$ as follows:

$$\frac{\partial f_{ui}}{\partial U_{u\cdot}} = -r_{ui}\sigma(-r_{ui}\hat{r}_{ui})V_{i\cdot} + \alpha_u U_{u\cdot}, \tag{3}$$

$$\frac{\partial f_{ui}}{\partial V_{i\cdot}} = -r_{ui}\sigma(-r_{ui}\hat{r}_{ui})(U_{u\cdot} + \bar{U}_{u\cdot}) + \alpha_v V_{i\cdot}, \tag{4}$$

$$\frac{\partial f_{ui}}{\partial b_u} = -r_{ui}\sigma(-r_{ui}\hat{r}_{ui}) + \beta_u b_u, \tag{5}$$

$$\frac{\partial f_{ui}}{\partial b_i} = -r_{ui}\sigma(-r_{ui}\hat{r}_{ui}) + \beta_v b_i, \tag{6}$$

$$\frac{\partial f_{ui}}{\partial W_{i'\cdot}} = -r_{ui}\sigma(-r_{ui}\hat{r}_{ui})\frac{1}{\sqrt{|\mathcal{B}_u|}}V_{i\cdot} + \alpha_w W_{i'\cdot}, i' \in \mathcal{B}_u, \tag{7}$$

where $\sigma(x) = \frac{1}{1+\exp(-x)}$ is the sigmoid function, and $\bar{U}_{u\cdot} = \frac{1}{|\mathcal{B}_u|}\sum_{i' \in \mathcal{B}_u} W_{i'\cdot}$ is a certain virtual user-specific latent feature vector of user u aggregated from the set of browsed items $\mathcal{B}_u$.

With the gradients in Equations (3)–(7), we can update the model parameters as follows:

$$\theta_{\tau+1} \leftarrow \theta_\tau - \gamma \frac{\partial f_{ui}}{\partial \theta_\tau}, \tag{8}$$

where $\theta_\tau \in \Theta$ denotes a certain model parameter, i.e., $U_{u\cdot}$, $V_{i\cdot}$, b_u, b_i or $W_{i'\cdot}$ with $i' \in \mathcal{B}_u$, and γ is the learning rate.

We depict the learning algorithm in Algorithm 1. In Algorithm 1, we can see that the algorithm mainly contains two loops. In the outer loop, we randomly sample a set of not-yet purchased items for each user to construct an expanded set of user behaviors with both positive feedback and negative feedback, i.e., $\mathcal{P} \cup \mathcal{A}$. In the inner loop, we update the model parameters based on each randomly drawn (u, i) pair from $\mathcal{P} \cup \mathcal{A}$. Notice that the knowledge transfer process is reflected in the prediction rule that fuses the two types of one-class feedback.

ALGORITHM 1: The algorithm of integrative RoToR with pointwise preference learning, i.e., RoToR (poi.,int.).

Input: Purchases $\mathcal{P}$ and Browses $\mathcal{B}$
Output: Top-K recommended items for each user
Initialization: Initialize model parameters Θ

```
for t = 1, ..., T do
    Randomly pick up a set A with |A| = ρ|P|
    for t2 = 1, ..., |P ∪ A| do
        Randomly pick up a pair (u, i) from P ∪ A
        Calculate r̂_ui via Equation (1)
        Calculate the gradients via Equations (3)–(7)
        Update the model parameters via Equation (8)
    end for
end for
```

2.2.2 Pairwise Preference Learning. In pairwise preference learning, we focus on the difference of user u's preference to item i and item j, i.e., $\hat{r}_{ui} - \hat{r}_{uj}$, from which we can see that the user bias b_u in $\hat{r}_{ui}^{(F)}$ of Equation (1) will then be of no use in such a difference. We thus remove b_u in pairwise preference learning, and have the prediction rule $\hat{r}_{ui} = \hat{r}_{ui}^{(F)} + \hat{r}_{ui}^{(N)} = b_i + U_{u\cdot}V_{i\cdot}^T + \frac{1}{\sqrt{|\mathcal{B}_u|}}\sum_{i' \in \mathcal{B}_u} W_{i'\cdot}V_{i\cdot}^T$. Finally, we adopt the classical pairwise objective function for purchases for preference learning [24],

$$\min_{\tilde{\Theta}} \sum_{u \in \mathcal{U}} \sum_{i \in \mathcal{P}_u} \sum_{j \in \mathcal{I} \setminus \mathcal{P}_u} f_{uij}, \tag{9}$$

where $\tilde{\Theta} = \{U_{u\cdot}, u = 1, 2, \ldots, n; W_{i\cdot}, V_{i\cdot}, b_i, i = 1, 2, \ldots, m\}$ denotes the set of parameters to be learned, and $f_{uij} = -\ln \sigma(\hat{r}_{ui} - \hat{r}_{uj}) + \frac{\alpha_u}{2}\|U_{u\cdot}\|^2 + \frac{\alpha_v}{2}\|V_{i\cdot}\|^2 + \frac{\alpha_v}{2}\|V_{j\cdot}\|^2 + \frac{\alpha_w}{2}\sum_{i' \in \mathcal{B}_u}\|W_{i'\cdot}\|_F^2 + \frac{\beta_v}{2}\|b_i\|^2 + \frac{\beta_v}{2}\|b_j\|^2$ is the tentative objective function for every two (user, item) pairs, i.e., (u, i) and (u, j).

We can see that our integrative RoToR with the pairwise preference learning paradigm, i.e., RoToR(pai.,int.), combines the prediction rule of SVD++ [13] and the pairwise preference assumption in BPR [24] in a principled way for the studied problem. Similar to that of RoToR(poi.,int.), our

RoToR(pai.,int.) is expected to inherit the merits of BPR in pairwise preference learning as well as that of SVD++ in heterogeneous preference handling.

To solve the optimization problem in Equation (9), we again follow the common practice of using the efficient SGD algorithm. Specifically, for a randomly sampled triple (u, i, j) in SGD, we have the gradients of $\frac{\partial f_{uij}}{\partial U_{u\cdot}}$, $\frac{\partial f_{uij}}{\partial V_{i\cdot}}$, $\frac{\partial f_{uij}}{\partial V_{j\cdot}}$, $\frac{\partial f_{uij}}{\partial b_i}$, $\frac{\partial f_{uij}}{\partial b_j}$ and $\frac{\partial f_{uij}}{\partial W_{i'\cdot}}$ as follows:

$$\frac{\partial f_{uij}}{\partial U_{u\cdot}} = -\sigma(-\hat{r}_{uij})(V_{i\cdot} - V_{j\cdot}) + \alpha_u U_{u\cdot}, \tag{10}$$

$$\frac{\partial f_{uij}}{\partial V_{i\cdot}} = -\sigma(-\hat{r}_{uij})(U_{u\cdot} + \bar{U}_{u\cdot}) + \alpha_v V_{i\cdot}, \tag{11}$$

$$\frac{\partial f_{uij}}{\partial V_{j\cdot}} = -\sigma(-\hat{r}_{uij})(-U_{u\cdot} - \bar{U}_{u\cdot}) + \alpha_v V_{j\cdot}, \tag{12}$$

$$\frac{\partial f_{uij}}{\partial b_i} = -\sigma(-\hat{r}_{uij}) + \beta_v b_i, \tag{13}$$

$$\frac{\partial f_{uij}}{\partial b_j} = -\sigma(-\hat{r}_{uij})(-1) + \beta_v b_j, \tag{14}$$

$$\frac{\partial f_{uij}}{\partial W_{i'\cdot}} = -\sigma(-\hat{r}_{uij})\frac{V_{i\cdot} - V_{j\cdot}}{\sqrt{|\mathcal{B}_u|}} + \alpha_w W_{i'\cdot}, i' \in \mathcal{B}_u, \tag{15}$$

where $\hat{r}_{uij} = \hat{r}_{ui} - \hat{r}_{uj}$ is the preference difference, and $\bar{U}_{u\cdot} = \frac{1}{|\mathcal{B}_u|}\sum_{i' \in \mathcal{B}_u} W_{i'\cdot}$ is the aggregated feature vector.

