Block-aware Item Similarity Models for Top-N Recommendation

YIFAN CHEN*, National University of Defense Technology, China
YANG WANG, Key Laboratory of Knowledge Engineering with Big Data, Ministry of Education, Hefei University of Technology, China
XIANG ZHAO, National University of Defense Technology, China
JIE ZOU, University of Amsterdam, The Netherlands
MAARTEN DE RIJKE, University of Amsterdam & Ahold Delhaize, The Netherlands

Top-N recommendations have been studied extensively. Promising results have been achieved by recent item-based collaborative filtering methods. The key to item-based collaborative filtering lies in the estimation of item similarities. Observing the *block-diagonal structure* of the item similarities in practice, we propose a block-diagonal regularization over item similarities for item-based collaborative filtering. The intuitions behind block-diagonal regularization are: (1) with block-diagonal regularization, item clustering is embedded into the learning of item-based collaborative filtering methods; (2) block-diagonal regularization induces sparsity of item similarities, which guarantees recommendation efficiency; and (3) block-diagonal regularization captures in-block transitivity to overcome rating sparsity. By regularizing the item similarity matrix of item similarity models with block-diagonal regularization, we obtain a block-aware item similarity model. Our experimental evaluations on a large number of datasets show that the block-diagonal structure is crucial to the performance of top-N recommendation.

CCS Concepts: • **Information systems** → **Recommender systems**.

Additional Key Words and Phrases: item collaborative filtering; item similarity model; top-N recommendation

ACM Reference Format:
Yifan Chen, Yang Wang, Xiang Zhao, Jie Zou, and Maarten de Rijke. 2020. Block-aware Item Similarity Models for Top-N Recommendation. *ACM Transactions on Information Systems* 1, 1 (July 2020), 26 pages. https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

Given a user profile with a record of purchases or ratings, the top-N recommendation task is to recommend a small set of N items from a large item collection [15], in order to *effectively* and

*Corresponding author.

This work is partially supported by NSFC under grants Nos. 61872446, 71690233 and PNSF of Hunan under grant No. 2019JJ20024. Yang Wang is supported by NSFC No. 61806035, U1936217 and The Key Research and Technology Development Projects of Anhui Province (No. 202004a05020043). Maarten de Rijke is partially supported by the Innovation Center for Artificial Intelligence (ICAI). All content represents the opinion of the authors, which is not necessarily shared or endorsed by their respective employers and/or sponsors.
Authors' addresses: Yifan Chen, National University of Defense Technology, Changsha, China, yfchen@nudt.edu.cn; Yang Wang, Key Laboratory of Knowledge Engineering with Big Data, Ministry of Education, Hefei University of Technology, China, yangwang@hfut.edu.cn; Xiang Zhao, National University of Defense Technology, Changsha, China, xiangzhao@nudt.edu.cn; Jie Zou, University of Amsterdam, Amsterdam, The Netherlands, j.zou@uva.nl; Maarten de Rijke, University of Amsterdam & Ahold Delhaize, Amsterdam, The Netherlands, m.derijke@uva.nl.

1046-8188/2020/7-ART
https://doi.org/10.1145/1122445.1122456

efficiently help the user identify the services and products that best fit his/her taste. A well-designed top-N recommendation algorithm should predict the recommendation scores for each user on each item in the pool of products, so as to recommend the top-N items with the highest scores.

Collaborative filtering (CF) has been successfully employed for top-N recommendations [47]. CF-based methods include latent space models [15] and neighborhood-based methods [17]. Although latent space models can be utilized to generate an ordered list of items, they were originally designed for rating prediction tasks and therefore they are sub-optimal for top-N recommendation. Neighborhood-based methods (user-based or item-based) identify similar users or items. Compared with other models, they deliver better performance for top-N recommendation [1, 17, 30, 42], and item-based methods outperform user-based methods [11].

Early item-based collaborative filtering (ICF) methods employ statistical measures, e.g., Pearson coefficient or cosine similarity, to estimate item similarities [17, 48]. Recommendations by such heuristic-based approaches are efficient but have inferior performance. Item similarity models (ISMs) are a later proposal. The sparse linear method (SLIM) [42] makes high-quality recommendations and ensures efficiency of recommendation by learning a *sparse* item similarity matrix. One inherent limitation of SLIM is that it can only model relations between items that have been co-rated by at least some users; their performance downgrades when ratings are sparse. To address this issue, the factored item similarity model (FISM) [30] factorizes the similarity matrix into low-rank matrices, so that *transitive* relations between items can be well captured. However, the item similarity matrix generated by FISM is dense. To ensure sparsity while enforcing low-rankness, the low-rank sparse linear method (LorSLIM) [10] uses rank regularization for the item similarity matrix, where the learned similarity matrix is empirically shown to have a *block-diagonal* structure.

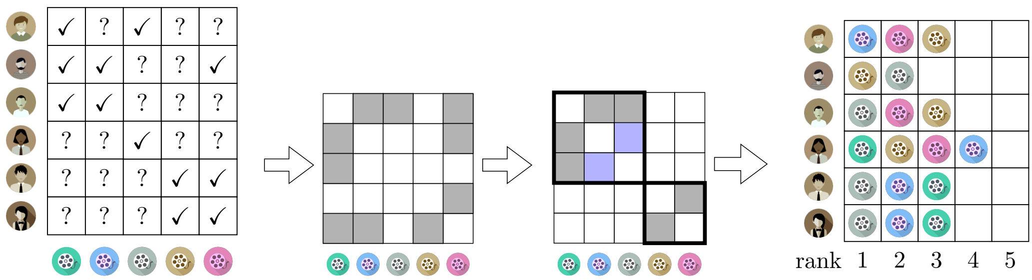
1.1 Motivation for block-diagonal structure

The block-diagonal structure is critical to top-N recommendation: it captures *latent item groups*, which are subsets of items so that items contained in them are more similar to each other than to items in other subsets. Latent item groups are used in a wide spectrum of real-world collaborative filtering applications. For instance, in the movie domain, "Inception" would be similar to "Interstellar" as both are science fiction and suspense movies, whereas its degree of similarity to "Titanic" is low as the latter belongs to the categories of romantic and disaster movies. In many real-world datasets, the item collection is increasingly large, making the top-N recommendation task increasingly hard. As shown by [4], training recommender systems globally for all items can leave many items badly-modeled and thus under-served. Rather than learning globally, we propose *block-diagonal regularization* (BDR) to enforce a block-diagonal structure in the item similarity matrix, so that similarities with a block can be better modeled locally.

Low-rankness enforced by LorSLIM can be seen as an *indirect* way of enforcing a block-diagonal structure. Theoretically, the block-diagonal matrix can only be generated under rigid conditions [39]. In practice, learned item similarity matrices are far from being block-diagonal [20, 38]. Even if the similarity matrix is block-diagonal, we cannot require it to exactly have a pre-specified number of blocks.

An alternative way of capturing latent item groups is to group items into sub-groups based on rating information. While clustering is prevalent in the context of collaborative filtering (CF) [11, 54, 56, 58–60], it has been less studied for item-based collaborative filtering (ICF). Recent work [2, 11, 12] studies user clustering for ICF. In these publications, users are clustered into subgroups based on ratings and a local ICF model is estimated for each cluster; hence, clustering and the estimation of local models are treated as separate procedures.

Different from these methods, the model proposed in this paper forms a multi-task learning framework, where item clustering and item similarity learning are optimized in an alternating



(a) Rating matrix (b) Without BDR (c) With BDR (d) Recommendation

Fig. 1. Example to show the effect of BDR. Figure 1(a) is a rating matrix from the movie recommendation domain, where the rows and columns represent movies and users, respectively. If a user has rated a movie, the corresponding entry is marked with "✓", otherwise with "?"; Figure 1(b) represents the item similarity matrix obtained by an ICF method without BDR, where the non-zero entries are grayed; Figure 1(c) is the learned item similarity matrix with BDR when $c = 2$; Figure 1(d) is the sorted list of recommendations of unrated movies. The item similarity matrix in Figure 1(c) has a block-diagonal structure, with two blocks inside the rectangles with thick borders. Sparsity is achieved as off-block similarities are penalized. Transitive relations are also recovered within the block (the blue grids).

manner. The two tasks mutually enhance each other: the optimized item similarities help to better categorize items and items within the same group are tend to be more similar than in different groups.

1.2 Our contributions

In this paper, we propose block-diagonal regularization (BDR) in order to obtain a block-diagonal structure in the item similarity matrices of item similarity models (ISMs) methods. BDR encourages the learned item similarity matrix to be, or to be close to, a c-block diagonal, where c is the number of blocks. BDR integrates item clustering into the learning of item similarities, where off-block similarities are penalized. The block-diagonal structure achieved by BDR is adaptively optimized during the training process.

Although many recent top-N recommendation methods are neural-based approaches [18, 25, 37, 57], doubts have been raised about the reproducibility of these methods; many are being outperformed by relatively simple heuristic methods [16]. Interestingly, the neural-based methods fail to consistently outperform SLIM. We view these findings as a justification to continue to improve linear top-N recommendation methods. We formulate block-aware item similarity model (BISM) based on SLIM, where we penalize the item similarity matrix by BDR in order to capture the block-diagonal structure. Figure 1 gives an illustrative example of how BDR works for ISMs. BISM is empirically shown to outperform SLIM consistently and significantly, and the superiority over other state-of-the-art baselines is established.

We demonstrate that with BDR, the learned item similarity matrix of ISMs enjoys the following properties: (1) *block-diagonality*: BDR captures latent item groups for fine-grained ICF; (2) *sparsity*: BDR ensures efficiency when performing top-N recommendation; and (3) *transitivity*: BDR captures transitive relations between items that are essential for good performance in sparse datasets.

