Partial Relaxed Optimal Transport for Denoised Recommendation

Yanchao Tan[1,2], Carl Yang[3,*], Xiangyu Wei[4], Ziyue Wu[5], Weiming Liu[4,†] and Xiaolin Zheng[4]

[1]The College of Computer and Data Science, Fuzhou University, Fuzhou 350116, China
[2]Fujian Key Laboratory of Network Computing and Intelligent Information Processing, Fuzhou University, Fuzhou 350116, China
[3]The Department of Computer Science, Emory University, Atlanta 30322, United State
[4]The College of Computer Science, Zhejiang University, Hangzhou 310027, China
[5]The School of Management, Zhejiang University, Hangzhou 310058, China

Abstract

The interaction data used by recommender systems (RSs) inevitably include noises resulting from mistaken or exploratory clicks, especially under implicit feedbacks. Without proper denoising, RS models cannot effectively capture users' intrinsic preferences and the true interactions between users and items. To address such noises, existing methods mostly rely on auxiliary data which are not always available. In this work, we ground on Optimal Transport (OT) to globally match a user embedding space and an item embedding space, allowing both non-deep and deep RS models to discriminate intrinsic and noisy interactions without supervision. Specifically, we firstly leverage the OT framework via Sinkhorn distance to compute the continuous many-to-many user-item matching scores. Then, we relax the regularization in Sinkhorn distance to achieve a closed-form solution with a reduced time complexity. Finally, to consider individual user behaviors for denoising, we develop a partial OT framework to adaptively relabel user-item interactions through a personalized thresholding mechanism. Extensive experiments show that our framework can significantly boost the performances of existing RS models.

1. Introduction

With the rapid growth of various activities on the Web, recommender systems (RSs) become fundamental in helping users alleviate the problem of information overload. However, users can click some items by mistake or out of curiosity, and many RSs will also recommend some less relevant items for exploration every now and then. Take movie watching for example. Since a user cannot always distinguish horrible movies from romantic ones by movie names, he/she can easily click a horrible movie by mistake among the many romantic movies he/she usually watches. In this case, a traditional method like CF cannot effectively reduce the probability of recommending a horrible movie unless the noisy clicks are extremely rare, which may further seduce even more noisy clicks.

Recently, to get out of such mistakes, existing research discovers users' intrinsic preferences with the help of external user behaviors [1], auxiliary item features [2] or extra feedbacks [3]. A key limitation is their reliance on auxiliary data, which costs significant effort and is not always attainable in RSs. Without supervision, can we develop a framework to automatically denoise user-item interactions for more accurate recommendations?

In this work, we approach the denoised recommendation problem in two steps: 1. distinguishing intrinsic and noisy interactions; 2. stressing intrinsic preferences and reducing noisy ones for recommendation. Inspired by a least modeling effort principle in an unsupervised fashion [4, 5], we refer to noises as *minor abnormal data that needs higher modeling effort* than the intrinsic ones.

Based on the above rationale, we propose to distinguish noises by ranking the global matching cost between the user and item embedding spaces, and flexibly integrate it with both non-deep and deep RS methods. In this way, the key to distinguishing noises boils down to finding a global matching matrix with the minimum matching cost. Subsequently, we advocate a principled denoised RS framework and calculate the matching matrix grounding on *Optimal Transport* (OT), which has been introduced to address the unsupervised matching problem between two distributions or spaces in many fields [6, 7]. Through minimizing the overall cost of the mismatched user-item pairs, OT finds a global optimal matching matrix, whose values represent the matching cost (*i.e.*, modeling effort). Specifically, *the low user-item matching cost indicates a user's intrinsic preferences while the high value indicates noisy ones.*

However, to apply OT to achieve denoised recommendation, three problems still remain: 1. Unlike the one-hot constraint in previous applications of OT, the nature of RSs requires a many-to-many matching; 2. The large number of users and items makes the matching process time-consuming; 3. A mechanism is needed to properly

DL4SR'22: Workshop on Deep Learning for Search and Recommendation, co-located with the 31st ACM International Conference on Information and Knowledge Management (CIKM), October 17-21, 2022, Atlanta, USA
*Corresponding author.
yctan@fzu.edu.cn (Y. Tan); j.carlyang@emory.edu (C. Yang); weixy@zju.edu.cn (X. Wei); wuziyue@zju.edu.cn (Z. Wu); 21831010@zju.edu.cn (W. Liu); xlzheng@zju.edu.cn (X. Zheng)

leverage the learned matching matrix and finally eliminate the effect of noises for denoised recommendation.

To address these challenges, we first leverage the OT framework via Sinkhorn distance [8]. Unlike a one-hot matching matrix that satisfies the requirements of the many widely studied tasks (*e.g.*, transfer learning [6, 9]), we compute a continuous many-to-many approximation to the discrete OT, so as to meet the nature of RSs. Furthermore, we propose to relax the regularization in Sinkhorn to achieve a closed-form solution, which can reduce the time complexity for large-scale data from $\mathcal{O}(\max(M,N)^3)$ to $\mathcal{O}(\max(M,N)^2)$. Then, we rank the learned continuous matching cost to differentiate intrinsic and noisy interactions for each user. Finally, we take into account that individual users' behaviors can vary in RSs, which should be handled differently during denoising. To this end, we design a personalized thresholding mechanism for relabeling user-item interactions, which efficiently finds a splitting point for each individual user according to the ranked matching cost.

Experiment results on one real-world dataset with synthesized noises demonstrate our proposed ProRec's ability of denoising under various levels of noises. Further experiments on three original real-world datasets also verify the advantages of ProRec, which demonstrate its significant performance boosts upon multiple popular RS models, in terms of effectiveness and efficiency.

2. Related work

Many research studies have pointed out that there exist a large proportion of noisy interactions in the implicit feedbacks (*i.e.*, the clicks) [10, 1, 11]. These feedbacks are easily affected by different factors, such as the position bias [12], caption bias [13] and exposure bias [10, 14]. Therefore, there exist large gaps between the implicit feedbacks and the actual user preferences due to various reasons, which are detrimental to the users' continuous behaviors and overall satisfaction [2, 5].

To alleviate the above issues, many researchers have considered incorporating auxiliary feedbacks to the identification of noises in the implicit feedbacks, such as dwell time [1] and scroll intervals [13]. These works usually design features manually and label items with external help (*e.g.*, from domain experts) [13, 2], which require extensive effort and cost, which are not always available across RSs. To this end, sheer observations on training losses have recently been explored to denoise the implicit feedbacks during training in an unsupervised fashion [5]. The method is purely heuristic and can be integrated with deep learning methods only. Based on a principle of the least modeling effort in different fields [4, 15], our proposed framework aims to denoise the user-item interactions without supervision, which is supported by Optimal transport (OT) theory with guaranteed optimality in the global matching and can be flexibly integrated with both deep and non-deep RS methods.