With the gradients, we can update the model parameters,

$$\tilde{\theta}_{\tau+1} \leftarrow \tilde{\theta}_\tau - \gamma \frac{\partial f_{uij}}{\partial \tilde{\theta}_\tau}, \tag{16}$$

where $\tilde{\theta}_\tau \in \tilde{\Theta}$ denotes a certain model parameter, i.e., $U_{u\cdot}$, $V_{i\cdot}$, $V_{j\cdot}$, b_i, b_j or $W_{i'\cdot}$ with $i' \in \mathcal{B}$, and γ is the learning rate.

We depict the learning algorithm in Algorithm 2. In Algorithm 2, we can see that in each of the $T|\mathcal{P}|$ loops, we update the model parameters w.r.t. a randomly drawn triple (u, i, j).

ALGORITHM 2: The algorithm of integrative RoToR with pairwise preference learning, i.e., RoToR(pai.,int.).

Input: Purchases $\mathcal{P}$ and Browses $\mathcal{B}$
Output: Top-K recommended items for each user
Initialization: Initialize model parameters $\tilde{\Theta}$

```
for t = 1, ..., T do
    for t2 = 1, ..., |P| do
        Randomly pick up a pair (u, i) from P
        Randomly pick up an item j from I \ P_u
        Calculate r̂_ui and r̂_uj via Equation (1) without b_u
        Calculate r_uij = r̂_ui − r̂_uj
        Calculate the gradients via Equations (10)–(15)
        Update the model parameters via Equation (16)
    end for
end for
```

2.3 Sequential Role-Based Transfer to Rank

The prediction rule in Equation (1) is well known as a seminal work of integrating heterogeneous feedback by two classical recommendation methods, i.e., $\hat{r}_{ui}^{(F)}$ for the factorization-based method and $\hat{r}_{ui}^{(N)}$ for the neighborhood-based method. The improved performance also demonstrates the complementarity of the two different types of feedback as well as the two different recommendation methods. However, the model of RoToR(int.) is of high complexity, considering the development, maintenance and upgrade because of the additional term $\hat{r}_{ui}^{(N)}$. With this observation, we further design a simplified solution without sacrificing the accuracy.

To achieve a good balance between accuracy and simplicity, we propose to decompose the prediction rule in Equation (1) in a spirit of reverse engineering. Specifically, we assume that the complementarity of the factorization-based method and the neighborhood-based method in the prediction rule of integrative RoToR as shown in Equation (1) is likely to be preserved if we can combine the two methods in a different but proper way. In RoToR(int.), the neighborhood-based method assists the factorization-based method via an additional term defined on the browses in the factorization framework, where the latter is more aggressive in approximating the feedback data than the former.

Following the above discussion, we propose to decompose the integrative variant, i.e., RoToR(int.), and combine the two units in a sequential and coarse-to-fine manner, i.e., first, neighborhood-based method, and then factorization-based method. This mechanism is designed to make the preference learning task from less aggressive to more aggressive. Mathematically, we represent the decomposition from an integrative manner to a sequential manner as follows:

$$\hat{r}_{ui}^{(N)} + \hat{r}_{ui}^{(F)} \approx \hat{r}_{ui}^{(N')} \rightarrow \hat{r}_{ui}^{(F)}, \tag{17}$$

where "$\approx$" and "$\rightarrow$" are the decomposition (or approximation) procedure and the sequential relationship, respectively. We denote the sequential variant as RoToR(seq.) and illustrate the decomposition process in the right part of Figure 1. In Figure 1, we can see that RoToR(seq.) contains two dependent phases with shared knowledge of candidate lists of items in a similar way to that of a recent work on transfer to rank (ToR) [20]. The differences between our RoToR(seq.) and ToR [20] are as follows: (i) the studied problem is different, i.e., purchases and browses in this article, and ratings and examinations in Reference [20]; (ii) the technique is different, i.e., a coarse-to-fine combination of a neighborhood-based method and a factorization-based method in this article, and two factorization-based methods with first pairwise preference learning and then pointwise preference learning in Reference [20]. Notice that we may extend the two-phase sequential approach in Equation (17) to a three- or four-phase approach to model more types of user behaviors.

In the first phase of RoToR(seq.), we obtain a candidate list of items via a neighborhood-based method, i.e., item-oriented collaborative filtering (ICF). Specifically, the prediction rule is as follows:

$$\hat{r}_{ui}^{(N')} = \sum_{i' \in \mathcal{N}_i \cap (\mathcal{P}_u \cup \mathcal{B}_u)} s_{i'i}^{(p)}, \tag{18}$$

where $s_{i'i}^{(p)}$ is a predefined similarity (Jaccard index) between item i' and item i based on $\mathcal{P} \cup \mathcal{B}$, and $\mathcal{N}_i$ contains the most similar neighbors of item i. Notice that we treat purchases and browses the same and union two sets of user behaviors when calculating the predefined similarity with the goal of identifying some likely to be examined items in this phase. Specifically, we follow Reference [20] and keep $3K$ items in the candidate list as the shared knowledge, which will be further refined in the second phase.

In the second phase of RoToR(seq.), we have two preference learning paradigms in parallel to that of integrative RoToR, i.e., pointwise preference learning and pairwise preference learning, which will be described in detail in the following two subsections.

2.3.1 Pointwise Preference Learning. For pointwise preference learning in the second phase of sequential RoToR, i.e., RoToR(poi.,seq.), we have a similar objective function to that of Equation (2) in RoToR(poi.,int.) as follows:

$$\min_{\Phi} \sum_{(u,i)\in\mathcal{P}\cup\mathcal{A}} f_{ui}^{(F)}, \tag{19}$$

where $\Phi = \{U_{u\cdot}, b_u, u = 1, 2, \ldots, n; V_{i\cdot}, b_i, i = 1, 2, \ldots, m\}$ are the model parameters, and $f_{ui}^{(F)} = \log(1 + \exp(-r_{ui}\hat{r}_{ui}^{(F)})) + \frac{\alpha_u}{2}||U_{u\cdot}||_F^2 + \frac{\alpha_v}{2}||V_{i\cdot}||_F^2 + \frac{\beta_u}{2}b_u^2 + \frac{\beta_v}{2}b_i^2$ is the tentative prediction rule defined on the (u, i) pair. Notice that the prediction rule is $\hat{r}_{ui}^{(F)} = b_u + b_i + U_{u\cdot}V_{i\cdot}^T$. And $r_{ui} = 1$ if $(u, i) \in \mathcal{P}$ and $r_{ui} = -1$ if $(u, i) \in \mathcal{A}$ denote the positive and negative preference orientation, respectively.

For each $(u, i) \in \mathcal{P} \cup \mathcal{A}$, we have the gradients of $\frac{\partial f_{ui}^{(F)}}{\partial U_{u\cdot}}$, $\frac{\partial f_{ui}^{(F)}}{\partial V_{i\cdot}}$, $\frac{\partial f_{ui}^{(F)}}{\partial b_u}$ and $\frac{\partial f_{ui}^{(F)}}{\partial b_i}$ similar to that of Equations (3)–(6).