Our main contributions in this paper are the following:

(1) we propose block-diagonal regularization (BDR) to capture the block-diagonal structure in item similarity matrices so as to improve item-based collaborative filtering (ICF);

Table 1. Notation.

Notation	Description
m	number of users
n	number of items
c	number of latent item groups
$R \in \mathbb{R}^{m\times n}$	user rating matrix
$S \in \mathbb{R}^{n\times n}$	item similarity matrix
$L_S \in \mathbb{R}^{n\times n}$	the laplacian matrix of S
$F \in \mathbb{R}^{n\times c}$	the auxiliary matrix of BDR
s_{ij}	the similarity between item i and j
$\mathcal{R}_u^+$	the set of items rated by user u
r_{ui}	the score of item i rated by user u
$\tilde{r}_{ui}$	the predicted score of rating r_{ui}

(2) we apply BDR to item similarity models (ISMs) and formulate block-aware item similarity model (BISM), whose effectiveness is theoretically guaranteed; and
(3) we conduct extensive experiments to assess block-aware item similarity model (BISM), which is shown to outperform the state-of-the-art.

2 PRELIMINARIES

Before introducing our techniques, we describe the notation used in the paper. All vectors are column vectors and represented by bold lowercase letters (e.g., $\boldsymbol{x}$). All matrices and constants are represented by uppercase letters (e.g., X) and Greek letters (e.g., α), respectively. Given a matrix X, x_{ij} represents the entry at the i-th row and j-th column. $\|X\|_1 = \sum_{i,j} |x_{ij}|$ and $\|X\|_F = (\sum_{i,j} x_{ij}^2)^{1/2}$ are the ℓ_1-norm and ℓ_F-norm of matrix X, respectively. We write I to denote the identity matrix.

We write m and n for the number of users and items, respectively. $R \in \mathbb{R}^{m\times n}$ represents user ratings, either explicit or implicit. Item similarity matrices are denoted by $S \in \mathbb{R}^{n\times n}$, where s_{ij} represents the similarity between item i and j. We summarize the notation used in this paper in Table 1. Given S, ICF methods predict the score of user u for target item i by:

$$\tilde{r}_{ui} = \sum_{j\in\mathcal{R}_u^+} s_{ji}, \tag{1}$$

where $\mathcal{R}_u^+$ indicates the set of items rated by u. To learn the item similarity matrix S, SLIM [42] formulates the following model:

$$\min_S \frac{1}{2}\|R - RS\|_F^2 + \alpha\|S\|_1 + \frac{\beta}{2}\|S\|_F^2 \text{ such that } \forall i, j, s_{ij} \geq 0 \text{ and } s_{ii} = 0. \tag{2}$$

3 THE PROPOSED METHOD

In this section, we propose a regularization method to achieve block-diagonality in item similarity matrices for ICF methods. In Section 3.1, we introduce the BDR and present theoretical findings of BDR. We then discuss negative effects of BDR and provide our solution in Section 3.2. Finally, we apply BDR to SLIM and introduce a BISM in Section 3.3.

3.1 Block-diagonal regularization

Block-diagonality. We recall some basic results from spectral graph theory [13]. Let S be an item similarity matrix. We define the Laplacian matrix of S, denoted by L_S, as:

$$L_S = \mathrm{Diag}(A\mathbf{1}) - A, \tag{3}$$

where $A = \frac{S+S^T}{2}$. $\mathrm{Diag}(\boldsymbol{x})$ forms a diagonal matrix from $\boldsymbol{x}$ with its i-th element on the diagonal being x_i. We use $\mathbf{1} \in \mathbb{R}^n$ to denote a vector whose elements are all 1. It is easy to see that L_S is positive semidefinite as $\boldsymbol{x}^T L_S \boldsymbol{x} \geq 0, \forall \boldsymbol{x} \in \mathbb{R}^n$. We recall the following theorem to capture the connection between the Laplacian matrix and clusters of items.

Theorem 3.1 ([41]). *Let S be an item similarity matrix. The multiplicity c of the eigenvalue 0 of the Laplacian matrix L_S is equal to the number of connected components of the graph underlying S.*

Theorem 3.1 indicates that if $\mathrm{rank}\,(L_S) = n - c$, then S provides an ideal assignment for items by partitioning items into c groups. To capture latent item groups, we can require that the item similarity matrix S learned by ICF methods follows this rank constraint, so that we learn S with a c-block-diagonal structure. However, the rank constraint brings great difficulty for optimization. Besides, having exactly c blocks is not always desirable for S, as in many cases, item groups are not non-overlapping. Instead, we introduce regularization to S, in order to enforce the rank of L_S, in place of the rank constraint.

We first recall Ky Fan's Theorem [19]:

$$\sum_{i=1}^{c} \sigma_i = \min_F \sum_{i,j}^{n} \|\boldsymbol{f}_i - \boldsymbol{f}_j\|_2^2 s_{ij}, \text{ such that } F \in \mathbb{R}^{n\times c}, F^T F = I, \tag{4}$$

where σ_i denotes the i-th smallest eigenvalue of L_S; F is an auxiliary matrix and $\boldsymbol{f}_i$ is the i-th row of F. As L_S is positive semidefinite, e.g., $\sigma_i \geq 0$, we can enforce $\sum_{i=1}^{c} \sigma_i$ to be zero, so as to achieve the c-block-diagonal structure. Thus, the BDR is given as:

$$\|S\|_B = \min_{F^T F = I} \sum_{i,j}^{n} \|\boldsymbol{f}_i - \boldsymbol{f}_j\|_2^2 s_{ij}. \tag{5}$$

Sparsity. Besides block-diagonality, BDR can also increase sparsity as the block-diagonal structure is also sparse. To see this, we establish Theorem 3.2.

Theorem 3.2. *BDR is a weighted ℓ_1-norm regularization if $S \geq 0$.*

Proof. Suppose $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n$ are the eigenvectors for L_S, which are in ascending order of eigenvalues. For all i, j, if $i = j$, we have: $\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2^2 = 0$, else we have $\boldsymbol{x}_i^T \boldsymbol{x}_j = 0$ and $\boldsymbol{x}_i^T \boldsymbol{x}_i = 1$, and we can derive $\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2^2$ as:

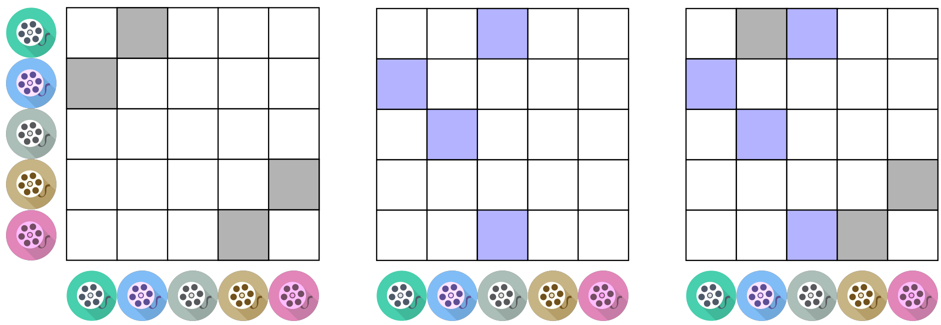
$$\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2^2 = \boldsymbol{x}_i^T \boldsymbol{x}_i + \boldsymbol{x}_j^T \boldsymbol{x}_j - 2\boldsymbol{x}_i^T \boldsymbol{x}_j = 2. \tag{6}$$

As $S \geq 0$, we can rewrite the block-diagonal regularization as:

$$\|S\|_B = \sum_{i,j}^{n} \|\boldsymbol{f}_i - \boldsymbol{f}_j\|_2^2 s_{ij} = \sum_{i,j}^{n} \left| d_{ij} s_{ij} \right| = \|D \circ S\|_1,$$

where D is a Euclidean distance matrix with $d_{ij} = \|\boldsymbol{f}_i - \boldsymbol{f}_j\|_2^2$. Therefore, BDR is a weighted ℓ_1-norm regularization and d_{ij} can be formulated as:

$$d_{ij} = \begin{cases} 2 - \sum_{l=c+1}^{n} (x_{il} - x_{jl})^2, & i \neq j \\ 0, & \text{otherwise.} \end{cases} \quad \square \tag{7}$$



(a) S^l (b) S^g (c) $S = S^l + S^g$

Fig. 2. BDR with $c = 3$.

Transitivity. We also show that the learned item similarity matrix S regularized by BDR can capture transitivity. We first rewrite Eq. (2) by introducing an auxiliary matrix S':

$$\min_{S,S'} \frac{1}{2}\|R - RS\|_F^2 + \frac{\gamma}{2}\|S - S'\|_F^2 + \lambda\|S'\|_B. \tag{8}$$

Eq. (8) is equivalent to Eq. (2) as long as γ is large enough. We first show that S' is learned to capture transitive relations among blocks. By fixing S, the closed-form solution of S' to Eq. (8) can be derived:

$$S' = S - \lambda D. \tag{9}$$

We then propose Theorem 3.3 to show the transitivity, indicating that if s'_{ij} and s'_{jk} are no less than a certain value, then s'_{ik} can be ensured to be non-negative. This implies that the relation is extended from i to k based on connections between i, j and j, k.