OT aims to find a matching between two distributions or spaces, which has been most widely used in transfer learning. It can match one instance in the source domain to another in the target domain with the least cost. Existing works mainly study discrete OT for one-to-one matching. For example, in computer vision, OT has been applied to domain adaption [4, 6] and style transfer [16]. OT has also been successfully applied in many natural language processing tasks, such as topic modeling [7, 17], text generation [18], and sequence-to-sequence learning [19]. Considering the effect of non-matched pairs, [9] proposed a partial one-to-one matching that leverages one shared threshold to address domain shift for domain adaption instead of taking all pairs into account.

However, the above discrete one-to-one matching cannot satisfy the nature of many-to-many user-item relationships in RSs, where each user can interact with many items and each item can interact with many users. Though OT is applied into cold-start recommendation problem in [20], they directly measure user-item similarity without taking the effect of noises into consideration. Moreover, partial matching with a shared threshold cannot reflect the individual user behaviors, and thus leads to suboptimal recommendation performances.

3. Methodology

In this section, we detail the proposed Partial relaxed optimal Transport for the denoised Recommendation (ProRec) framework, as shown in Figure 1. Specifically, we first formulate the denoised recommendation as a two-step process. Then, we introduce the Optimal Transport (OT) framework for distinguishing noises through global user-item matching. Furthermore, we elaborate on our proposed ProRec framework, which can be applied to achieve denoised recommendations.

3.1. Problem Statement

Following the setting of recommendation, we denote the user-item interaction matrix as $\boldsymbol{X} \in \{0,1\}^{M\times N}$, where M denotes the number of users and N denotes the number of items. We construct user-item relationships across a *user embedding space* $\mathcal{D}^u$ and an *item embedding space* $\mathcal{D}^v$. The problem is that user-item pairs contain some noisy interactions which need to be relabeled.

In this work, we approach denoised recommendation through two steps. Firstly, we distinguish intrinsic and noisy interactions across $\mathcal{D}^u$ and $\mathcal{D}^v$ by a global matching matrix $\boldsymbol{\pi}^*$. Then, to stress users' intrinsic preferences and reduce noises, we rank the learned $\boldsymbol{\pi}^*$ and keep only

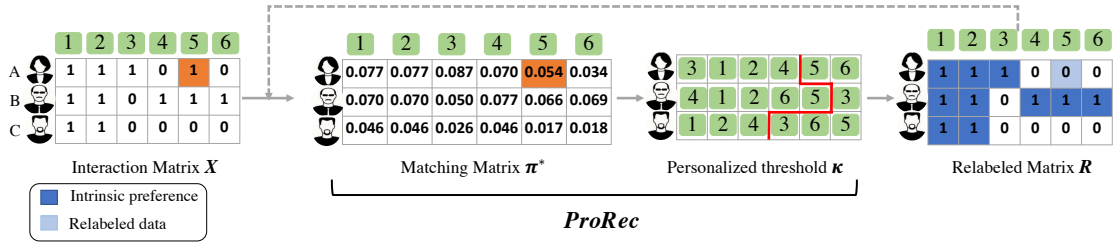


Figure 1: A toy example illustrating the denoised recommendation based on our ProRec framework.

the top κ interactions of each user via a personalized thresholding mechanism. In this way, we learn a relabeled matrix $\boldsymbol{R} \in R^{M \times N}$ for denoised recommendation.

3.2. Global Matching for Distinguishing Noises

As motivated in Section 1, a framework that can denoise user-item interactions for more accurate recommendation is in great need. However, without supervision, it is hard to define noisy interactions and then stress intrinsic preferences together while reducing noisy ones.

To distinguish noises in an unsupervised fashion, several works [4, 15, 21, 5] have pointed out a principled way to locate noises by finding the data that needs the most modeling effort or cost. Inspired by this principle of least modeling effort, in this work, we propose to refer to the noises as minor abnormal data. Compared with the interactions from intrinsic preferences, noises take much effort or cost to model.

Based on the above least effort rationale, we propose a novel denoising framework to distinguish noises by ranking the global matching between user and item embedding spaces. The framework is served as a plug-in, so as to be flexibly integrated to both non-deep and deep RS methods. In this scenario, the key to distinguishing noises boils down to finding a global user-item matching across the user space $\mathcal{D}^u$ and the item space $\mathcal{D}^v$ with minimal matching cost.

To address the unsupervised matching problem between two spaces, we advocate a principled denoised RS framework and ground on *Optimal Transport* (OT), which has been successfully applied in many fields such as CV [6] and NLP [7]. As studied theoretically in [22], the OT formulation derived from the Earth Movers Distance (EMD) can be expressed as

$$\begin{aligned} &\min_{\boldsymbol{\pi} \in \mathcal{X}(\mathcal{D}^u, \mathcal{D}^v)} \sum_{i=1}^{M} \sum_{j=1}^{N} \mathcal{M}_{ij} \pi_{ij}, \\ &s.t.\ \mathbf{1}_M \boldsymbol{\pi} = \frac{1}{N} \mathbf{1}_N, \mathbf{1}_N \boldsymbol{\pi}^T = \frac{1}{M} \mathbf{1}_M, \end{aligned} \tag{1}$$

where $\mathcal{X}(\mathcal{D}^u, \mathcal{D}^v)$ denotes the joint probability distribution of user embedding space $\mathcal{D}^u$ and item embedding space $\mathcal{D}^v$. $\mathbf{1}_M \in \mathbb{R}^{1 \times M}, \mathbf{1}_N \in \mathbb{R}^{1 \times N}$ with all elements equal to 1. M is the number of users and N is the number of items. $\boldsymbol{\pi}^*$ denote the optimal transport plan.

Specifically, the matching cost matrix $\mathcal{M}$ directly reflects the modeling effort, and thus the low user-item matching cost can indicate users' intrinsic preferences while the high values can indicate noises. Based on the definition of $\mathcal{M}$, we can integrate the denoising process with both non-deep and deep RS models as follows:

$$\mathcal{L}(\boldsymbol{U}, \boldsymbol{V}) = ||\boldsymbol{X} - \boldsymbol{U}\boldsymbol{V}^T||_F^2 + \zeta(||\boldsymbol{U}||_2^2 + ||\boldsymbol{V}||_2^2), \tag{2}$$

$$\mathcal{L}(\boldsymbol{U}, \boldsymbol{V}) = \mathcal{C}(\boldsymbol{X}, G_y(\boldsymbol{U}, \boldsymbol{V})) + \zeta(||\boldsymbol{U}||_2^2 + ||\boldsymbol{V}||_2^2), \tag{3}$$

where $\boldsymbol{U} \in \mathbb{R}^{M \times d}$ and $\boldsymbol{V} \in \mathbb{R}^{N \times d}$ represent the user and item embeddings. G_y denotes a classification network. The hyperparameter ζ is to balance the loss and the regularization term. In non-deep learning models, Eq. 2 can be solved effectively through many methods (*e.g.*, Alternative Least Square). Since the prediction scores represent the user-item similarities where $\mathcal{M}$ has negative values, we calculate the cost matrix as $\mathcal{M} = -\boldsymbol{U}\boldsymbol{V}^T$ by using inner product. In deep learning models, we can optimize Eq. 3 by adopting the widely used binary cross-entropy loss (*i.e.*, $\mathcal{C}(\cdot, \cdot)$). Then, similar calculations can be made for those deep learning ones, where $\mathcal{M} = -G_y(\boldsymbol{U}, \boldsymbol{V})$.