With the gradients, we can update the model parameters as follows,

$$\phi_{\tau+1} \leftarrow \phi_\tau - \gamma \frac{\partial f_{ui}^{(F)}}{\partial \phi_\tau}, \tag{20}$$

where $\phi_\tau \in \Phi$ denotes a certain model parameter, i.e., $U_{u\cdot}$, $V_{i\cdot}$, b_u or b_i, and γ is the learning rate.

We depict the learning algorithm of RoToR(poi.,seq.) in Algorithm 3. In Algorithm 3, we can see that it first conducts neighborhood-based preference learning so as to generate $3K$ candidate items, and then refines the candidate list by the learned factorization-based pointwise preference learning model. The shared knowledge between the two tasks is the candidate list of items, which is the same with that of ToR [20].

ALGORITHM 3: The algorithm of sequential RoToR with pointwise preference learning, i.e., RoToR (poi.,seq.).

1: **Input**: Purchases $\mathcal{P}$ and browses $\mathcal{B}$
2: **Output**: Top-K recommended items for each user
3: Initialize model parameters Φ
4: Conduct neighborhood-based preference learning via Equation (18) and obtain $3K$ candidate items with highest predicted scores
5: Conduct factorization-based pointwise preference learning via the algorithm in Algorithm 1 with the gradients $\frac{\partial f_{ui}^{(F)}}{\partial U_{u\cdot}}$, $\frac{\partial f_{ui}^{(F)}}{\partial V_{i\cdot}}$, $\frac{\partial f_{ui}^{(F)}}{\partial b_u}$ and $\frac{\partial f_{ui}^{(F)}}{\partial b_i}$ similar to that of Equations (3)–(6) and update rule in Equation (20); predict the scores on those $3K$ candidate items and refine the candidate list of items

2.3.2 Pairwise Preference Learning. For pairwise preference learning in the second phase, i.e., RoToR(pai.,seq.), we follow the optimization problem in BPR [24],

$$\min_{\tilde{\Phi}} \sum_{u\in\mathcal{U}} \sum_{i\in\mathcal{P}_u} \sum_{j\in\mathcal{I}\setminus\mathcal{P}_u} f_{uij}^{(F)}, \tag{21}$$

where $\tilde{\Phi} = \{U_{u\cdot}, u = 1, 2, \ldots, n; V_{i\cdot}, b_i, i = 1, 2, \ldots, m\}$ denotes the set of parameters to be learned, $f_{uij}^{(F)} = -\ln\sigma(\hat{r}_{ui}^{(F)} - \hat{r}_{uj}^{(F)}) + \frac{\alpha_u}{2}\|U_{u\cdot}\|^2 + \frac{\alpha_v}{2}\|V_{i\cdot}\|^2 + \frac{\alpha_v}{2}\|V_{j\cdot}\|^2 + \frac{\beta_v}{2}\|b_i\|^2 + \frac{\beta_v}{2}\|b_j\|^2$, and

$\hat{r}_{ui}^{(F)} = b_i + U_u . V_{i\cdot}^T$ is the prediction rule without the user bias b_u because it is of no use in preference difference calculation, i.e., $\hat{r}_{ui}^{(F)} - \hat{r}_{uj}^{(F)}$, in the pairwise preference learning paradigm similar to that of RoToR(pai.,int.).

For each triple (u, i, j), we have the gradients of $\frac{\partial f_{uij}^{(F)}}{\partial U_{u\cdot}}$, $\frac{\partial f_{uij}^{(F)}}{\partial V_{i\cdot}}$, $\frac{\partial f_{uij}^{(F)}}{\partial V_{j\cdot}}$, $\frac{\partial f_{uij}^{(F)}}{\partial b_i}$, and $\frac{\partial f_{uij}^{(F)}}{\partial b_j}$ similar to that of Equations (10)–(14).

With the gradients, we can update the model parameters,

$$\tilde{\phi}_{\tau+1} \leftarrow \tilde{\phi}_\tau - \gamma \frac{\partial f_{ui}^{(F)}}{\partial \tilde{\phi}_\tau}, \tag{22}$$

where $\tilde{\phi}_\tau \in \tilde{\Phi}$ denotes a certain model parameter, i.e., $U_{u\cdot}$, $V_{i\cdot}$, $V_{j\cdot}$, b_i or b_j, and γ is the learning rate.

Once we have learned the model parameters, we can apply the learned model to the candidate list and obtain an updated and finalized recommendation list. We depict the learning algorithm of RoToR(pai.,seq.) in Algorithm 4. In Algorithm 4, we can see that the algorithm is similar to that of RoToR(poi.,seq.) in Algorithm 3 except the preference learning paradigm in the second phase.

ALGORITHM 4: The algorithm of sequential RoToR with pairwise preference learning, i.e., RoToR(pai.,seq.)

1: **Input**: Purchases $\mathcal{P}$ and browses $\mathcal{B}$
2: **Output**: Top-K recommended items for each user
3: Initialize model parameters $\tilde{\Phi}$
4: Conduct neighborhood-based preference learning via Equation (18) and obtain $3K$ candidate items with highest predicted scores
5: Conduct factorization-based pairwise preference learning via the algorithm in Algorithm 2 with the gradients $\frac{\partial f_{uij}^{(F)}}{\partial U_{u\cdot}}$, $\frac{\partial f_{uij}^{(F)}}{\partial V_{i\cdot}}$, $\frac{\partial f_{uij}^{(F)}}{\partial V_{j\cdot}}$, $\frac{\partial f_{uij}^{(F)}}{\partial b_i}$, and $\frac{\partial f_{uij}^{(F)}}{\partial b_j}$ similar to that of Equations (10)–(14) and update rule in Equation (14); predict the scores on those $3K$ candidate items and refine the candidate list of items

3 EXPERIMENTAL RESULTS

In this section, we conduct empirical studies to verify the following two hypotheses:

- —RoToR(poi.,int.) and RoToR(pai.,int.) are designed to integrate two different types of one-class feedback in one single prediction rule, which is then embedded in the classical pointwise and pairwise preference learning paradigms. We thus would like to see whether RoToR(poi.,int.) and RoToR(pai.,int.) inherit the merits of SVD++ [13] for combining the neighborhood-based method and the factorization-based method, and that of logistic matrix factorization (LogisticMF) [11] and Bayesian personalized ranking (BPR) [24] for pointwise and pairwise preference learning, respectively.
- —RoToR(poi.,seq.) and RoToR(pai.,seq.) are proposed to simplify RoToR(poi.,int.) and RoToR(pai.,int.) by decomposing their prediction rules into two separate and dependent parts, which results in two coarse-to-fine sequential learning algorithms. We would like to verify the hypothesis that RoToR(poi.,seq.) and RoToR(pai.,seq.) will not sacrifice the accuracy much as compared with RoToR(poi.,int.) and RoToR(pai.,int.), respectively, because the merit of the neighborhood-based method and the factorization-based method are likely to be retained in the sequential algorithms.