Theorem 3.3. *Given $s'_{ij} \geq 0$ and $s'_{jk} \geq 0$, if $s'_{ij} > s_{ij} - \frac{1}{4}s_{ik}$ and $s'_{jk} > s_{jk} - \frac{1}{4}s_{ik}$, then $s'_{ik} \geq 0$.*

Proof. According to Eq. (9), we have:

$$\begin{aligned} s'_{ij} &= s_{ij} - \lambda d_{ij} \\ s'_{jk} &= s_{jk} - \lambda d_{jk} \\ s'_{ik} &= s_{ik} - \lambda d_{ik}. \end{aligned} \tag{10}$$

As D is a Euclidean distance matrix, the triangle inequality holds:

$$\sqrt{d_{ik}} \leq \sqrt{d_{ij}} + \sqrt{d_{jk}}. \tag{11}$$

Therefore, we have

$$\begin{aligned} s'_{ik} &\geq z_{ik} - \lambda\left(\sqrt{d_{ij}} + \sqrt{d_{jk}}\right)^2 \\ &= z_{ik} - \lambda\left(\sqrt{\frac{1}{\lambda}(z_{ij} - s'_{ij})} + \sqrt{\frac{1}{\lambda}(z_{jk} - s'_{jk})}\right)^2 > 0. \end{aligned}$$ □

Since S is equal or close to S' if γ is large enough, the learned S can also capture transitive relations.

3.2 Global item similarities

In the previous section we established basic theoretical properties of BDR. However, directly penalizing S by BDR can trigger an adversarial effect: some columns of S will be entirely zero-value. The reason behind this is that if the pre-defined value c is larger than the intrinsic number of latent item groups, some lonely items that do not show much affiliation with any of the groups could be sacrificed. Recall the example rating in Figure 1. If we set $c = 3$, the third column of S is learned to be all-zero, as shown in Figure 2(a). This is justifiable as the BDR tries to encourage three blocks, where the third item is itself a block, so that every off-diagonal entry within the third column is encouraged to be zero. While this conforms to three blocks, it is not desirable for recommendation purposes as the movie in gray cannot be recommended.

To address the adversarial effect noted above, besides learning an item similarity matrix regularized by BDR, we introduce another item similarity matrix, which is not penalized by BDR. Since the one penalized by BDR captures latent item groups, we denote it by S^l, namely local item similarity matrix (Figure 2(a)). Similarly, we denote the one without the regularization of BDR by S^g, namely global similarity matrix (Figure 2(b)). The effect of learning a combination of S^g and S^l is two-fold: (1) it compensates for the negative effect of BDR that some columns of S^l will learn to be entirely zero-value; and (2) it captures similarities among different blocks. As shown in Figure 2(c), the combination of S^l and S^g can capture the underlying relations among items.

3.3 Block-aware item similarity model

Based on the above discussions, we can formulate the proposed BISM by the following equation:

$$\begin{gathered}\min_{S^l,S^g,F} \frac{1}{2}\|R - R(S^l + S^g)\|_F^2 + \alpha\|S^g\|_1 + \frac{\beta}{2}\left(\|S^l\|_F^2 + \|S^g\|_F^2\right) + \lambda\|S^l\|_B \\ \text{such that } S^l, S^g \geq 0, \operatorname{diag}(S^l) = \operatorname{diag}(S^g) = 0 \text{ and } F^TF = I.\end{gathered} \tag{12}$$

Let us explain BISM in some detail. (1) The first term in the objective forms the loss function by ICF, as given in Eq. (1). The difference with Eq. (1) is that we construct the item similarity s_{ij} as the linear summation of s^l_{ij} and s^g_{ij}. (2) We penalize S^l by BDR to capture the block-diagonal structure behind item similarities. The structure of S^l is close to c-block-diagonal if λ is large enough. (3) The ℓ_1-norm regularization is introduced to S^g to encourage sparsity. The ℓ_1-norm is not used for S^l since the BDR can encourage sparsity. (4) Both S^l and S^g are penalized by the ℓ_F-norm to avoid overfitting. (5) The constraint on the diagonal of S^l and S^g is proposed to avoid the trivial solution that $S^l + S^g = I$. (6) We follow Eq. (2) and require both S^l and S^g to be non-negative, in order to learn meaningful similarities.

4 OPTIMIZATION

BISM learns the combination of a local similarity matrix S^l and a global similarity matrix S^g. Recall from the definition in Eq. (5) that BDR for ISMs involves another variable F, which is an auxiliary variable introduced to adaptively optimize BDR according to item similarities. Therefore, we introduce an alternating minimization algorithm to optimize BISM.

4.1 Fixing S^l, S^g and update F

When fixing S^l, Eq. (12) is reduced to the following problem:

$$\min_F \operatorname{Tr}\left(F^T L_{S^l} F\right) \text{ such that } F^TF = I, \tag{13}$$

where L_{S^l} is the Laplacian matrix of S^l (see Eq. (3)). A closed-form solution for F can be obtained as the c eigenvectors corresponding to the c smallest eigenvalues of L_{S^l}.

4.2 Fixing F and update S^l, S^g

We then optimize Eq. (12) with fixed F. Due to the independence of columns of S^l and S^g, we can rewrite Eq. (12) by decoupling it into a set of n independent optimization problems:

$$\min_{\boldsymbol{s}_i^l, \boldsymbol{s}_i^g} \frac{1}{2}\|\boldsymbol{r}_i - R(\boldsymbol{s}_i^l + \boldsymbol{s}_i^g)\|_2^2 + \alpha\|\boldsymbol{s}_i^g\|_1 + \frac{\beta}{2}\left(\|\boldsymbol{s}_i^l\|_2^2 + \|\boldsymbol{s}_i^g\|_2^2\right) + \lambda\sum_{j=1}^{n} d_{ij}s_{ij}^l \quad \text{such that } \boldsymbol{s}_i^l, \boldsymbol{s}_i^g \geq 0, s_{ii}^l = s_{ii}^g = 0, \tag{14}$$

where $\boldsymbol{r}_i$, $\boldsymbol{s}_i^l$ and $\boldsymbol{s}_i^g$ are the i-th column of R, S^l and S^g, respectively, and $d_{ij} = \lambda\|\boldsymbol{f}_i - \boldsymbol{f}_j\|_2^2$. Learning S^l and S^g can easily be parallelized given the n independent problems in Eq. (14). Due to the non-negative constraint on $\boldsymbol{s}_i^l$ and $\boldsymbol{s}_i^g$, we apply the multiplicative update method [34] for efficient updating. The multiplicative update method is an iterative updating method that ensures that during each iteration, the variables to be updated are non-negative.

We derive the update rule for $\boldsymbol{s}^l$. We denote J as a shorthand for the objective function in Eq. (14) regarding $\boldsymbol{s}^l$ only, which is written as follows:

$$J = \frac{1}{2}\|\boldsymbol{r}_i - R(\boldsymbol{s}_i^l + \boldsymbol{s}_i^g)\|_2^2 + \frac{\beta}{2}\|\boldsymbol{s}_i^l\|_2^2 + \lambda\sum_{j=1}^{n} d_{ij}s_{ij}^l \tag{15}$$

Then the partial derivative over $\boldsymbol{s}^l$ is:

$$\frac{\partial J}{\partial \boldsymbol{s}^l} = R^T R(\boldsymbol{s}^l + \boldsymbol{s}^g) - R^T\boldsymbol{r}_i + \boldsymbol{d}_i + \beta\boldsymbol{s}^l, \tag{16}$$

where $\boldsymbol{d}_i$ is the i-th column of D. Applying the Karush-Kuhn-Tucker first-order optimality conditions [14] to J, we derive

$$\boldsymbol{s}^l \geq 0, \frac{\partial J}{\partial \boldsymbol{s}^l} \geq 0, \boldsymbol{s}^l \circ \frac{\partial J}{\partial \boldsymbol{s}^l} = 0, \tag{17}$$

where $\circ$ is the element-wise multiplication between two matrices of the same dimension. This leads to the following update rule:

$$\boldsymbol{s}_i^l \leftarrow \boldsymbol{s}_i^l \circ \frac{R^T\boldsymbol{r}_i}{\left[R^T R(\boldsymbol{s}_i^l + \boldsymbol{s}_i^g) + \boldsymbol{d}_i + \beta\boldsymbol{s}_i^l\right]}, \tag{18}$$

where $\frac{[\cdot]}{[\cdot]}$ denotes the element-wise matrix division operator. The update rule for $\boldsymbol{s}^g$ can be similarly derived:

$$\boldsymbol{s}_i^g \leftarrow \boldsymbol{s}_i^g \circ \frac{R^T\boldsymbol{r}_i}{\left[R^T R(\boldsymbol{s}_i^l + \boldsymbol{s}_i^g) + \alpha + \beta\boldsymbol{s}_i^g\right]}. \tag{19}$$

We summarize the resulting algorithm in Algorithm 1.

Time complexity of Algorithm 1. For optimizing $\boldsymbol{s}^l$ and $\boldsymbol{s}^g$, we compute $R^T R$ and $R^T\boldsymbol{r}_i$ in an offline fashion. Due to the sparsity of $R^T R$, updating of each iteration by Eq. (18) and (19) has complexity of $O(nz)$, where z is the average number of non-zeros in the rows of $R^T R$.

When optimizing F, we only need the c eigenvectors corresponding to the c smallest eigenvalues, with complexity $O(n^2c)$. This is superior compared with clustering-based methods since applying clustering on rating matrix R has complexity $O(mnc)$, which is prohibitive with a large number

Algorithm 1: Alternating minimization

while not converge **do**
  $F \leftarrow c$ eigenvectors corresponding to the c smallest eigenvalues;
  while not converge **do**
    for $i = 1, \ldots n$ **do**
      $\boldsymbol{s}_i^l \leftarrow \boldsymbol{s}_i^l \circ \frac{R^T \boldsymbol{r}_i}{\left[R^T R(\boldsymbol{s}_i^l + \boldsymbol{s}_i^g) + \boldsymbol{d}_i + \beta \boldsymbol{s}_i^l\right]}$;
      $\boldsymbol{s}_i^g \leftarrow \boldsymbol{s}_i^g \circ \frac{R^T \boldsymbol{r}_i}{\left[R^T R(\boldsymbol{s}_i^l + \boldsymbol{s}_i^g) + \alpha + \beta \boldsymbol{s}_i^g\right]}$;

of users. Besides, packages like ARPACK[1] provide additional benefit to calculate the eigenvectors when S^l is sparse, which can further reduce the complexity in optimizing F.