However, there are three technical challenges when applying OT to denoised recommendation:

Challenge 1. In RSs, each user can interact with many items and each item interact with many users, which corresponds to a many-to-many interaction matrix. Unlike previous applications of OT in CV or NLP, directly applying the one-hot constraint specified in Eq. 1 is counter-intuitive for recommendation.

Challenge 2. In RSs, the number of users and items in the real world tends to be large, making the matching process time-consuming.

Challenge 3. Besides distinguishing noises, it is necessary to design a mechanism to eliminate the effect of noisy interactions. A naïve way is to set one global thresholding hyperparameter κ to keep the top κ items with the least matching cost for each user. However, the global thresholding hyperparameter cannot accommodate individual user behaviors.

3.3. The ProRec Framework

In this section, we propose a Partial relaxed optimal Transport for denoised Recommendation (ProRec) framework to address the above three challenges.

Many-to-many matching to meet the nature of RSs. To address Challenge 1, we look for a smoother version of OT by lowering the sparsity and increasing the entropy of the matching matrix [4]. Among the many OT algorithms (*e.g.*, ADMM and Proximal Point Method [23, 24]), we choose the Sinkhorm algorithm [8] due to its simplicity and efficiency. We achieve OT with a many-to-many matching through the Sinkhorn algorithm as follows:

$$\begin{aligned} &\min_{\boldsymbol{\pi}\in\mathcal{X}(\mathcal{D}^u,\mathcal{D}^v)} \sum_{i=1}^{M}\sum_{j=1}^{N}\mathcal{M}_{ij}\pi_{ij} + \gamma H(\boldsymbol{\pi}) \\ &s.t., \mathbf{1}_M\boldsymbol{\pi} = \boldsymbol{p}, \mathbf{1}_N\boldsymbol{\pi}^T = \boldsymbol{q}, \end{aligned} \tag{4}$$

where $H(\boldsymbol{\pi}) = \sum_{i=1}^{M}\sum_{j=1}^{N}\pi_{ij}\left(\log(\pi_{ij})-1\right)$. $\mathbf{1}_M \in \mathbb{R}^{1\times M}$, $\mathbf{1}_N \in \mathbb{R}^{1\times N}$ are with all elements equal to 1. γ is a hyperparameter to balance the entropy regularization and OT.

Compared with most studies of discrete OT in CV, the major advantages of the Sinkhorn algorithm above are as follows: i) By relaxing the discrete constraint, we can obtain a continuous matrix instead of a one-hot one. In this way, the obtained matching matrix can capture the many-to-many relationships between users and items. ii) By adding an entropy regularization, we address the sparsity problem of $\boldsymbol{\pi}^*$. The dense cost matrix can be used for subsequent ranking. iii) By modifying the distribution constraint of $\boldsymbol{\pi}$ from uniform (*i.e.*, $\mathbf{1}_M\boldsymbol{\pi} = \frac{1}{N}\mathbf{1}_N$) to popularity-based (*i.e.*, $\mathbf{1}_M\boldsymbol{\pi} = \boldsymbol{p}$), we model non-uniform distributions where users and items have different number of interactions, and successfully adapt the optimization to the recommendation scenario. Specifically, we count the interactions and then normalize them to obtain $\boldsymbol{p}_i$ and $\boldsymbol{q}_j$, so $\sum_{i=1}^{N}\boldsymbol{p}_i = 1$ and $\sum_{j=1}^{M}\boldsymbol{q}_j = 1$.

Relaxed OT to reduce the time complexity. Note that the optimal $\boldsymbol{\pi}^*$ is dense and needs to be computed through multiple iteractions. Therefore, it is time-consuming in RSs with large numbers of users and items. Inspired by the Relaxed OT for EMD [17], we extend the original Sinkhorn algorithm with a relaxed regularization. Specifically, we use two auxiliary distances, each essentially is the Sinkhorn with only one of the constraints in Eq. 4:

$$\min_{\boldsymbol{\pi}\in\mathcal{X}(\mathcal{D}^u,\mathcal{D}^v)} \sum_{i=1}^{M}\sum_{j=1}^{N}\mathcal{M}_{ij}\pi_{ij} + \gamma H(\boldsymbol{\pi})\ s.t.\ \mathbf{1}_M\boldsymbol{\pi} = \boldsymbol{p}, \tag{5}$$

$$\min_{\boldsymbol{\pi}\in\mathcal{X}(\mathcal{D}^u,\mathcal{D}^v)} \sum_{i=1}^{M}\sum_{j=1}^{N}\mathcal{M}_{ij}\pi_{ij} + \gamma H(\boldsymbol{\pi})\ s.t.\ \mathbf{1}_N\boldsymbol{\pi}^T = \boldsymbol{q}. \tag{6}$$

Proposition 1. *The constraint of problem in Eq. 4 is relaxed by Eq.5 and Eq. 6. By defining $\tilde{\boldsymbol{\mathcal{M}}} = e^{-\frac{\mathcal{M}}{\gamma}}$, the closed-form solution for the global optimal matching matrix is:*

$$\boldsymbol{\pi}^* = \max(\tilde{\mathcal{M}}\,\mathrm{diag}(\boldsymbol{p}\oslash\mathbf{1}_M\tilde{\mathcal{M}}), \mathrm{diag}(\boldsymbol{q}\oslash\mathbf{1}_N\tilde{\mathcal{M}}^T)\tilde{\mathcal{M}}), \tag{7}$$

where $\oslash$ corresponds to the element-wise division.

Proof. Based on Eq. 6 and using the Lagrange multiplier method we have:

$$\mathcal{L} = \sum_{i=1}^{M}\sum_{j=1}^{N}\mathcal{M}_{ij}\boldsymbol{\pi}_{ij} + \gamma H(\boldsymbol{\pi}) - (\mathbf{1}_N\boldsymbol{\pi}^T - \boldsymbol{q})\boldsymbol{f}^T, \tag{8}$$

where $\boldsymbol{f}^T$ is Lagrangian multipliers.