Table 2. Description of the Datasets Used in the Experiments, Including the Numbers of Users ($|\mathcal{U}|$), Items ($|\mathcal{I}|$), Purchases ($|\mathcal{P}|$), and Browses ($|\mathcal{B}|$) in Training Data, and the Numbers of Purchases ($|\mathcal{P}(val.)|$) in Validation Data, and the Number of Purchases ($|\mathcal{P}(te.)|$) in Test Data

Dataset	$\lvert\mathcal{U}\rvert$	$\lvert\mathcal{I}\rvert$	$\lvert\mathcal{P}\rvert$	$\lvert\mathcal{B}\rvert$	$\lvert\mathcal{P}(val.)\rvert$	$\lvert\mathcal{P}(te.)\rvert$	$\lvert\mathcal{P}\rvert : \lvert\mathcal{B}\rvert$
ML10M	71567	10681	309317	4000024	308673	308702	1:12.93
Netflix	480189	17770	4554888	39628846	4556347	4558506	1:8.700
IJCAI-15	28059	32339	408308	1555412	28059	28059	1:3.809

3.1 Datasets and Evaluation Metrics

In our empirical studies, we study the performance of our RoToR on three large datasets, including two simulated HOCCF data MovieLens 10M (ML10M)[1] and Netflix[2] used in Reference [22], and a real-world HOCCF data used in the IJCAI-15 competition.[3] ML10M and Netflix are two well-known datasets in the research community of recommender systems, which contain about 10 million numerical ratings in $\{0.5, 1, 1.5, \ldots, 4.5, 5\}$, and about 0.1 billion scores in $\{1, 2, 3, 4, 5\}$, respectively. For both ML10M and Netflix, we first randomly divide the data into five parts with equal numbers of (u, i, r_{ui}) triples, then take three parts and keep the (u, i) pairs with $r_{ui} = 5$ in one part as purchases for training, that in one part as purchases for validation, and that in another part as purchases for test, and, finally, take the remaining two parts and keep all the (u, i) pairs as browses. We repeat this procedure three times and obtain three copies of (i) purchases for training, (ii) browses for training, (iii) purchases for validation, and (iv) purchases for test. In the IJCAI-15 competition, we have a real-world HOCCF data with 3,292,144 purchasing records and 4,8550,713 browsing records from Tmall.com. In particular, the data is processed as follows: (i) we keep the earliest purchasing record only for each (user, item) pair if it is associated with more than one purchasing record, and obtain a temporary set of purchasing records *target1*; (ii) we keep users with more than 12 purchasing records and items with more than 16 purchasing records from *target1*, and obtain a user set *user1*, an item set *item1*, and a refined set of purchasing records *target2*; (iii) we keep users with more than 12 purchasing records from *target2*, and obtain a refined user set *user2* and a further refined set of purchasing records *target3*; (iv) we keep all items in *target3*, and obtain a refined item set *item2*; (v) we take the most recent purchasing record and the second recent one in *target3* as test data and validation data, respectively, and the remaining earlier purchasing records as training data; (vi) we then keep the browsing records with users in *user2* and items in *item2*, and obtain *auxiliary1*; (vii) we remove browsing records that appear in *target3*, and obtain *auxiliary2*; (viii) we keep the earliest browsing record only for each (user, item) pair if it is associated with more than one browsing record, and obtain *auxiliary3*; and, finally, (ix) we remove any browsing record that is associated with a timestamp larger than that of the purchasing record of the corresponding user in the validation data, and obtain a final set of browsing records *auxiliary4*. Notice that the above two different ways of constructing the training data, validation data, and test data means that we use time-independent evaluation for ML10M and Netflix, and time-dependent evaluation for IJCAI-15 [3]. We put the statistics for the first copy of each dataset in Table 2.

[1] http://grouplens.org/datasets/movielens/10m.
[2] https://www.netflix.com.
[3] https://tianchi.aliyun.com.

For evaluation, we use five commonly used ranking-oriented metrics in information retrieval and item recommendation, including Precision@K, Recall@K, F1@K, NDCG@K, and 1-call@K for the generated top-K ranked list of items.

3.2 Baselines and Parameter Settings

Because HOCCF is a very recently studied recommendation problem, only a few solutions have been proposed. In our empirical studies, we thus include the existing and the proposed solutions for HOCCF, as well as the state-of-the-art methods for OCCF.

- —ICF (item-oriented collaborative filtering) [6] is a classical neighborhood-based recommendation method for OCCF, in which the Jaccard index is used as the similarity measurement for every two items. Notice that ICF with both purchases and browses is also used in the first phase of our sequential RoToR.
- —MF (matrix factorization) [25] can be applied to recommendation with homogeneous one-class feedback (i.e., OCCF) by sampling some unobserved (user, item) pairs as negative feedback [17]. We use two different types of loss functions in MF, i.e., square loss and logistic loss, which are denoted as MF(SquareLoss) and MF(LogisticLoss), respectively. Notice that MF(LogisticLoss) is also used in the second phase of our sequential RoToR with pointwise preference learning.
- —LDA (latent Dirichlet allocation) [2] models the purchasing behaviors in OCCF by taking users as documents and items as words [35]. LDA is also a latent factor model similar to that of matrix factorization, but it is more closer to that of non-negative matrix factorization, because the learned parameters can be interpreted as non-negative probabilities.
- —BPR (Bayesian personalized ranking) [24] is an accurate recommendation algorithm for homogeneous one-class feedback such as purchases in OCCF, which captures users' preferences by assuming that a user prefers a purchased item to an unpurchased one. Notice that BPR is also used in the second phase of our sequential RoToR with pairwise preference learning.
- —FISM (factored item similarity models) [12] aims to improve the predefined similarity in neighborhood-based methods for OCCF such as ICF by learning the similarities among items, which is expected to model the users' one-class feedback more accurately for different data.
- —ABPR (adaptive Bayesian personalized ranking) [21] generalizes BPR [24] from the task of modeling purchases only in OCCF to that with both purchases and browses in HOCCF, where a confidence weight is learned for each browsing behavior, resolving the uncertainty of the users' implicit preferences beneath browses.
- —TJSL (transfer via joint similarity learning) [19] is the state-of-the-art method for modeling heterogeneous one-class feedback in HOCCF, which jointly learns the similarity between a candidate item and a purchased item, and the similarity between a candidate item and a likely-to-purchase item. Notice that TJSL is an extension of FISM [12] from learning one type of similarity to two different types of similarities.
- —RBPR (role-based Bayesian personalized ranking) [22] is a very recent solution for consuming two different types of one-class feedback in HOCCF, where a two-stage re-ranking based approach is designed. In each step of RBPR, the pairwise preference learning algorithm, i.e., BPR, is adopted for preference learning.
- —RoToR (role-based transfer to rank) is proposed in this article, which contains two variants, i.e., an integrative variant called RoToR(int.) and a sequential variant called RoToR(seq.). In RoToR(int.) and RoToR(seq.), the heterogeneous one-class feedback are modeled by the

Table 3. Recommendation Performance of ICF, MF, LDA, BPR, FISM, ABPR, TJSL, RBPR, and our RoToR on Heterogeneous One-Class Feedback Constructed from ML10M Using Prec@5, Rec@5, F1@5, NDCG@5, and 1-call@5