Convergence analysis of Algorithm 1. We prove that the alternating minimization optimization in Algorithm 1 will converge. We first show that the update rule for $\boldsymbol{s}_i^l$ ensures convergence. The convergence of $\boldsymbol{s}_i^g$ can be proved in a similar manner.

Theorem 4.1. *The objective function J in Eq. (15) is non-increasing under the update rule Eq. (18). J is invariant under the update rule if and only if $\boldsymbol{s}_i^l$ is at a stationary point.*

Proof. The objective function J in Eq. (15) is bounded from below by zero. We only need to show that the objective J is non-increasing under the update rule Eq. (18). We follow a similar procedure as described in [5] based on auxiliary functions. We write $J_j(\boldsymbol{s})$, $J_j'(\boldsymbol{s})$, and $J_j''(\boldsymbol{s})$ for the objective function, the first and second order derivatives of J over the j-th element of $\boldsymbol{s} \in \mathbb{R}^n$:

$$J_j(\boldsymbol{s}) = \frac{1}{2}\|\boldsymbol{r}_i - R(\boldsymbol{s} + \boldsymbol{s}_i^g)\|_2^2 + \frac{\beta}{2} s_j^2 + \lambda d_{ij} s_j, \tag{20}$$

where s_j is the j-th element of $\boldsymbol{s}$. $J_j'(\boldsymbol{s})$ and $J_j''(\boldsymbol{s})$ can be written as:

$$J_j'(\boldsymbol{s}) = \left[R^T R(\boldsymbol{s} + \boldsymbol{s}_i^g) - R^T \boldsymbol{r}_i\right]_j + \beta s_j + \lambda d_{ij} \tag{21}$$

$$J_j''(\boldsymbol{s}) = \left[R^T R\right]_{jj} + \beta. \tag{22}$$

The auxiliary function is defined as:

$$G(s_j, s_j^0) = J_j(\boldsymbol{s}^0) + J_j'(\boldsymbol{s}^0)(s_j - s_j^0) + \frac{\left[R^T R(\boldsymbol{s}^0 + \boldsymbol{s}_i^g)\right]_j + \beta s_j^0 + d_{ij}}{2 s_j^0}(s_j - s_j^0)^2. \tag{23}$$

We show that the minimization based on the auxiliary function is equivalent to the update rule in Eq. (18):

$$s_j^1 = \arg\min_{s_j} G(s_j, s_j^0) = s_j^0 - s_j^0 \frac{J_j'(\boldsymbol{s}^0)}{\left[R^T R(\boldsymbol{s}^0 + \boldsymbol{s}_i^g)\right]_j + \beta s_j^0 + d_{ij}} = s_j^0 \cdot \frac{\left[R^T \boldsymbol{r}_i\right]_j}{\left[R^T R(\boldsymbol{s}^0 + \boldsymbol{s}_i^g)\right]_j + \beta s_j^0 + d_{ij}}. \tag{24}$$

We then write $J_{ij}(s)$ by a Taylor series expansion:

$$J_j(\boldsymbol{s}) = J_j(\boldsymbol{s}^0) + J_j'(\boldsymbol{s}^0)(s_j - s_j^0) + \frac{1}{2} J_j''(\boldsymbol{s}^0)(s_j - s_j^0)^2. \tag{25}$$

[1] https://www.caam.rice.edu/software/ARPACK/

It is immediate that $G(s_j^0, s_j^0) = J_j(\mathbf{s}^0)$. To prove $G(s_j, s_j^0) \geq J_j(\mathbf{s})$, we need to show:

$$\frac{\left[R^T R(\mathbf{s}^0 + \mathbf{s}_i^g)\right]_j + \beta s_j^0 + d_{ij}}{s_j^0} \geq \left[R^T R\right]_{jj} + \beta, \tag{26}$$

which immediately holds as $d_{ij} \geq 0$. Thus we have:

$$J_j(\mathbf{s}^1) \leq G(s_j^1, s_j^0) \leq G(s_j^0, s_j^0) = J_j(\mathbf{s}^0). \tag{27}$$

Therefore, we have shown that $\forall j$, $J_j(\mathbf{s})$ is non-increasing under the update rule. The equal sign in Eq. (27) holds if and only if $s_j^1 = s_j^0$, which indicates that J is invariant under the update rule if and only if $\mathbf{s}_i^l$ is at a stationary point. □

Theorem 4.1 guarantees the convergence of $\mathbf{s}^l$ under the update rule in Eq. (18). The convergence of $\mathbf{s}^g$ can be similarly guaranteed. We write $S = \left\{S^l, S^g\right\}$ for the combination of S^l and S^g. We write $J(F, S)$ as the objective function of Eq. (12). Thus $S^{(t+1)}$ is optimal w.r.t. $J(F^{(t+1)}, S)$. We then prove the convergence of Algorithm 1.

Theorem 4.2. *The sequence $\left\{S^{(t)}, F^{(t)}\right\}$ generated by Algorithm 1 has at least one limit point. Any limit point $\{S^*, P^*\}$ is a stationary point of Eq. (12).*

Proof. As $F^{(t+1)}$ and $S^{(t+1)}$ are optimal w.r.t. $J(F, S^{(t)})$ and $J(F^{(t+1)}, S)$, and S is β-strongly convex w.r.t. $J(S, F^{(t+1)})$, according to Eq. (22), we have

$$\begin{aligned} J(F^{(t+1)}, S^{(t+1)}) &\leq J(F^{(t+1)}, S^{(t)}) - \frac{\beta}{2}\|S^{(t+1)} - S^{(t)}\|_F^2 \\ &\leq J(F^{(t)}, S^{(t)}) - \frac{\beta}{2}\|S^{(t+1)} - S^{(t)}\|_F^2. \end{aligned} \tag{28}$$

Summing over Eq. (28), we have:

$$\sum_{t=1}^{+\infty} \frac{\beta}{2}\|S^{(t+1)} - S^{(t)}\|_F^2 \leq J(S^{(0)}, F^{(0)}), \tag{29}$$

which implies

$$S^{(t+1)} - S^{(t)} \to 0. \tag{30}$$

Based on Eq. (30), as $F^{(t+1)}$ is obtained as the c eigenvectors of $L_{S^{(t)}}$, we have:

$$F^{(t+1)} - F^{(t)} \to 0. \tag{31}$$

Therefore, the sequence $\left\{S^{(t)}, F^{(t)}\right\}$ has at least one limit point. According to [23, Corollary 2], any limit point of the sequence is a stationary point of Eq. (12). □

5 EXPERIMENTAL SETUP

In this section, we introduce our experimental setup.

5.1 Research questions

Our research questions are:

(RQ1) What is the overall performance of BISM in comparison to state-of-the-art linear and neural-based methods for top-N recommendation?

(RQ2) How do BISM and the baselines perform when recommending different numbers of items to users?

(RQ3) What is the impact of BDR on the learned item similarity matrix and the performance of top-N recommendation?

Table 2. Descriptive statistics of the datasets: #user, #item and #rating denote the number of users, items and ratings, respectively. Density is calculated as #rating/(#user×#item).

Name	#user	#item	#rating	Density
Amazon	5,653	11,944	86,149	0.13%
BookX	5,671	5,367	86,354	0.28%
Yahoo	7,594	8,641	106,593	0.16%
MovieLens	6,040	3,706	1,000,209	4.47%
Pinterest	55,187	9,916	1,500,809	0.27%

(RQ4) What is the impact of regularization parameters on the performance of BISM?

5.2 Datasets

We evaluate the performance of BISM on five benchmark datasets. Table 2 lists descriptive statistics of the datasets.

- *Amazon*:[2] A dataset based on the Amazon product catalogue [40]; we select one of the categories, Sports & Outdoors, which contains transactions between different product items and users indicated with multivariate rating values.
- *BookX*:[3] A subset of the Book-Crossing dataset, containing implicit feedback from users, which was collected by [62] from the Book-Crossing community.
- *Yahoo*:[4] A small sample of the Yahoo!Movies community's preferences for various movies, rated on a scale from A+ to F.

Following the common setting for evaluating top-N recommendation, we binarize the ratings if it is explicit. We also adopt two datasets for a fair comparison against DeepICF and NAIS [25, 57]:

- *MovieLens*:[5] The MovieLens 1M Dataset released by the GroupLens research project.
- *Pinterest*: The implicit feedback data is constructed by [21] for evaluating content-based image recommendation.

5.3 Evaluation methodology

We evaluate the methods using leave-one-out cross-validation (LOOCV): we hold out one interaction of each user as the test data and use the remaining interactions as training set. The validation set consists of a randomly drawn interaction for each user from the training set. This evaluation method is widely utilized for top-N recommendations [3, 27, 46].

To perform top-N recommendation for a user, the widely used method is to rank all items that the user has not rated and the first N items are recommended to her. Since ranking all items can be time-consuming during evaluation, existing work tries to manually and randomly construct a relatively small set of candidate items for each user [25, 26, 57]. While sampling candidate items ensures the efficiency of evaluation, it can introduce randomness for testing [45]. The performance of top-N recommender systems varies when candidate items are constructed differently. Therefore, we first evaluate the top-N recommendation performance by recommending from all unrated items, which is a comparably difficult task. For a fair comparison with the state-of-the-art neural methods, NAIS [25] and DeepICF [57], we also evaluate by sampling 100 candidate items (we use the same candidate set of items as those two papers) to compare BISM.