Then, in order to find the optimal solution, we let the gradient $\frac{\partial\mathcal{L}}{\partial\boldsymbol{\pi}_{ij}} = 0$, which means:

$$\frac{\partial\mathcal{L}}{\partial\boldsymbol{\pi}_{ij}} = \mathcal{M}_{ij} + \gamma\log\boldsymbol{\pi}_{ij} - \boldsymbol{f}_i = 0, \tag{9}$$

In this case, we can obtain the solution of $\boldsymbol{\pi}_{ij}$ as

$$\boldsymbol{\pi}_{ij} = e^{\frac{f_i - \mathcal{M}_{ij}}{\gamma}}. \tag{10}$$

The users and items that have different interactions, which means both users and items are based on different degrees of popularity. To take such popularity into consideration, we count and norm the number of items that have interacted with user i. Based on the popularity of user i (*i.e.*, $\boldsymbol{q}_i$), we add constraints of the matching matrix π by $\sum_{j=1}^{N}\boldsymbol{\pi}_{ij} = \boldsymbol{q}_i$. Then, we have

$$\sum_{j=1}^{N} e^{\frac{\boldsymbol{f}_i - \mathcal{M}_{ij}}{\gamma}} = \boldsymbol{q}_i. \tag{11}$$

Since the Lagrange multiplier $\boldsymbol{f}_i$ is not related to j, we can extract this part and rewrite the formula as:

$$\frac{1}{\boldsymbol{q}_i}\sum_{j=1}^{N} e^{-\frac{\mathcal{M}_{ij}}{\gamma}} = e^{-\frac{\boldsymbol{f}_i}{\gamma}}. \tag{12}$$

Take the logarithm of both sides of the above equation and move γ to the other side:

$$f_i = -\gamma\log(\frac{1}{\boldsymbol{q}_i}\sum_{j} e^{-\frac{\mathcal{M}_{ij}}{\gamma}}). \tag{13}$$

To keep the formula simple and clear, we define $\tilde{\boldsymbol{\mathcal{M}}} = e^{-\frac{\mathcal{M}}{\gamma}}$, so the formula above can be written as:

$$\boldsymbol{f} = \gamma\log\boldsymbol{q} - \gamma\log(\mathbf{1}_N\tilde{\boldsymbol{\mathcal{M}}}^T). \tag{14}$$

According to Eq. 10 and Eq. 12, we have

$$\boldsymbol{\pi}_{ij} = e^{\frac{\boldsymbol{f}_i - \mathcal{M}_{ij}}{\gamma}} = \boldsymbol{q}_i\frac{e^{-\frac{\mathcal{M}_{ij}}{\gamma}}}{\sum_j e^{-\frac{\mathcal{M}_{ij}}{\gamma}}}. \tag{15}$$

Finally, based on Eq. 15 and the definition of $\tilde{\boldsymbol{\mathcal{M}}} = e^{-\frac{\mathcal{M}}{\gamma}}$, we obtain the closed-form solution via a matrix form as follow

$$\boldsymbol{\pi}_V = \mathrm{diag}(\boldsymbol{q}\oslash\mathbf{1}_N\tilde{\mathcal{M}}^T)\tilde{\mathcal{M}}, \tag{16}$$

where $\oslash$ corresponds to element-wise division.

Following the same derivation, we can obtain the closed-form solution of Eq. 5 as follow

$$\boldsymbol{\pi}_U = \tilde{\mathcal{M}}\,\mathrm{diag}(\boldsymbol{p} \oslash \mathbf{1}_M \tilde{\mathcal{M}}). \tag{17}$$

The original Sinkhorn minimizes π under two constraints. By relaxing one constraint, we can balance the computation efficiency and improved preformance. To approximate the solution under the original constraints, we adopt the max operation, which is consistent with the results in previous work [16]. So we have

$$\boldsymbol{\pi}^* = \max(\boldsymbol{\pi}_U, \boldsymbol{\pi}_V) \tag{18}$$

This is equivalent to:

$$\boldsymbol{\pi}^* = \max(\tilde{\mathcal{M}}\,\mathrm{diag}(\boldsymbol{p} \oslash \mathbf{1}_M \tilde{\mathcal{M}}), \mathrm{diag}(\boldsymbol{q} \oslash \mathbf{1}_N \tilde{\mathcal{M}}^T)\tilde{\mathcal{M}}). \tag{19}$$

□

Note that, the closed-form solution yields a time complexity of $O(\max(M, N)^2)$. This shows that by relaxing the OT formulation with auxiliary limitation from only one side, our model can achieve a lower time complexity than the $O(\max(M, N)^3)$ reported in [6], which uses OT for one-to-one matching.

Personalized thresholding mechanism for denoised recommendation. To flexibly discriminate the intrinsic and noisy user-item interactions for individual users, we propose a non-parametric personalized thresholding mechanism. We first normalize the matching cost by each user as ${\pi_{ij}}^{*\prime} = \frac{\pi^*_{ij}}{\sum_j \pi^*_{ij}}$, and rank them as $\boldsymbol{\rho_i} = \mathrm{sort}(\boldsymbol{\pi}^{*\prime}_i)$. According to the definition of the matching matrix, each row represents users' preferences towards the items. We define $\boldsymbol{\kappa} = \{\kappa_1, \ldots, \kappa_M\}$ as the index of the threshold which can filter out users' noisy preferences. κ_i denotes user i's threshold. As shown in Figure 1, some users with consistent (narrow) interests (*i.e.*, user A) would benefit from a reasonably low threshold since a few items can already represent their preferences and more items can introduce more noises. Meanwhile, some users with diverse (wide) interests (*i.e.*, user B) would require the threshold to be relatively high to model a broader range of preferences and satisfy the need for exploration. To find a personalized splitting point κ_i for each user according to the learned matrix, the model requires an automatic splitting mechanism. Inspired by the threshold selection mechanism used in CART [25], we efficiently learn the personalized thresholds κ_i for different users by minimizing the following sum of square errors:

$$\kappa_i = \arg\min_{\eta} \left[\sum_{j=1}^{\eta} \left(\rho_{ij} - c_\eta^1\right)^2 + \sum_{j=\eta+1}^{N} \left(\rho_{ij} - c_\eta^2\right)^2 \right], \tag{20}$$

where ρ_{ij} is the j-th value of the sorted ρ_i and the index $\eta \in \{1, 2, \cdots, N-1\}$. $c_\eta^1 = \frac{1}{\eta}\sum_{j=1}^{\eta} \rho_{ij}$ represents the mean value of top η items while $c_\eta^2 = \frac{1}{N-\eta}\sum_{j=\eta+1}^{N} \rho_{ij}$ represents mean of the rest $N - \eta$ items. After obtaining the index of splitting points κ, we have the corresponding threshold values ρ_{i,κ_i}. Based on the personalized threshold ρ_{i,κ_i}, we can relabel by $r_{ij} \in \mathbf{R}$ as follows:

$$r_{ij} = \frac{1}{1 + \exp(-\beta(\rho_{ij} - \rho_{i,\kappa_i}))}, \tag{21}$$

where r_{ij} is a monotonically increasing function according to the value of $\rho_{ij} - \rho_{i,\kappa_i}$. β is a hyperparameter to control the inclination of the function.