Method	Prec@5	Rec@5	F1@5	NDCG@5	1-call@5
ICF	0.0458±0.0002	0.0598±0.0001	0.0437±0.0003	0.0629±0.0004	0.1948±0.0004
MF(SquareLoss)	0.0533±0.0002	0.0792±0.0003	0.0540±0.0001	0.0742±0.0009	0.2316±0.0010
MF(LogisticLoss)	0.0688±0.0005	0.0963±0.0006	0.0672±0.0006	0.0963±0.0007	0.2881±0.0018
LDA	0.0548±0.0001	0.0657±0.0010	0.0497±0.0002	0.0723±0.0004	0.2290±0.0009
BPR	0.0629±0.0002	0.0855±0.0006	0.0603±0.0003	0.0861±0.0004	0.2648±0.0017
FISM	0.0631±0.0015	0.0917±0.0023	0.0629±0.0016	0.0889±0.0026	0.2699±0.0058
ABPR	0.0657±0.0009	0.0893±0.0017	0.0632±0.0009	0.0905±0.0014	0.2752±0.0039
TJSL	0.0669±0.0006	0.1006±0.0001	0.0679±0.0005	0.0958±0.0002	0.2864±0.0014
RBPR	0.0719±0.0013	0.0977±0.0017	0.0690±0.0014	0.0994±0.0020	0.2990±0.0050
RoToR(pai.,int.)	0.0797±0.0005	0.1117±0.0015	0.0776±0.0007	0.1107±0.0011	0.3295±0.0016
RoToR(pai.,seq.)	0.0762±0.0002	0.1040±0.0005	0.0734±0.0000	0.1081±0.0002	0.3130±0.0019
RoToR(poi.,int.)	**0.0811**±0.0004	**0.1173**±0.0005	**0.0805**±0.0004	**0.1149**±0.0007	**0.3361**±0.0013
RoToR(poi.,seq.)	0.0779±0.0001	0.1066±0.0006	0.0751±0.0002	0.1110±0.0004	0.3192±0.0020

The number of latent dimensions and the number of nearest neighbors are fixed as 20. Notice that the significantly best results are marked in bold (p value < 0.01).

concept of neighborhood-based preference learning and factorization-based preference learning in an integrative manner and in a sequential manner, respectively. RoToR can also be configured with different preference learning paradigms such as pointwise preference learning and pairwise preference learning, resulting in four specific algorithms, including RoToR(poi.,int.), RoToR(pai.,int.), RoToR(poi.,seq.), and RoToR(pai.,seq.).

For MF, BPR, FISM, ABPR, TJSL, RBPR, and our RoToR, we adopt the commonly used SGD method for model training instead of the batch method because of its low efficiency, fix the dimension as $d = 20$ and the learning rate as $\gamma = 0.01$ [22], and initialize the model parameters in the same way with that of RBPR [22]. For BPR and RBPR on ML10M and Netflix, we directly use the results from Reference [22]. For ICF, we set the size of neighborhood as 20. For LDA, we set the number of topics as 20. For MF and FISM, we fix $\rho = 3$ [12] and randomly sample $3|\mathcal{P}|$ (user, item) pairs not included in the purchase data. For each factorization-based algorithm on each dataset, the tradeoff parameters are searched from $\{0.001, 0.01, 0.1\}$ and the iteration number are chosen from $\{100, 500, 1000\}$, using the performance of NDCG@15 on the validation data. Notice that for each of the factorization-based methods except RoToR(poi.,int.) and RoToR(pai.,int), the tradeoff parameters on different regularization terms are associated with the same values. For RoToR(poi.,int.) and RoToR(pai.,int), the tradeoff parameter α_u and other tradeoff parameters (i.e., $\alpha_v, \alpha_w, \beta_u, \beta_v$) are treated separately. In our sequential RoToR, we fix the size of the candidate list of items as $3K = 15$. We report the results of the average and standard deviation on three copies of ML10M and Netflix, and the results of the single copy of IJCAI-15.

3.3 Main Results

We report the main recommendation performance in Tables 3–5, from which we can have the following observations:

Table 4. Recommendation Performance of ICF, MF, LDA, BPR, FISM, ABPR, TJSL, RBPR, and our RoToR on Heterogeneous One-Class Feedback Constructed from Netflix Using Prec@5, Rec@5, F1@5, NDCG@5, and 1-call@5

Method	Prec@5	Rec@5	F1@5	NDCG@5	1-call@5
ICF	0.0800±0.0004	0.0532±0.0002	0.0506±0.0002	0.0927±0.0004	0.3077±0.0011
MF(SquareLoss)	0.0567±0.0001	0.0437±0.0004	0.0388±0.0001	0.0656±0.0003	0.2387±0.0003
MF(LogisticLoss)	0.0732±0.0001	0.0535±0.0002	0.0483±0.0001	0.0848±0.0001	0.2938±0.0008
LDA	0.0662±0.0006	0.0369±0.0002	0.0381±0.0003	0.0736±0.0008	0.2585±0.0021
BPR	0.0716±0.0007	0.0480±0.0005	0.0446±0.0005	0.0818±0.0011	0.2846±0.0022
FISM	0.0687±0.0013	0.0493±0.0017	0.0451±0.0012	0.0789±0.0016	0.2802±0.0044
ABPR	–	–	–	–	–
TJSL	–	–	–	–	–
RBPR	0.0797±0.0002	0.0595±0.0004	0.0527±0.0003	0.0939±0.0003	0.3174±0.0011
RoToR(pai.,int.)	0.0837±0.0001	0.0622±0.0004	0.0552±0.0001	0.0980±0.0003	0.3301±0.0007
RoToR(pai.,seq.)	**0.0918**±0.0003	0.0672±0.0003	0.0602±0.0003	**0.1089**±0.0005	0.3508±0.0013
RoToR(poi.,int.)	0.0837±0.0006	0.0670±0.0005	0.0575±0.0004	0.0993±0.0006	0.3333±0.0016
RoToR(poi.,seq.)	0.0915±0.0003	**0.0679**±0.0004	**0.0605**±0.0004	**0.1089**±0.0005	**0.3511**±0.0014

The number of latent dimensions and the number of nearest neighbors are fixed as 20. Notice that the significantly best results are marked in bold (p value < 0.01), and "–" denotes the case that the training process cannot be finished within 168 hours.

Table 5. Recommendation Performance of ICF, MF, LDA, BPR, FISM, ABPR, TJSL, RBPR, and our RoToR on Heterogeneous One-Class Feedback of IJCAI-15 Dataset Using Prec@5, Rec@5, F1@5, NDCG@5, and 1-call@5

Method	Prec@5	Rec@5	F1@5	NDCG@5	1-call@5
ICF	0.0035	0.0173	0.0058	0.0113	0.0173
MF(SquareLoss)	0.0008	0.0042	0.0014	0.0025	0.0042
MF(LogisticLoss)	0.0012	0.0059	0.0020	0.0035	0.0059
LDA	0.0010	0.0048	0.0016	0.0028	0.0048
BPR	0.0015	0.0076	0.0025	0.0051	0.0076
FISM	0.0015	0.0077	0.0026	0.0047	0.0077
ABPR	0.0017	0.0085	0.0028	0.0053	0.0085
TJSL	0.0017	0.0084	0.0028	0.0054	0.0084
RBPR	0.0020	0.0098	0.0033	0.0062	0.0098
RoToR(pai.,int.)	0.0017	0.0084	0.0028	0.0054	0.0084
RoToR(pai.,seq.)	**0.0051**	**0.0255**	**0.0085**	**0.0163**	**0.0255**
RoToR(poi.,int.)	0.0016	0.0081	0.0027	0.0050	0.0081
RoToR(poi.,seq.)	0.0048	0.0241	0.0080	0.0155	0.0241

The number of latent dimensions and the number of nearest neighbors are fixed as 20. Notice that the best results are marked in bold.

—RoToR performs significantly better (p-value < 0.01 on ML10M and Netflix)[4] than all nine baselines on all five evaluation metrics across the three datasets, which clearly shows the

[4]We calculate the p-value of the statistical significance test (two-sample t-test) via the MATLAB function ttest2.m as described at http://www.mathworks.com/help/stats/ttest2.html.

effectiveness of our integrative and/or sequential modeling mechanisms for heterogeneous one-class feedback.

—In most cases, the recommendation methods exploiting both purchases and browses (i.e., heterogeneous one-class feedback) are better than those making use of purchases only (i.e., homogeneous one-class feedback), which shows the complementarity of those two types of users' feedback.