[2] http://jmcauley.ucsd.edu/data/amazon/
[3] http://www2.informatik.uni-freiburg.de/~cziegler/BX/
[4] https://webscope.sandbox.yahoo.com/catalog.php
[5] https://grouplens.org/datasets/movielens/

We use hit rate (HR) and average reciprocal hit-rank (ARHR) [17, 17, 30] to evaluate the performance:

$$HR = \frac{\#hit}{\#user}, \quad ARHR = \frac{1}{\#user}\sum_{i=1}^{\#hit}\frac{1}{pos(i)}. \tag{32}$$

where #users is the total number of users, #hits is the number of hits in the top-N recommendations across all users, and $pos(i)$ is the position of the test item in the ranked list of recommendations for the i-th hit. ARHR is a weighted version of HR, which takes the ranking position of the test item i in the list of recommendations into account. Note that when evaluating using LOOCV, HR and ARHR can be regarded as Recall and mean reciprocal rank (MRR), respectively. We also use normalized discounted cumulative gain (nDCG) [25] as evaluation metric:

$$DCG@N = \sum_{i=1}^{N}\frac{rel_i}{\log_2(i+1)},$$

where rel_i indicates whether the item at position i is relevant. The objective of nDCG is to compare any given ranked list of items with a benchmark that represents the optimal ranking:

$$nDCG@N = \frac{DCG@N}{IDCG@N},$$

where the idealized discounted cumulative gain (IDCG) with cut at N, i.e., the best possible $DCG@N$, is used to normalize discounted cumulative gain (DCG) value so that $nDCG@N$ is within $[0, 1]$.

5.4 Methods used for comparison

Baselines. We compare BISM with the following baselines, including both linear and neural methods.[6] Besides, to show whether learning global similarities is helpful or not, we also implement *localized item similarity model* (LISM), the simplified version of BISM, which learns local similarities only.

- *Bayesian personalized ranking* (BPR) [46]: A ranking/retrieval criteria-based method. We train a latent space model with the pair-wise loss function;
- *Factored item similarity model* (FISM) [30]: An ISM that factorizes item similarity matrix into two low-rank matrices. We use the implementation in [25] for the experiments, which takes advantage of advanced learning algorithms.
- *Item-based k nearest neighbors approach* (item$k$NN) [17]: An early ICF method that heuristically computes item similarities. We choose cosine as the similarity function and apply shrinkage to the similarities;
- *Sparse linear method* (SLIM) [42]: An ISM that learns a sparse item similarity matrix;
- SLIM_{local}: A SLIM with item clustering. Items are clustered into c groups based on the rating matrix, where for each group we learn a local SLIM. For each user, the predicted score of a target item is calculated by the local SLIM of the group that the item belongs to.
- *Embarrassingly shallow autoencoders* (EASE) [49]: a simplified version of SLIM. EASE drops the ℓ_1-norm and the non-negative constraint in SLIM. Due to the simplification, closed-form solution is available for EASE;

[6] We exclude LorSLIM [10], the low-rank sparse linear model, from our experimental comparisons. We failed to generate a set of reasonable recommendations using LorSLIM on all datasets and we were also unable to reproduce the results obtained using LorSLIM as reported in [10]. The source code of LorSLIM on MovieLens with 100k ratings (ML-100k) is evaluated with 336 items, rather than all 1,682 items. For a fair comparison, we evaluate BISM in the same setting, which provides much better results than reported in their paper, i.e., HR@10 = 0.574, ARHR@10 = 0.265 against HR@10 = 0.397, ARHR@10 = 0.207. A similar issue exists with the lorSLIMappro method proposed in [31], which approximates the nuclear norm used in lorSLIM. Therefore, we exclude the two methods from our experiments.

- *Pure Singular-Value-Decomposition* (PureSVD) [15]: A latent space model designed for top-N recommendation;
- *Weighted regularized matrix factorization* (WRMF) [29]: A latent space model specially for implicit datasets;
- *Multinomial variational auto-encoder* (mVAE) [37]: A state-of-the-art neural method for top-N recommendation. It utilizes variational auto-encoder (VAE) and assume multinomial likelihood function to capture implicit feedback;
- *Neural attentive item similarity model* (NAIS) [25]: A neural-based ISM that utilizes the attention mechanism to capture similarities between the target item and user rated items. We compare with both implementations with different choices of attention function. NAIS$_{concat}$ denotes the use of f_{concat}, which simply concatenates $\boldsymbol{p}_i$ and $\boldsymbol{q}_j$ to learn the attention weight w_{ij}. NAIS$_{prod}$ denotes the use of f_{prod}, which feeds the element-wise product of $\boldsymbol{p}_i$ and $\boldsymbol{q}_j$ into the attention network.
- *Deep item-based collaborative filtering* (DeepICF) [57]: A neural-based ISM that accounts for the nonlinear and higher-order relationships among items;

Implementation details. We use LibRec [24] to run the experiments for item$k$NN, SLIM, BPR and WRMF. We use the source code implementation in [25] to run experiments for FISM and NAIS[7] (both NAIS$_{concat}$ and NAIS$_{prod}$), the implementation in [37] for mVAE[8] and that in [57] for DeepICF[9]. As shown by [25, 57], both NAIS and DeepICF suffer from slow convergence and poor performance when all model parameters are initialized randomly. Therefore, we follow their solution to pretrain item embeddings of NAIS and DeepICF by FISM. Following the experimental settings of [25, 57], we train NAIS$_{concat}$, NAIS$_{prod}$ and DeepICF with binary cross-entropy loss and the optimizer Adagrad.

We implement BISM and LISM in PyTorch. Instead of following the auto-gradient optimization, we update parameters manually according to the Algorithm 1. We also implement PureSVD and SLIM$_{local}$.

Parameters. The parameters of all methods are explored within the parameter space. We select parameters based on the best performance in terms of HR@10 on the validation set. For BISM we tune the ℓ_1, ℓ_F-norm regularization parameter α, β, block-diagonal regularization parameter λ and the number of item groups c (explored within $\{1, 2, \dots, 10\}$).

The parameters tuned for the baselines are the following: (1) For BPR we tune the parameter of the latent dimension k. (2) For FISM$_{rmse}$ and FISM$_{auc}$ we tune the neighbor agreement parameter α (explored within $\{0.1, 0.2, \dots, 1\}$), ℓ_F-norm regularization parameter β, the ℓ_2-norm regularization of item bias λ and the latent dimension k. (3) For item$k$NN we tune the number of nearest neighbors k. (4) For SLIM we tune the ℓ_1-norm regularization parameter α and the ℓ_F-norm regularization parameter β. (5) For SLIM$_{local}$ we tune the ℓ_1-norm and ℓ_F-norm regularization parameter α and β. (6) For PureSVD we tune the parameter of the latent dimension k. (7) For WRMF we tune the confidence level α (explored within $\{0.1, 0.2, \dots, 1\}$) and the latent dimension k. (8) For DeepICF we choose a three-layer perceptron for the deep neural network structure with $k, 100, 50$ as the number of neurons, where k is also the latent dimension of user/item embeddings. We tune the parameter k of the latent dimension. For DeepICF we tune the neighborhood agreement α. Both α and β are explored within $\{0.1, 0.2, \dots, 1\}$. (9) For mVAE we tune the Kullback–Leibler (KL) term regularization parameter β. (10) For NAIS we follow the paper to fix the neighborhood agreement as $\alpha = 0$, which empirically leads to the best performance. We tune parameter for the latent

[7] https://github.com/AaronHeee/Neural-Attentive-Item-Similarity-Model
[8] https://github.com/dawenl/vae_cf
[9] https://github.com/linzh92/DeepICF

Table 3. Comparison of top-N recommendation methods on Amazon and BookX datasets. The best result is shown in **boldface** and the best result achieved by the baselines (except BISM and LISM) is underlined. We conducted two-sided tests for the null hypothesis that the best and the second best have identical average values. Asterisks indicate the best score if the improvement over the second best is statistically significant; we use an asterisk * if $p < 0.05$ and two asterisks ** if $p < 0.01$.