Finally, to make the relabeling process more flexible, we propose to use a hyperparameter λ to control the degree of relabeling and obtain a new interaction matrix as follows:

$$\boldsymbol{X} = \lambda \boldsymbol{X} + (1 - \lambda)\boldsymbol{R} \odot \boldsymbol{X}, \tag{22}$$

where $\odot$ corresponds to the element-wise product. Considering that the noises in sparse positive interactions would induce worse effects in recommendation [5], in this work, we only target the part of the original dataset where $X_{ij} = 1$. In this way, the newly generated $\boldsymbol{X}$ can keep users' intrinsic preferences and reduce the noise ones. We summarize the learning procedure of our proposed ProRec in Algorithm 1, which is proved to guarantee the local convergence for denoised recommendation.

4. Experiments

In this section, we first conduct controlled experiments with synthetic noises on one dataset, so as to investigate ProRec's performance to different levels of noisy interactions. Then, we evaluate ProRec's performance on three original real-world datasets of recommendation with implicit feedback.

4.1. Experimental Settings

4.1.1. Dataset and evaluation protocols.

We use ML-100k for the noise-controlled experiments and do a 4:1 random splitting for train/test data following [26]. To simulate the noise, we randomly select 5%/10%/15%/20% of ground-truth records and flip labels in the train set. The selected records can be regarded as noises during training. For real data experiments, we use Amazon music (AMusic)[1], Amazon toys (AToy)[1], and Yahoo[2]. These datasets have been widely adopted in previous literature [27, 5, 28], and their statistics are summarized in Table 1. We do a 5:2:3 splitting for them following [28]. We evaluate the ability to distinguish noise by Hit Ratio, and recommendation performances by normalized discounted cumulative, mean average precision, and recall.

[1]https://github.com/familyld/DeepCF/tree/master/Data
[2]https://webscope.sandbox.yahoo.com/catalog.php?datatype=r

Algorithm 1: Partial Relaxed Optimal Transport for Denoised Recommendation (ProRec)

Data: The noisy user-item interactions $\boldsymbol{X}$, user popularity $\boldsymbol{p}$ and item popularity $\boldsymbol{q}$
Result: Denoised user embeddings $\boldsymbol{U}$ and item embeddings $\boldsymbol{V}$, a relabeled matrix $\boldsymbol{R}$.

while *not converged* **do**
  1. Update $\boldsymbol{U}$ and $\boldsymbol{V}$;
  if *Learning $\boldsymbol{U}$ and $\boldsymbol{V}$ via non-deep methods* **then** Update $\boldsymbol{U}$ and $\boldsymbol{V}$ via $||\boldsymbol{X} - \boldsymbol{U}\boldsymbol{V}^T||_F^2 + \zeta(||\boldsymbol{U}||_2^2 + ||\boldsymbol{V}||_2^2$ in Eq. 2 and calculate $\mathcal{M} = -\boldsymbol{U}\boldsymbol{V}^T$;
  else Update $\boldsymbol{U}$ and $\boldsymbol{V}$ via $\mathcal{C}(\boldsymbol{X}, G_y(\boldsymbol{U}, \boldsymbol{V})) + \zeta(||\boldsymbol{U}||_2^2 + ||\boldsymbol{V}||_2^2)$ in Eq. 3, where $\mathcal{C}(\cdot,\cdot)$ is binary cross-entropy loss; Then calculate $\mathcal{M} = -G_y(\boldsymbol{U}, \boldsymbol{V})$, where G_y is a classification network;
  2. Update $\boldsymbol{X}$;
  Model denoised recommendation problem by minimizing $\sum_{i=1}^{M}\sum_{j=1}^{N}\mathcal{M}_{ij}\pi_{ij} + \gamma H(\boldsymbol{\pi})$, where $\mathbf{1}_M\boldsymbol{\pi} = \boldsymbol{p}, \mathbf{1}_N\boldsymbol{\pi}^T = \boldsymbol{q}$, in Eq. 4;
  Compute a global optimal matching matrix $\boldsymbol{\pi}^*$ by the closed-form solution $\boldsymbol{\pi}^* = \max(\tilde{\mathcal{M}}\,\mathrm{diag}(\boldsymbol{p} \oslash \mathbf{1}_M\tilde{\mathcal{M}}), \mathrm{diag}(\boldsymbol{q} \oslash \mathbf{1}_N\tilde{\mathcal{M}}^T)\tilde{\mathcal{M}})$ in Eq. 7;
  Compute personalized thresholds $\boldsymbol{\kappa}$ by automatic splitting mechanism in Eq. 20;
  Relabel user-item interaction by sorting $\boldsymbol{\kappa}$ in Eq. 21;
  Update $\boldsymbol{X}$ by $\boldsymbol{X} = \lambda\boldsymbol{X} + (1-\lambda)\boldsymbol{R} \odot \boldsymbol{X}$ in Eq. 22;
end

Table 1
Statistics of the datasets used in our experiments.

Dataset	# User	# Item	# Interaction	Sparsity
ML-100k	942	1,447	55,375	0.9594
AMusic	1,776	12,929	46,087	0.9980
AToy	3,137	33,953	84,642	0.9992
Yahoo	1,948,882	46,110	48,817,561	0.9995

4.1.2. Methods for comparison

We compare two types of algorithms for recommendation. The first type is non-deep algorithms, which include: SVD [29] – A similarity-based method that constructs a similarity matrix through SVD decomposition of the implicit matrix $\boldsymbol{X}$; NCE [28] – Noise contrastive estimation item embeddings combined with a personalized user model based on PLRec [30]. The second type is deep learning methods, including CDAE [10] – Collaborative Denoising Autoencoder is a deep learning method specifically optimized for implicit feedback recommendation tasks; WRMF [10] – Weighted Regularized Matrix Factorization, which is a deep MF framework; VAE-CF [31] – Variational Autoencoder for Collaborative Filtering, which is one of the state-of-the-art deep learning based recommendation algorithm [28]; LightGCN [32] – LightGCN devises a light graph convolution for training efficiency and generation ability on recommendation scenario; EGLN [33] – EGLN enhances graph learning and node embedding modules iteratively based on mutual information maximization.