—For the models exploiting homogeneous one-class feedback, i.e., ICF, MF(SquareLoss), MF(LogisticLoss), LDA, BPR, and FISM, the relative performance ordering is MF(LogisticLoss) > FISM > BPR > MF(SquareLoss) > LDA > ICF on ML10M, ICF > MF(LogisticLoss) > BPR > FISM > LDA > MF(SquareLoss) on Netflix, and ICF > BPR > FISM > MF(LogisticLoss) > LDA > MF(SquareLoss) on IJCAI-15, which shows the effectiveness of the learned similarity in FISM on ML10M and the predefined similarity in ICF on Netflix and IJCAI-15. In particular, we find that the performance of MF(LogisticLoss) and BPR are relatively stable. More interestingly, we can see that the pointwise method MF(LogisticLoss) performs better than the pairwise method BPR in more cases, which demonstrates the effectiveness of a pointwise method with a proper loss function for one-class feedback.

—For the models exploiting heterogeneous one-class feedback such as ABPR, we can see that ABPR is usually better than BPR on ML10M and IJCAI-15, though it is not be able to deliver recommendations within 168 hours on Netflix. For TJSL and FISM, the observation is similar. And the two-stage approach RBPR is usually more accurate and efficient.

—For RoToR(pai.,int.) and RoToR(pai.,seq.) on ML10M and Netflix, their performance are close as expected, but the decomposed one, i.e., RoToR(pai.,seq.), is more flexible and easier for maintenance. As far as we know, RoToR(pai.,seq.) is the first method that decomposes an integrative method for HOCCF. For RoToR(poi.,int.) and RoToR(poi.,seq.) on ML10M and Netflix, the observations are similar. Notice that sequential RoToR performs much better than the corresponding integrative RoToR on IJCAI-15. The reason is that the ratio between the number of purchases and the number of browses in IJCAI-15 is much larger than that in ML10M and Netflix as shown in Table 2, which makes learning on the purchase data in the second phase more reliable.

—For pointwise RoToR and pairwise RoToR, we can see that pairwise RoToR performs a bit worse than pointwise RoToR on ML10M and Netflix, and it performs a bit better than pointwise RoToR on IJCAI-15. This observation is similar to the performance between pairwise recommendation algorithms such as BPR [24] and pointwise recommendation algorithms with logistic loss (instead of square loss) such as MF(LogisticLoss) [11], i.e., their performance are usually close though their preference assumptions are different.

3.4 Results of Top-K Recommendation List

In this subsection, we study the top-K performance with different values of $K \in \{1, 2, 3, 4, 5, 10, 15\}$. Specifically, we choose ICF and MF(LogisticLoss)/BPR as the major baselines for comparative studies with the best performing method on each data, i.e., RoToR(poi.,int.) on ML10M, RoToR(poi.,seq.) on Netflix and RoToR(pai.,seq.) on IJCAI-15, because they are mainly built on those two basic and representative recommendation methods. We report the results on F1@K and NDCG@K in Figures 2–4. Notice that the results on other metrics are similar. From Figures 2–4, we can have the following observations:

—RoToR(poi.,int.) and RoToR(poi.,seq.) perform significantly better than MF(LogisticLoss) and ICF on ML10M and Netflix, which clearly showcases the advantage of combining pointwise method and logistic loss function in modeling one-class feedback.

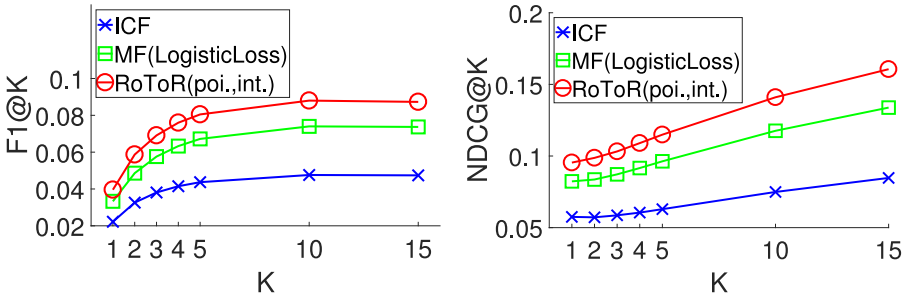

Fig. 2. Top-K recommendation performance with different values of $K \in \{1, 2, 3, 4, 5, 10, 15\}$ on ML10M.

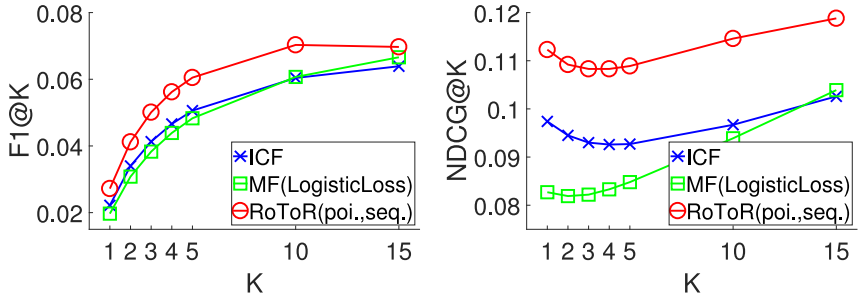

Fig. 3. Top-K recommendation performance with different values of $K \in \{1, 2, 3, 4, 5, 10, 15\}$ on Netflix.

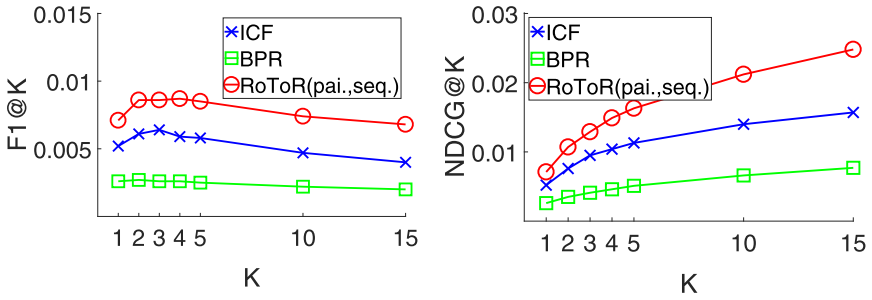

Fig. 4. Top-K recommendation performance with different values of $K \in \{1, 2, 3, 4, 5, 10, 15\}$ on IJCAI-15.

—RoToR(pai.,seq.) performs significantly better than BPR and ICF on IJCAI-15, which again shows the effectiveness of our sequential modeling approach and the merit of our transfer learning method in exploiting their complementarity.
—MF(LogisticLoss) performs better than ICF on ML10M but worse than ICF on Netflix, and BPR performs worse than ICF on IJCAI-15, which showcases another merit of our proposed role-based transfer learning method, i.e., the performance is stable on different datasets as compared with the other two methods.

3.5 Results of Different Components

In this subsection, we study the effectiveness of the decomposed approach, i.e., RoToR(seq.). Specifically, we check the performance of the two components separately to have some insights. The recommendation performances are shown in Table 6. We can see that the second component can improve the results of the first one in most cases. The improvement also echoes the observations

Table 6. Recommendation Performance of the Two Components in RoToR(seq.)