	Method	α	β	λ	k	c	HR@10	ARHR@10	nDCG@10
Amazon	BPR [46]	–	–	–	500	–	0.0603	0.0238	0.0328
	FISM [25]	0.5	0.01	10	100	–	0.0686	0.0244	0.0346
	item$k$NN [17]	–	–	–	–	10	0.0663	0.0251	0.0355
	SLIM [42]	0.01	1	–	–	–	0.0528	0.0230	0.0298
	SLIM_{local}	10.0	0.01	–	–	5	0.0692	0.0291	0.0384
	PureSVD [15]	–	–	–	20	–	0.0475	0.0171	0.0226
	WRMF [29]	4	–	–	100	–	0.0666	0.0267	0.0360
	EASE [49]	–	100	–	–	–	0.0800	0.0345	0.0451
	DeepICF [57]	0	–	10	100	–	0.0513	0.0176	0.0262
	mVAE [37]	–	0.5	–	–	–	0.0570	0.0222	0.0278
	NAIS_{concat} [25]	0	0.5	10	100	–	0.0402	0.0144	0.0197
	NAIS_{prod} [25]	0	0.5	10	100	–	0.0435	0.0167	0.0228
	LISM	0.01	100	100	–	7	0.0849*	0.0358	0.0472
	BISM	1	100	10	–	9	**0.0867****	**0.0372****	**0.0488****
BookX	BPR [46]	–	–	–	500	–	0.1047	0.0520	0.0564
	FISM [25]	0.5	0.01	10	500	–	0.1095	0.0543	0.0673
	item$k$NN [17]	–	–	–	–	10	0.0908	0.0409	0.0555
	SLIM [42]	0.1	1	–	–	–	0.1135	0.0599	0.0720
	SLIM_{local}	1.0	0.01	–	–	2	0.1001	0.0472	0.0582
	PureSVD [15]	–	–	–	500	–	0.0920	0.0504	0.0610
	WRMF [29]	3	–	–	200	–	0.1126	0.0554	0.0710
	EASE [49]	–	100	–	–	–	0.1247	0.0638	0.0781
	DeepICF [57]	–	–	0.1	500	–	0.0741	0.0324	0.0451
	mVAE [37]	–	0.7	–	–	–	0.0813	0.0388	0.0483
	NAIS_{concat} [25]	–	–	0.1	500	–	0.0779	0.0359	0.0429
	NAIS_{prod} [25]	–	–	0.1	500	–	0.0827	0.0335	0.0446
	LISM	0.01	100	10	–	2	0.1315**	0.0654*	0.0809**
	BISM	10	100	10	–	6	**0.1333****	**0.0663****	**0.0819****

dimension k and set the attention factor $a = k$. We tune k, the latent dimension (or the number of neighbors), from $\{10, 20, 50, 100, 200, 500\}$. All the parameters for regularization are explored from $\{0.01, 0.1, 1, 10, 100\}$.

6 EXPERIMENTAL RESULTS

We answer the research questions listed in Section 5.1 based on the experimental results.

6.1 RQ1: Top-N recommendation performance

To answer RQ1, we compare BISM with state-of-the-art baselines, both linear and neural. The overall results of all methods on the Amazon, BookX, MovieLens and Yahoo datasets are reported in

Table 4. Comparison of top-N recommendation methods on MovieLens and Yahoo datasets.

	Method	α	β	λ	k	c	HR@10	ARHR@10	nDCG@10
MovieLens	BPR [46]	–	–	–	500	–	0.2353	0.0977	0.1284
	FISM [25]	0.5	1	10	100	–	0.2001	0.0725	0.0994
	item$k$NN [17]	–	–	–	–	200	0.1740	0.0705	0.0944
	SLIM [42]	0.01	1	–	–	–	0.2122	0.0907	0.1289
	SLIM$_{local}$	0.1	10	–	–	2	0.2334	0.0990	0.1304
	PureSVD [15]	–	–	–	–	20	0.2142	0.0920	0.1219
	WRMF [29]	2	–	–	–	20	0.2339	0.0967	0.1331
	EASE [49]	–	100	–	–	–	0.2542	0.1093	0.1431
	DeepICF [57]	–	–	0.1	100	–	0.2382	0.0968	0.1298
	mVAE [37]	–	0.9	–	–	–	0.2318	0.0926	0.1219
	NAIS$_{concat}$ [25]	–	–	0.1	100	–	0.2139	0.0831	0.1100
	NAIS$_{prod}$ [25]	–	–	0.1	100	–	0.2172	0.0872	0.1183
	LISM	0.01	100	1	–	7	0.2571	**0.1107**	0.1437
	BISM	0.1	100	10	–	1	**0.2602***	0.1104	**0.1460**
Yahoo	BPR [46]	–	–	–	500	–	0.3460	0.1675	0.2055
	FISM [25]	0.5	1	1	50	–	0.2541	0.1167	0.1486
	item$k$NN [17]	–	–	–	–	500	0.3368	0.1611	0.2038
	SLIM [42]	0.01	1	–	–	–	0.3934	0.2011	0.2479
	SLIM$_{local}$	1.0	0.10	–	–	2	0.3791	0.1910	0.2352
	PureSVD [15]	–	–	–	–	10	0.2385	0.1029	0.1307
	WRMF [29]	6	–	–	–	20	0.3458	0.1574	0.2031
	EASE [49]	–	100	–	–	–	0.4076	0.2089	0.2555
	DeepICF [57]	–	–	1	50	–	0.3069	0.1343	0.1735
	mVAE [37]	–	0.9	–	–	–	0.3745	0.1762	0.2065
	NAIS$_{concat}$ [25]	–	–	1	50	–	0.3028	0.1379	0.1668
	NAIS$_{prod}$ [25]	–	–	1	50	–	0.3080	0.1385	0.1681
	LISM	0.01	100	1	–	5	0.4058	0.2058	0.2515
	BISM	0.1	100	0.1	–	6	**0.4101**	**0.2091**	**0.2560**

Table 3 and 4. In both tables, we report and compare HR@10, ARHR@10 and nDCG@10. For each method, the results and the parameter settings corresponding to the best HR@10 on the validation set are reported.

We discuss the results per dataset. First, the Amazon dataset has the largest number of items, the smallest number of users, and the most sparse feedback. Therefore, the overall accuracy for the Amazon dataset is low. EASE is the best performing baseline. BISM and LISM outperform EASE and the difference with BISM is significant. Besides, SLIM$_{local}$ also shows good performance. While SLIM is outperformed by FISM, SLIM$_{local}$ beats FISM by clustering items. Therefore, the effectiveness of capturing latent item groups is well confirmed on the Amazon dataset. The neural-based methods generally show poor performance on this dataset. The best performance is achieved by DeepICF, which is outperformed by SLIM.

Second, on the BookX dataset, while results are similar to Amazon, the overall performance is better. While SLIM outperforms FISM and WRMF, its performance is still inferior to that of

Table 5. Top-N recommendation from 100 candidate items of compared methods at embedding size 16.

Method	MovieLens		Pinterest	
	HR@10	nDCG@10	HR@10	nDCG@10
FISM [25]	0.6647	0.3949	0.8740	0.5522
NAIS_{concat} [25]	0.6972	0.4196	0.8844	0.5720
NAIS_{prod} [25]	0.6969	0.4194	**0.8844**	**0.5722**
DeepICF [57]	0.6881	0.4113	0.8806	0.5631
EASE [49]	0.7096	**0.4495**	0.8150	0.5439
LISM	0.7146	0.4445	0.8648	0.5581
BISM	**0.7190**	0.4459	0.8702	0.5632

EASE. Although SLIM_{local} fails to perform better than SLIM, both BISM and LISM show significant improvement over SLIM. This shows that while capturing latent item groups is helpful for recommendation, the generated item similarity matrix via the static way of clustering is sub-optimal or even harmful. Again, the neural-based methods show poor performance. The effectiveness of neural-based methods is conditioned on the number of training samples. However, both Amazon and BookX datasets are very sparse, which means that they are less qualified to train these complex models.

Next, the overall performance on the MovieLens dataset is high since this dataset has the least sparse ratings. While EASE is still the best performed baseline, the second best performed baseline is the neural model DeepICF. Due to dense ratings, this is the only case when the neural-based methods can outperform linear ones. Again, BISM and LISM improves over DeepICF and EASE and the improvement w.r.t. HR@10 is significant. And finally, on the Yahoo dataset, while it is also relatively sparse, the overall performance is the best among all the datasets. The superiority of ISMs is clearly visible on this dataset. SLIM outperforms mVAE substantially (5.0% w.r.t. HR@10 and 14.1% w.r.t. ARHR@10), and BISM improves over SLIM significantly (4.5% w.r.t. HR@10 and 3.8% w.r.t. ARHR@10). Although BISM still performs better than EASE, the improvement is not significant.

The experiments discussed so far show that linear methods generally show better performance than neural methods for top-N recommendation. The poor performance of mVAE can be explained by the suggestion in [49] that the zero constraint of the diagonal of item similarities may be more effective on sparse data than neural methods. For other neural methods (NAIS and DeepICF), where the zero constraint has been considered, we conduct further experiments to analyze their relatively poor performance. For a fair comparison, we follow the same experimental settings used for NAIS and DeepICF. To be more specific, rather than ranking all items to perform the top-N recommendation, we follow the setting of sampling 99 negative items together with 1 positive item for a user to form the candidate items. We run experiments on the two datasets (MovieLens and Pinterest). The authors open-source the two datasets, the split and the sampled negative items. Our comparison can therefore be conducted under the exact same experimental settings. We run BISM and LISM on these two datasets and compare with the results reported in [25, 57]. Since EASE is the best performing baseline, it is also taken as a baseline to compare.

Results are recorded in Table 5. On the MovieLens dataset, the effectiveness of neural methods is clearly demonstrated. However, they fail to outperform EASE. While BISM and LISM reach higher HR@10 scores than EASE, in terms of nDCG@10 EASE performs slightly better. On the Pinterest dataset, however, EASE cannot achieve comparable performance. While BISM and LISM perform better than EASE, they also fail to outperform neural methods, except that BISM beats DeepICF w.r.t. nDCG@10.

We summarize the above experimental analysis and conclude as follows: (1) the effectiveness of BDR is well demonstrated since BISM and LISM outperform baseline methods on all datasets except for the Pinterest and the outperformance is significant generally; (2) comparing with the state-of-the-art linear baseline method EASE, BISM show better performance in most cases, which further confirms the effectiveness of BDR; and (3) neural methods show their effectiveness on the MovieLens and Pinterest datasets, which have comparably more data samples, indicating that neural methods generally require more data to be well trained;

6.2 RQ2: Top-N recommendation with different N

To better illustrate the gains achieved by BISM over competing approaches, we show the HR and ARHR scores of all algorithms for different values of N (i.e., 5, 10, 15, 20) on the Amazon, BookX, MovieLens and Yahoo datasets. For ease of illustration, we separate the comparison of BISM with linear and with neural methods. Figures 3 and 4 show the comparison of results. Overall, BISM consistently outperforms other methods w.r.t. all metrics and on all datasets.