Besides the above methods, T-CE [5] and SGDL [34] are the existing frameworks for denoised recommendations. The first one discards the large-loss samples with a dynamic threshold in each iteration and the second one collects memorized interactions at the early stage of the training and leverages those data as denoising signals to guide the following training. Both of them are designed as plug-ins for deep learning methods. We add CDAE + T-CE and LightGCN + SGDL as baselines to compare the ability of denoising with the proposed ProRec, which are shown to win the best performance in their original works [5, 34].

For all the algorithms (except for CDAE + T-CE and LightGCN + SGDL), we add the proposed ProRec on top of them, allowing both non-deep and deep RS models to discriminate intrinsic and noisy interactions in an unsupervised fashion. Specifically, we have SVD + ProRec, NCE + ProRec, CDAE + ProRec, WRMF + ProRec, VAE-CF + ProRec, LightGCN + ProRec, and EGLN + ProRec, respectively.

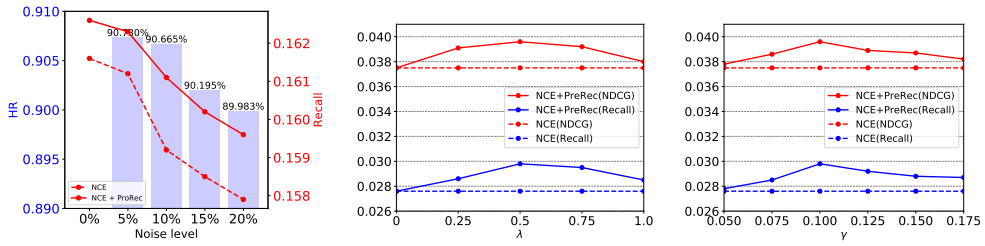
4.1.3. Implementation details

Implementations of the compared baselines are from open-source projects (SVD[3], NCE[3], CDAE[3], WRMF[3], VAE-CF[3], LightGCN[4], EGLN[5], T-CE[6] and SGDL[7]). We optimize ProRec with standard Adam. We tune all hyperparameters through grid search. In particular, learning rate in {0.0005, 0.001, 0.005, 0.01}, λ in {0.25, 0.5, 0.75, 1.0}, γ in {0.05, 0.075, 0.1, 0.125, 0.15, 0.175}, β in {1, 5, 10, 20, 50}, ζ in {0.0005, 0.001, 0.005, 0.01}, and the batch size in $\{100, 250, 500, 1000\}$ for different datasets. We also carefully tuned the hyperparameters of all baselines through cross-validation to achieve their best performances.

4.2. Controlled Experiments (Synthetic Noises)

Before looking at the performance of recommendation, we first inspect ProRec's ability to distinguish noises.

[3] https://github.com/wuga214/NCE_Projected_LRec
[4] https://github.com/gusye1234/LightGCN-PyTorch
[5] https://github.com/yimutianyang/SIGIR2021-EGLN
[6] https://github.com/WenjieWWJ/DenoisingRec
[7] https://github.com/zealscott/SGDL



(a) HR/Recall, controlled dataset (b) Varying λ on AMusic (c) Varying γ on AMusic

Figure 2: (a) Testing HR/Recall with different noise levels on the noise-controlled dataset. (b)-(c) Hyperparameter effect on the proposed ProRec.

Table 2
Comparison between the original baselines and the ones plus ProRec on three benchmark datasets. The best performances of non-deep and deep learning methods are in boldface. Statistically tested significant improvements with p-value < 0.01 brought by ProRec compared with the original methods are indicated with *.

model	N@5	M@5	R@5	N@5	M@5	R@5	N@5	M@5	R@5
		AMusic			AToy			Yahoo	
SVD [29]	0.0363	0.0667	0.0264	0.0116	0.0212	0.0084	0.1614	0.2316	0.1430
NCE [28]	0.0375	0.0693	0.0276	0.0143	0.0262	0.0104	0.2498	0.3440	0.2249
CDAE [10]	0.0074	0.0124	0.0060	0.0046	0.0080	0.0033	0.0716	0.1022	0.0652
WRMF [10]	0.0236	0.0435	0.0171	0.0086	0.0157	0.0062	0.2503	0.3244	0.2321
VAE-CF [31]	0.0146	0.0258	0.0115	0.0117	0.0212	0.0084	0.2622	0.3423	0.2380
LightGCN [32]	0.0406	0.0830	0.0368	0.0162	0.0296	0.0117	0.2750	0.3729	0.2494
EGLN [33]	0.0422	0.0866	0.0386	0.0166	0.0303	0.0120	0.2806	0.3809	0.2543
CDAE + T-CE [5]	0.0080	0.0143	0.0062	0.0049	0.0080	0.0038	0.0714	0.1032	0.0648
LightGCN + SGDL [34]	0.0417	0.0852	0.0378	0.0166	0.0304	0.0120	0.2825	0.3830	0.2561
SVD + ProRec	0.0374*	0.0677	0.0275*	0.0123*	0.0220*	0.0093*	0.1987*	0.2820*	0.1763*
NCE + ProRec	**0.0396***	**0.0723***	**0.0298***	**0.0149***	**0.0267***	**0.0108***	**0.2619***	**0.3562***	**0.2364***
CDAE + ProRec	0.0080*	0.0148*	0.0065*	0.0049	0.0084	0.0038*	0.0718	0.1028	0.0660*
WRMF + ProRec	0.0241*	0.0442*	0.0179*	0.0096*	0.0165*	0.0077*	0.2699*	0.3358*	0.2403*
VAE-CF + ProRec	0.0162*	0.0266*	0.0120	0.0129*	0.0231*	0.0102*	0.2743*	0.3535*	0.2472*
LightGCN + ProRec	0.0424*	0.0866*	0.0379*	0.0168*	0.0302*	0.0120*	0.2924*	0.3918*	0.2636*
EGLN + ProRec	**0.0445***	**0.0915***	**0.0406***	**0.0173***	**0.0317***	**0.0124***	**0.2973***	**0.4012***	**0.2705***

Since we know the index of the randomly added noises in the dataset, we evaluate the level of distinguished noises by Hit Ratio (HR) of the real added noises in Figure 2(a). Specifically, we observe a consistent high HR of NCE + ProRec around 90% when the level of noises increases. Furthermore, we test Recall of NCE and NCE + ProRec under different levels of noises. It can be clearly observed that NCE + ProRec always performs better than NCE, as the former can accurately relabel some noises to eliminate the effect of noises. For example, when the noise level is at 20%, NCE + ProRec significantly improves NCE from 0.1579 to 0.1596.

4.3. Real Data Experiments

4.3.1. Overall Performance Comparison

We compare the recommendation results of the proposed ProRec to those of the baseline models. Table 2 shows the NDCG, MAP, and Recall scores on three datasets. We have the following observations.