Component	ML10M		Netflix		IJCAI-15	
	F1@5	NDCG@5	F1@5	NDCG@5	F1@5	NDCG@5
Browser in RoToR(pai.,seq.)	0.0642	0.0950	0.0574	0.1070	0.0080	0.0151
Purchaser in RoToR(pai.,seq.)	0.0734	0.1081	0.0602	0.1089	0.0085	0.0163
Improvement	14.33%	13.79%	4.88%	1.78%	6.25%	7.95%
Browser in RoToR(poi.,seq.)	0.0642	0.0950	0.0574	0.1070	0.0080	0.0151
Purchaser in RoToR(poi.,seq.)	0.0751	0.1110	0.0605	0.1089	0.0080	0.0155
Improvement	16.98%	16.84%	5.40%	1.78%	-	2.65%

of the results in Tables 3–5, i.e., the complementarity of the neighborhood-based method and the factorization-based method. The results in Table 6 also show the effectiveness of our proposed coarse-to-fine decomposition mechanism as illustrated in Figure 1.

4 RELATED WORK

In this section, we discuss some related works in two folds, including a revisit of (i) users' preference modeling from the perspective of different roles, and (ii) recommendation algorithms for heterogeneous one-class feedback in HOCCF.

4.1 Different Roles in User Preference Modeling

User preference modeling is one of the central themes in research and practice of personalization and recommendation techniques. Generally, there are at least three problem settings with different types of users' feedback, i.e., rating prediction with graded scores [4, 25, 33], item recommendation with purchases, browses or likes [9, 12, 24, 26, 29], and collaborative filtering with heterogeneous feedback of numerical ratings and implicit examinations [13, 23]. To capture the underlying preferences beneath the user feedback, various techniques have been proposed, including neighborhood-based method [6] and factorization-based method [12, 13, 23, 24, 25]. We find that those different techniques can actually be interpreted in a unified perspective of users' roles. For example, we can categorize the techniques for the aforementioned three recommendation problem settings from the perspective of different users' roles, i.e., *rater* in rating prediction, *purchaser* or *browser* in item recommendation, and *mixer* in collaborative filtering with heterogeneous feedback.

For *rater* in rating prediction, the state-of-the-art methods are matrix factorization methods or latent factorization models [8, 25]. In factorization models, the preference of a rater to an item, i.e., a numerical rating, is assumed to be represented by a user-specific latent feature vector and an item-specific latent feature vector. For *purchaser* or *browser* in item recommendation, the most well-known works include neighborhood-based method and factorization-based method. For instance, in item-oriented neighborhood-based method [6], the items are ranked or recommended based on the aggregated similarity score between the candidate item and the previously preferred items by the purchaser or browser. In factorization-based methods [12, 24], the preference of the purchaser or browser is modeled in a similar way to that for rating prediction but is usually associated with different prediction rules or loss functions. For *mixer* in collaborative filtering with heterogeneous feedback, some representative methods include SVD++ [13] and factorization machine [23]. In

those methods, the role of mixer is usually shown by an expanded prediction rule with a mix of two parts, e.g., one for rater and the other for purchaser or browser.

We can see that the role of RoToR(int.) is a mixer, and that of RoToR(seq.) is a browser and a purchaser. The decomposition-based relationship between the two variants of RoToR also unveils the interesting relationship among mixer, browser, and purchaser.

4.2 Different Recommendation Algorithms for Heterogeneous One-Class Feedback

Heterogeneous one-class feedback arising from users' different online actions such as purchases and browses are very common in different e-commerce systems. Recommendation with HIF [21] or HOCCF [19] is a recently studied problem, which has been attracting more and more attentions [22, 30]. Besides the heterogeneity of the two types of feedback, the main challenges include the scarcity of the purchase data for each individual user and the uncertainty of the users' preferences beneath browsing behaviors. To model the two different types of one-class feedback well, some works formulate the problem as a machine learning problem [30] and others treat it as a transfer learning problem [19, 21].

For addressing the HOCCF problem by machine learning techniques, the feedback data are usually transformed to a new data via feature engineering to feed them to a supervised machine learning algorithm such as gradient boosted decision tree [7] or even deep learning [14]. Once the intermediate results or the confidence weight of each feedback have been obtained, existing pointwise or pairwise one-class factorization-based methods can then be applied [30]. For addressing the HOCCF problem by transfer leaning techniques, a confidence learning method called ABPR [21], a similarity learning method called TJSL [19], and a role-based sequential method called RBPR [22] have been proposed. Notice that RBPR [22] is different from our RoToR, because RoToR is a principled solution with roles of mixer, browser and purchaser.

There is another line of research, i.e., meta path [27], for modeling heterogeneous information. Meta path [27] is a powerful framework for estimating similarity semantics or extracting ranking features from attribute-rich heterogeneous information networks (HIN) such as DBLP-based article citation data and IMDb-augmented movie rating data, which has been successfully applied to citation recommendation [15] and movie recommendation [34]. Notice that using some different similarity measurements or ranking features from rich attributes in the first or second phase of our RoToR framework is a vertical research direction to our focus in this article.

As compared with the above works regarding different roles in preference learning and different recommendation algorithms for heterogeneous one-class feedback, our RoToR is the first principled role-based transfer learning solution for the HOCCF problem.

5 CONCLUSIONS AND FUTURE WORK

In this article, we propose a novel transfer learning solution for an important recommendation problem called HOCCF. Specifically, we design a novel role-based preference learning framework, i.e., RoToR, which contains an integrative variant RoToR(int.) and a sequential variant RoToR(seq.). Each variant can be further configured with pointwise or pairwise preference learning paradigm. Extensive empirical studies on three large datasets show that our RoToR is significantly more accurate than the state-of-the-art methods for either OCCF or HOCCF. Furthermore, we showcase the effectiveness of decomposing the sophisticated recommendation model, i.e., RoToR(int.), into a simple and flexible one, i.e., RoToR(seq.), without sacrificing the accuracy, where the latter is more likely to be preferred by practitioners considering its simplicity in implementation, deployment and maintenance.

For future work, we are interested in (i) generalizing our RoToR to include more types of users' roles in a real online system such as searcher [1], reviewer [28], follower, and followee [36];

(ii) studying the effect of our decomposition mechanism for other advanced recommendation methods in heterogeneous problem settings with linked open data [16] or social media data [10]; (iii) designing some novel listwise learning to rank algorithms for modeling one-class behaviors; and (iv) modeling the sequential relationship in one-class behaviors.

REFERENCES

[1] Xiao Bai, Ioannis Arapakis, Berkant Barla Cambazoglu, and Ana Freire. 2017. Understanding and leveraging the impact of response latency on user behaviour in web search. *ACM Transactions on Information Systems* 36, 2 (2017), 21:1–21:42.

[2] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet allocation. *Journal of Machine Learning Research* 3 (2003), 993–1022.

[3] Pedro G. Campos, Fernando Díez, and Iván Cantador. 2014. Time-aware recommender systems: A comprehensive survey and analysis of existing evaluation protocols. *User Modeling and User-Adapted Interaction* 24, 1–2 (2014), 67–119.

[4] Da Cao, Xiangnan He, Liqiang Nie, Xiaochi Wei, Xia Hu, Shunxiang Wu, and Tat-Seng Chua. 2017. Cross-platform app recommendation by jointly modeling ratings and texts. *ACM Transactions on Information Systems* 35, 4 (2017), 37:1–37:27.

[5] Wenyuan Dai, Qiang Yang, Gui-Rong Xue, and Yong Yu. 2007. Boosting for transfer learning. In *Proceedings of the 24th International Conference on Machine Learning (ICML'07)*. ACM, 193–200. DOI: https://doi.org/10.1145/1273496.1273521

[6] Mukund Deshpande and George Karypis. 2004. Item-based Top-N recommendation algorithms. *ACM Transactions on Information Systems* 22, 1 (2004), 143–177.