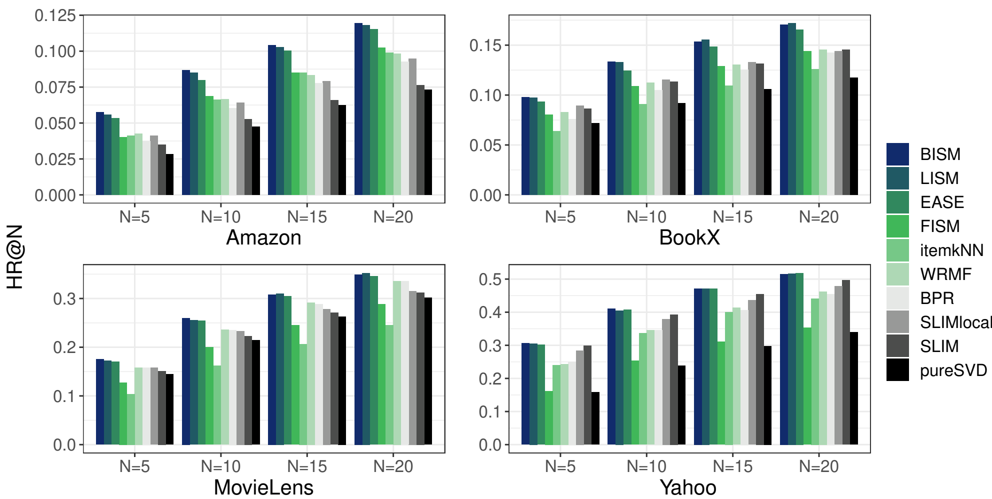
Figure 3 compares BISM with linear methods. We discuss the results per dataset: (1) As shown in Figure 3(a) (top-left), on the Amazon dataset, BISM performs the best, followed by LISM and EASE. Besides, FISM, item*k*NN and WRMF show similar performance. While FISM is inferior than item*k*NN and WRMF when $N = 5$, it outperforms item*k*NN and WRMF when N is larger. As for ARHR@N in Figure 3(b) (top-left), item*k*NN and WRMF outperform FISM constantly. The superiority of BDR is well demonstrated since both BISM and LISM improve over other compared methods, regardless of the length of recommendation lists. (2) On the BookX dataset, as shown in Figure 3(a) (top-right), while both BISM and LISM outperform other methods, BISM is outperformed by LISM when $N = 15$ and 20. Besides, SLIM and LorSLIM show their effectiveness. Except for $N = 20$, they outperform other methods except BISM and LISM. This trend can also be observed for ARHR@N in Figure 3(b) (top-right). (3) On the MovieLens dataset (Figure 3(a) and 3(b) (bottom-left)), while the best results are also generated by BISM, LISM and EASE, comparable results are achieved by WRMF, BPR, SLIM, LorSLIM and PureSVD. Besides BISM, LISM and EASE, WRMF and BPR achieves the best result on HR@N and ARHR@N, respectively. (4) On the Yahoo dataset, results of BISM, LISM and EASE are similar, followed by SLIM, which outperforms other baselines.

Figure 4 compares BISM with neural methods. The superiority of BISM and LISM are better illustrated, especially w.r.t. ARHR@N. We also discuss the results per dataset: (1) On the Amazon dataset, besides BISM and LISM, DeepICF performs the best w.r.t. HR@N (the top-left of Figure 4(a)) and NAIS_{prod} performs the best w.r.t. ARHR@N (the top-left of Figure 4(b)). (2) On the BookX dataset, while mVAE is significantly outperformed by BISM and LISM, it outperforms other neural methods (the top-left of Figure 4(a) and 4(b)). (3) On the MovieLens dataset, DeepICF and mVAE show comparable performance and outperform other methods, though still being outperformed by BISM and LISM. (4) On the Yahoo dataset, mVAE is a promising method. When $N = 20$, mVAE can achieve comparable results to BISM and LISM. While BISM performs better than LISM w.r.t. HR@5 and HR@10, it has been outperformed by LISM w.r.t. other metrics.

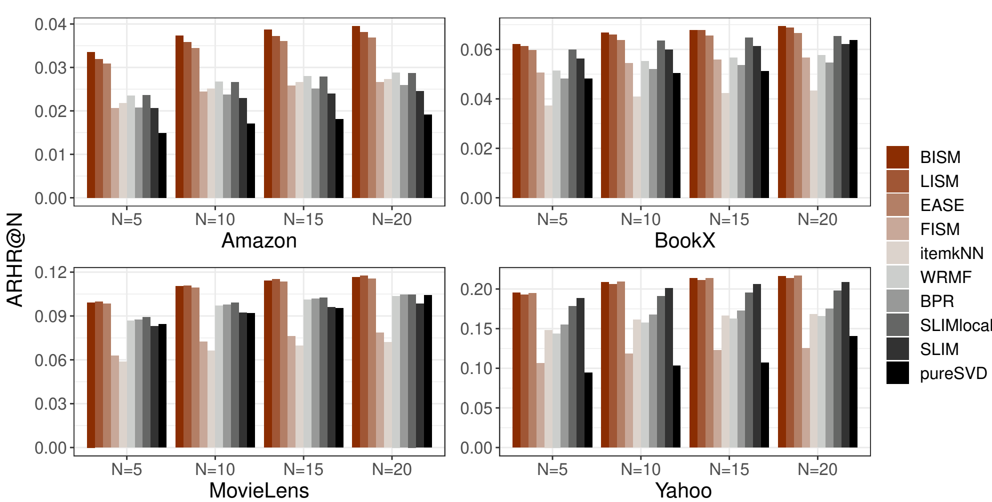
To summarize, despite the differences shown when performing top-N recommendation with different values of N, methods with BDR (BISM and LISM) always generate better results regardless of metrics and the length of the list of recommended items.

6.3 RQ3: Impact from the latent item groups

We further analyze the impact of BDR on the retrieval performance to answer RQ4. We first conduct a qualitative evaluation. We visualize the learned item similarity matrix by different



(a) Performance of N = 5, 10, 15, 20 w.r.t. HR@N



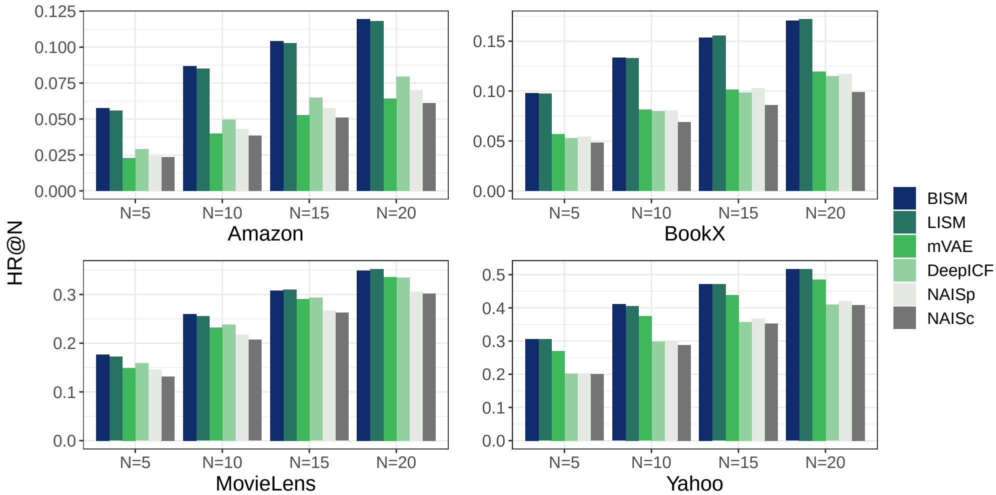
(b) Performance of N = 5, 10, 15, 20 w.r.t. ARHR@N

Fig. 3. Comparison with linear methods for different values of N.

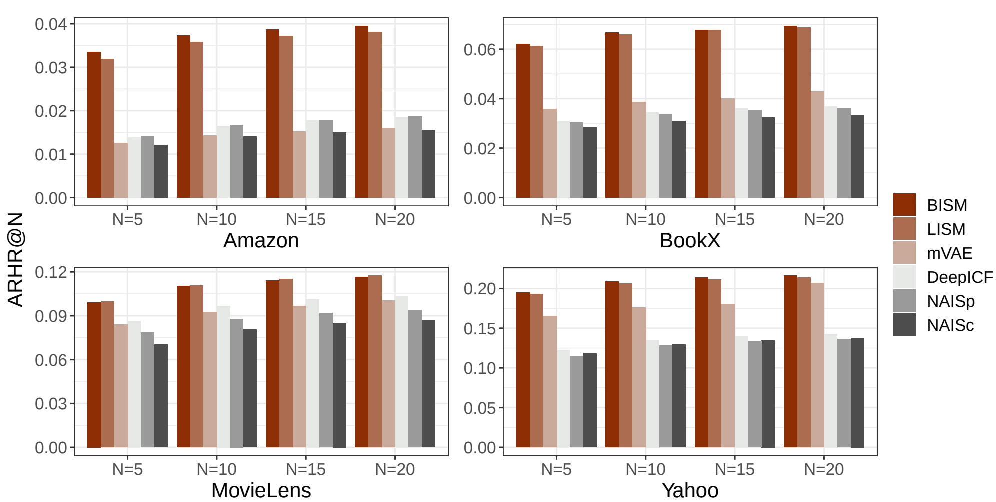
models. However, in real applications, the block-diagonal structure is not easily visible. Therefore, we consider the qualitative evaluation on a smaller dataset, i.e., ML-100k[10].