In three different domains, by integrating ProRec with both non-deep and deep methods, we can consistently improve strong baseline models and achieve start-of-the-art performance. Since the graph-based model can leverage high-order relations among users and items, EGLN outperforms all baselines on three datasets. Furthermore, our proposed EGLN + ProRec outperforms all deep-learning baselines on three datasets, ranging from 3.34% (achieved on AToy on Recall@5) to 6.38% (achieved on Yahoo on Recall@5), while NCE + ProRec outperforms all non deep-learning baselines on three datasets, ranging from 1.91% (achieved on AToy on MAP@5) to 7.97% (achieved on AMusic on Recall@5). All of these show that ProRec is capable of effective recommendation on different datasets.

One step further, we observe that the methods based on the proposed ProRec framework all outperform the original models, which indicates the advantage of the denoising process. Compared with existing denoising

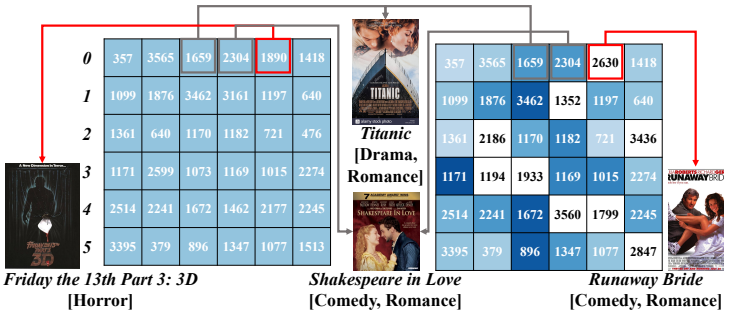


Figure 3: The denoising process of ProRec.

Table 3
Impact of different model components of ProRec on AMusic.

Metric	N@5	R@5	Time	N@5	R@5	Time
Method	NCE			EGLN		
Base	0.0375	0.0276	64s	0.0422	0.0386	82s
+EMD	0.0360	0.0259	98s	0.0406	0.0371	124s
+Sinkhorn	0.0384	0.0281	132s	0.0431	0.0398	174s
+ROT	0.0383	0.0281	76s	0.0430	0.0392	109s
+ProRec	0.0396	0.0298	82s	0.0445	0.0406	118s

frameworks (*i.e.*, T-CE and SGDL), ProRec can not only be flexibly integrated with both deep and non-deep RS methods but also gain more improvements.

Moreover, on top of existing RS methods, our proposed ProRec in Eq. 4 does not introduce significant computations. In the third and sixth columns in Table 3, we compare the model's training time under the same experimental setting (*e.g.*, embedding dimension). As can be observed, the training time of the proposed NCE + ProRec and LightGCN + ProRec are within the same scale as NCE and LightGCN on AMusic dataset.

Note that the improvements brought by the denoising process on the denser datasets (*e.g.*, the performance gains between 0% noise level and 5% noise level on ML-100k shown in Figure 2(a)) are less than those on the sparse datasets (*e.g.*, the performance gains between EGLN + ProRecn and EGLN on Yahoo in Table 2). Although we can explicitly eliminate some effects of noisy interactions (as shown in Figure 2(a)), they only have a minor impact on the learned embeddings when interactions are enough in the observed data, which is consistent with the results in recent studies [35].

4.3.2. Model Ablation and Hyperparameter Study

In this section, three groups of ablation studies are conducted to evaluate the impacts of our proposed many-to-many matching, relaxed regularization, and personalized thresholding mechanisms, as well as the effects of hyperparameters.

The effect of continuous many-to-many matching. Compared with the many-to-many matching of NCE + Sinkhorn, it is ineffective to directly adopt one-to-one matching (*i.e.*, NCE + EMD) due to the removal of many matched interactions. For example, NCE + Sinkhorn significantly improves NCE from 0.0276 to 0.0281 (on Recall@5 on AMusic) while both metrics of NCE + EMD are lower than those of the original NCE.

The efficiency of relaxed regularization. By balancing the performance and the runtime via relaxing the optimized constraints, we can observe that NCE + ROT can reduce around half of the runtime while keeping almost the same NDCG and Recall on AMusic and Yahoo.

The effect of personalized thresholding mechanism. We first use one global threshold for ROT to study the effectiveness of the personalized thresholding mechanism. In the experiment, we search the hyperparameter of global thresholds σ in $\{5, 10, 15, 20, 25\}$ and find that NCE + ROT achieves the best performance when $\sigma = 10$. Compared with NCE + ROT, NCE + ProRec can individually discriminate intrinsic preferences and noises without tuning the hyperparameter of the threshold, and then improve the performance of the partial OT framework. For example, the performance of NCE + ProRec achieves 0.0298, which outperforms 0.0281 of NCE + ROT on Recall@5 on AMusic dataset.

The effect of hyperparameters. Our proposed ProRec framework introduces four hyperparameters: γ, β, ζ and λ, where we empirically set β to 20 and ζ to 0.001. λ controls the degree of the relabeling and γ controls the sparsity of the matching matrix. Take NCE + ProRec for example (shown in Figure 2(b)-2(c)), the optimal λ and γ on AMusic are found to be 0.5 and 0.1, respectively. In general, we can set λ to 0.5 for denser datasets and 0.25 for the sparser datasets and always set γ to 0.1.

4.3.3. Denoising Case Study

To demonstrate the advantages of the denoising process, we visualize the interaction matrix given by ML-100k on the left and the learned matrix of the proposed NCE + ProRec on the right (as shown in Figure 3). The numbers in the boxes are actual movie IDs. The colors in the original interaction matrix (left) are the same, indi-

cating uniform weights, whereas those in the relabeled matrix (right) are different. The different depths of blue represent the matching cost (the deeper, the lower) and white boxes denote the intrinsically preferred items. As we can see from the different colors, our many-to-many matching matrix effectively discriminates intrinsic and noisy interactions. Moreover, we can also observe the personalized thresholding mechanism to work as different numbers of items is being replaced for different users. Furthermore, we show several movies relevant to user 0. Since most interacted movies of user 0 are from the romance category, such as *Titanic* and *Shakespeare in Love*, the horror movie is abnormal and is unlikely to reflect her/his intrinsic preference. NCE + ProRec automatically learns to highlight the two romantic movies. After downweighing *Friday the 13th Part 3: 3D*, the scoring of the user's preference changes, and thus the ranking of another romantic movie (*Runaway Bride*) rises to show up in the top $K = 6$ candidate list for user 0.

5. Conclusion

In this paper, we propose a novel ProRec framework for the recommendation with implicit feedback. ProRec can effectively denoise the user-item interactions by flexibly serving as a plug-in, so as to further improve the recommendation accuracy for both non-deep and deep learning RS methods. We demonstrate the superior performance of ProRec in recommendation through extensive experiments and showcase its effectiveness in denoising user-item interactions through insightful case studies. Since we achieve denoising in a fully unsupervised fashion without accessing any additional user or item data, we expect ProRec to incur no more negative societal impact than any basic recommender system.