[7] Jerome H. Friedman. 2000. Greedy function approximation: A gradient boosting machine. *Annals of Statistics* 29 (2000), 1189–1232.

[8] Guibing Guo, Jie Zhang, and Neil Yorke-Smith. 2016. A novel recommendation model regularized with user trust and item ratings. *IEEE Transactions on Knowledge and Data Engineering* 28, 7 (2016), 1607–1620.

[9] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proceedings of the 26th International Conference on World Wide Web (WWW'17)*. ACM, 173–182.

[10] Shuhui Jiang, Xueming Qian, Tao Mei, and Yun Fu. 2016. Personalized travel sequence recommendation on multi-source big social media. *IEEE Transactions on Big Data* 2, 1 (2016), 43–56.

[11] Christopher C. Johnson. 2014. Logistic matrix factorization for implicit feedback data. In *Proceedings of the Workshop on Distributed Machine Learning and Matrix Computations at NIPS 2014*. http://stanford.edu/~rezab/nips2014workshop/submits/logmat.pdf.

[12] Santosh Kabbur, Xia Ning, and George Karypis. 2013. FISM: Factored item similarity models for Top-N recommender systems. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'13)*. ACM, 659–667.

[13] Yehuda Koren. 2008. Factorization meets the neighborhood: A multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'08)*. ACM, 426–434.

[14] Yann Lecun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *Nature* 521 (5 2015), 436–444.

[15] Xiaozhong Liu, Yingying Yu, Chun Guo, and Yizhou Sun. 2014. Meta-path-based ranking with pseudo relevance feedback on heterogeneous graph for citation recommendation. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management (CIKM'14)*. ACM, 121–130.

[16] Tommaso Di Noia, Vito Claudio Ostuni, Paolo Tomeo, and Eugenio Di Sciascio. 2016. SPrank: Semantic path-based ranking for Top-N recommendations using linked open data. *ACM Transactions on Intelligent Systems and Technology* 8, 1 (2016), 9:1–9:34.

[17] Rong Pan, Yunhong Zhou, Bin Cao, Nathan N. Liu, Rajan Lukose, Martin Scholz, and Qiang Yang. 2008. One-class collaborative filtering. In *Proceedings of the 8th IEEE International Conference on Data Mining (ICDM'08)*. IEEE, 502–511.

[18] Sinno Jialin Pan and Qiang Yang. 2010. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering* 22, 10 (2010), 1345–1359.

[19] Weike Pan, Mengsi Liu, and Zhong Ming. 2016. Transfer learning for heterogeneous one-class collaborative filtering. *IEEE Intelligent Systems* 31, 4 (2016), 43–49. DOI: https://doi.org/10.1109/MIS.2016.19

[20] Weike Pan, Qiang Yang, Yuchao Duan, Ben Tan, and Zhong Ming. 2017. Transfer learning for behavior ranking. *ACM Transactions on Intelligent Systems and Technology* 8, 5 (2017), 65:1–65:23.

[21] Weike Pan, Hao Zhong, Congfu Xu, and Zhong Ming. 2015. Adaptive Bayesian personalized ranking for heterogeneous implicit feedbacks. *Knowledge-Based Systems* 73 (2015), 173–180.

[22] Xiaogang Peng, Yaofeng Chen, Yuchao Duan, Weike Pan, and Zhong Ming. 2016. RBPR: Role-based Bayesian personalized ranking for heterogeneous one-class collaborative filtering. In *Late-Breaking Results, Posters, Demos, Doctoral Consortium and Workshops Proceedings of the 24th ACM Conference on User Modeling, Adaptation and Personalisation (UMAP'16)*, Federica Cena, Michel C. Desmarais, and Darina Dicheva (Eds.). Vol. 1618. CEUR-WS.org.

[23] Steffen Rendle. 2012. Factorization machines with libFM. *ACM Transactions on Intelligent Systems and Technology* 3, 3 (2012), 57:1–57:22.

[24] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence (UAI'09)*. AUAI Press, 452–461.

[25] Ruslan Salakhutdinov and Andriy Mnih. 2008. Probabilistic matrix factorization. In *Proceedings of the Annual Conference on Neural Information Processing Systems (NIPS'08)*. Curran Associates, Inc., 1257–1264.

[26] Lei Shi, Wayne Xin Zhao, and Yi-Dong Shen. 2017. Local representative-based matrix factorization for cold-start recommendation. *ACM Transactions on Information Systems* 36, 2 (2017), 22:1–22:28.

[27] Yizhou Sun, Jiawei Han, Xifeng Yan, Philip S. Yu, and Tianyi Wu. 2011. PathSim: Meta path-based Top-K similarity search in heterogeneous information networks. In *Proceedings of the VLDB Endowment (PVLDB'11)* 4, 11 (2011), 992–1003.

[28] Yunzhi Tan, Min Zhang, Yiqun Liu, and Shaoping Ma. 2016. Rating-boosted latent topics: Understanding users and items with ratings and reviews. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence 2016*. IJCAI/AAAI Press, 2640–2646.

[29] João Vinagre, Alípio Mário Jorge, and João Gama. 2014. Fast incremental matrix factorization for recommendation with positive-only feedback. In *Proceedings of the 22nd International Conference on User Modeling, Adaptation, and Personalization*. Springer, 459–470.

[30] Jing Wang, Lanfen Lin, Heng Zhang, and Jiaqi Tu. 2016. Confidence-learning based collaborative filtering with heterogeneous implicit feedbacks. In *Proceedings of the 18th Asia-Pacific Web Conference (APWeb'16)*. Springer, 444–455.

[31] Markus Weimer, Alexandros Karatzoglou, Quoc Viet Le, and Alex Smola. 2007. COFIRANK maximum margin matrix factorization for collaborative ranking. In *Proceedings of the 20th International Conference on Neural Information Processing Systems (NIPS'07)*. Curran Associates, Inc., 1593–1600.

[32] Fen Xia, Tie-Yan Liu, Jue Wang, Wensheng Zhang, and Hang Li. 2008. Listwise approach to learning to rank: Theory and algorithm. In *Proceedings of the 25th International Conference on Machine Learning (ICML'08)*. ACM, 1192–1199.

[33] Yan Yan, Mingkui Tan, Ivor W. Tsang, Yi Yang, Chengqi Zhang, and Qinfeng Shi. 2015. Scalable maximum margin matrix factorization by active riemannian subspace search. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence*. AAAI Press, 3988–3994.

[34] Xiao Yu, Xiang Ren, Yizhou Sun, Quanquan Gu, Bradley Sturt, Urvashi Khandelwal, Brandon Norick, and Jiawei Han. 2014. Personalized entity recommendation: A heterogeneous information network approach. In *Proceedings of the 7th ACM International Conference on Web Search and Data Mining (WSDM'14)*. ACM, 283–292.

[35] Haijun Zhang, Zhoujun Li, Yan Chen, Xiaoming Zhang, and Senzhang Wang. 2014. Exploit latent Dirichlet allocation for one-class collaborative filtering. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management (CIKM'14)*. ACM, 1991–1994.

[36] Wayne Xin Zhao, Sui Li, Yulan He, Edward Y. Chang, Ji-Rong Wen, and Xiaoming Li. 2016. Connecting social media to E-commerce: Cold-start product recommendation using microblogging information. *IEEE Transactions on Knowledge and Data Engineering* 28, 5 (2016), 1147–1159.

[37] Yu Zheng. 2015. Methodologies for cross-domain data fusion: An overview. *IEEE Transactions on Big Data* 1, 1 (2015), 16–33.

Received December 2017; revised May 2018; accepted July 2018