References

[1] Y. Kim, A. Hassan, R. W. White, I. Zitouni, Modeling dwell time to predict click-level satisfaction, in: WSDM, 2014.

[2] H. Lu, M. Zhang, W. Ma, C. Wang, F. xia, Y. Liu, L. Lin, S. Ma, Effects of user negative experience in mobile news streaming, in: SIGIR, 2019.

[3] B. Yang, S. Lee, S. Park, S.-g. Lee, Exploiting various implicit feedback for collaborative filtering, in: WWW, 2012.

[4] N. Courty, R. Flamary, D. Tuia, A. Rakotomamonjy, Optimal transport for domain adaptation, TPAMI 39 (2017) 1853–1865.

[5] W. Wang, F. Feng, X. He, L. Nie, T.-S. Chua, Denoising implicit feedback for recommendation, WSDM (2021).

[6] B. Bhushan Damodaran, B. Kellenberger, R. Flamary, D. Tuia, N. Courty, Deepjdot: Deep joint distribution optimal transport for unsupervised domain adaptation, in: ECCV, 2018.

[7] V. Huynh, H. Zhao, D. Phung, Otlda: A geometry-aware optimal transport approach for topic modeling, NIPS (2020).

[8] M. Cuturi, Sinkhorn distances: Lightspeed computation of optimal transport (2013).

[9] R. Xu, P. Liu, Y. Zhang, F. Cai, J. Wang, S. Liang, H. Ying, J. Yin, Joint partial optimal transport for open set domain adaptation, in: IJCAI, 2020.

[10] Y. Hu, Y. Koren, C. Volinsky, Collaborative filtering for implicit feedback datasets, in: ICDM, 2008.

[11] Y. Liu, Q. Liu, Y. Tian, C. Wang, Y. Niu, Y. Song, C. Li, Concept-aware denoising graph neural network for micro-video recommendation, in: Proceedings of the 30th ACM International Conference on Information & Knowledge Management, 2021, pp. 1099–1108.

[12] R. Jagerman, H. Oosterhuis, M. de Rijke, To model or to intervene: A comparison of counterfactual and online learning to rank from user interactions, in: SIGIR, 2019.

[13] H. Lu, M. Zhang, S. Ma, Between clicks and satisfaction: Study on multi-phase user preferences and satisfaction for online news reading, in: SIGIR, 2018, pp. 435–444.

[14] T. Schnabel, A. Swaminathan, A. Singh, N. Chandak, T. Joachims, Recommendations as treatments: Debiasing learning and evaluation, JMLR (2016).

[15] L. Jiang, Z. Zhou, T. Leung, L.-J. Li, L. Fei-Fei, Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels, in: ICML, 2018.

[16] N. Kolkin, J. Salavon, G. Shakhnarovich, Style transfer by relaxed optimal transport and self-similarity, in: CVPR, 2019.

[17] M. Kusner, Y. Sun, N. Kolkin, K. Weinberger, From word embeddings to document distances, in: ICML, 2015, pp. 957–966.

[18] L. Chen, S. Dai, C. Tao, H. Zhang, Z. Gan, D. Shen, Y. Zhang, G. Wang, R. Zhang, L. Carin, Adversarial text generation via feature-mover's distance, in: NIPS, 2018.

[19] L. Chen, Y. Zhang, R. Zhang, C. Tao, Z. Gan, H. Zhang, B. Li, D. Shen, C. Chen, L. Carin, Improving sequence-to-sequence learning via optimal transport, in: ICLR, 2019.

[20] Y. Meng, X. Yan, W. Liu, H. Wu, J. Cheng, Wasserstein collaborative filtering for item cold-start recommendation, in: Proceedings of the 28th ACM Conference on User Modeling, Adaptation and Personalization, 2020, pp. 318–322.

[21] Q. Xu, J. Xiong, X. Cao, Q. Huang, Y. Yao, From social to individuals: a parsimonious path of multi-level models for crowdsourced preference aggregation, TPAMI (2018).
[22] A. Gretton, D. Sejdinovic, H. Strathmann, S. Balakrishnan, M. Pontil, K. Fukumizu, B. K. Sriperumbudur, Optimal kernel choice for large-scale two-sample tests, in: NIPS, 2012.
[23] S. Boyd, N. Parikh, E. Chu, Distributed optimization and statistical learning via the alternating direction method of multipliers, 2011.
[24] N. Parikh, S. Boyd, Proximal algorithms, Foundations and Trends in optimization (2014).
[25] N. Angelopoulos, J. Cussens, Exploiting informative priors for bayesian classification and regression trees., in: IJCAI, 2005.
[26] J. Ding, Y. Quan, Q. Yao, Y. Li, D. Jin, Simplify and robustify negative sampling for implicit collaborative filtering (2020).
[27] Z.-H. Deng, L. Huang, C.-D. Wang, J.-H. Lai, S. Y. Philip, Deepcf: A unified framework of representation learning and matching function learning in recommender system, in: AAAI, volume 33, 2019, pp. 61–68.
[28] G. Wu, M. Volkovs, C. L. Soon, S. Sanner, H. Rai, Noise contrastive estimation for one-class collaborative filtering, in: SIGIR, 2019.
[29] P. Cremonesi, Y. Koren, R. Turrin, Performance of recommender algorithms on top-n recommendation tasks, in: RecSys, 2010.
[30] S. Sedhain, H. Bui, J. Kawale, N. Vlassis, B. Kveton, A. K. Menon, T. Bui, S. Sanner, Practical linear models for large-scale one-class collaborative filtering, in: IJCAI, 2016.
[31] D. Liang, R. G. Krishnan, M. D. Hoffman, T. Jebara, Variational autoencoders for collaborative filtering, in: WWW, 2018.
[32] X. He, K. Deng, X. Wang, Y. Li, Y. Zhang, M. Wang, Lightgcn: Simplifying and powering graph convolution network for recommendation, in: SIGIR, 2020, pp. 639–648.
[33] Y. Yang, L. Wu, R. Hong, K. Zhang, M. Wang, Enhanced graph learning for collaborative filtering via mutual information maximization, in: SIGIR, 2021, pp. 71–80.
[34] Y. Gao, Y. Du, Y. Hu, L. Chen, X. Zhu, Z. Fang, B. Zheng, Self-guided learning to denoise for robust recommendation, in: SIGIR, 2022.
[35] S. Pal, S. Malekmohammadi, F. Regol, Y. Zhang, Y. Xu, M. Coates, Non-parametric graph learning for bayesian graph neural networks, PMLR (2020).