To see the structure difference of item similarity matrix by different ISMs, we visualize the matrix in Figure 5, using ML-100k dataset. As shown by Figure 5(a), SLIM cannot capture latent item groups as the structure of item similarity matrix is not block-diagonal. The matrix learned by LorSLIM captures the main component in the top left of Figure 5(b), which can be regarded as a single block. Besides the block discovered by LorSLIM, LISM further captures a block in the bottom right of Figure 5(c), within which the transitive relations have also been recovered.

[10]https://grouplens.org/datasets/movielens/100k/



(a) Performance of $N = 5, 10, 15, 20$ w.r.t. HR@N

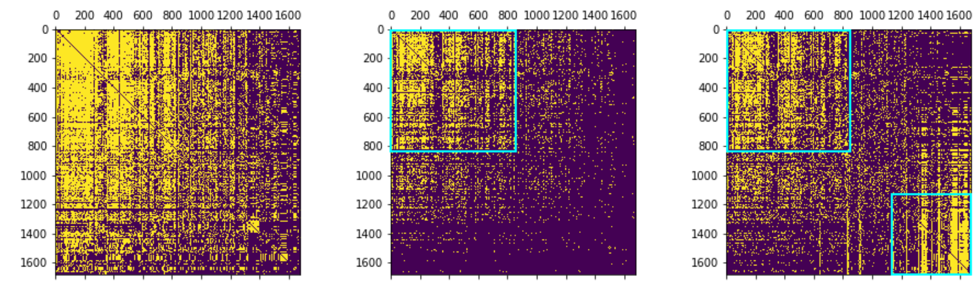


(b) Performance of $N = 5, 10, 15, 20$ w.r.t. ARHR@N

Fig. 4. Comparison with neural methods for different values of N.

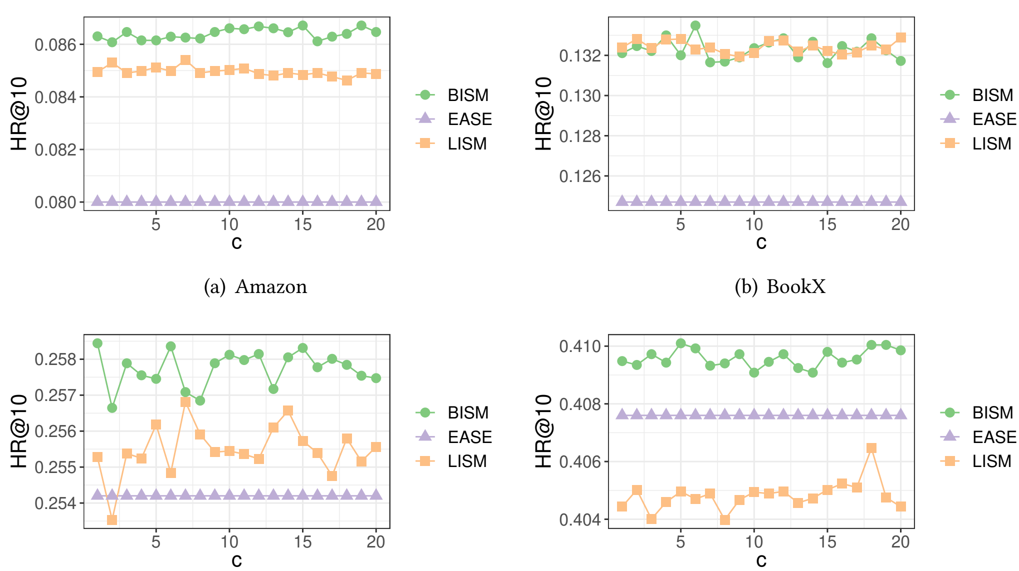
To further see the impact of BDR on the top-N recommendation performance, we also evaluate BISM and LISM with different values of c. EASE is taken as the baseline for compare, which does not consider item grouping. The results are plotted as line-point figures in Figure 6, where we use the same parameter settings as Table 3 and 4 but vary c from 1 to 20 with step 1. Clearly, learning different numbers of item groups has an impact on the performance of top-N recommendation.

We find the following. (1) On the Amazon dataset, as shown in Figure 6(a), LISM outperforms EASE significantly and BISM further improves over LISM. While LISM shows better performance when c is small, BISM performs better when c is large. This is due to the learning of global similarities, which overcomes the negative effect of BDR. The effectiveness of item grouping is well demonstrated on the Amazon dataset, which has the largest candidate items. (2) On the BookX



(a) SLIM (b) lorSLIM (c) LISM with $c = 2$

Fig. 5. Item Similarity Matrix on ML-100k.



(a) Amazon (b) BookX

(c) MovieLens (d) Yahoo

Fig. 6. Top-N recommendation performance when learning different numbers of item groups.

dataset, as shown in Figure 6(b), while both LISM and BISM outperform EASE significantly, LISM and BISM show similar performance, except that BISM reaches the top when $c = 6$. The negative effect of BDR is not shown on the BookX dataset, indicating that the intrinsic number of item groups may be large. (3) On the MovieLens dataset, as shown in Figure 6(c), the value of c shows greater impact on the performance. Both LISM and BISM are unstable, especially when c is small. BISM gradually stabilizes with the growth of c, whereas LISM keeps fluctuating. BDR has a higher impact on LISM and BISM in the MovieLens dataset. This may be that the MovieLens dataset has the least number of items, resulting in the sensitivity to item grouping. (4) On the Yahoo dataset, as shown in Figure 6(d), LISM fail to outperform EASE due to the negative effect of BDR. By learning global similarities, the negative effect can be overcome, where BISM outperforms EASE.

To conclude, on different datasets, the number of item groups has various impact. The performance of BISM and LISM vary with different number of latent item groups. BISM generally shows

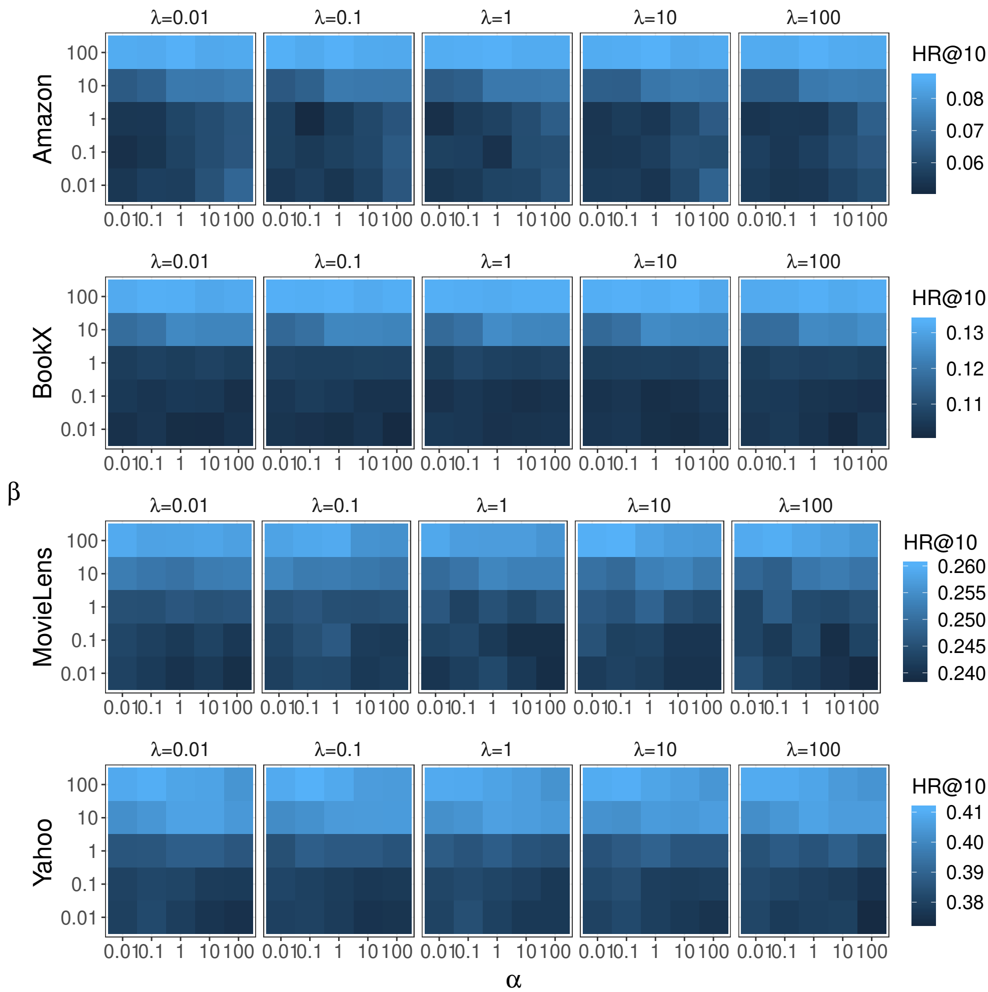


Fig. 7. Impact of block-aware similarity regularizations. The color intensity corresponds to HR@10.

better and more stable performance than LISM by also learning global similarities. Compared with BISM, which is less sensitive to c, we need to carefully tune c to reach the peak performance of LISM. Compared with $c = 1$, $c > 1$ generally leads to better performance, which means that capturing multiple latent item groups helps to improve performance.

6.4 RQ4: The effect of regularization

Finally, we evaluate the impact of regularization parameters on the performance of BISM. Recall that the learning process for BISM is controlled by several regularization terms. To avoid the impact from the number of item groups in this experiments, we fix $c = 5$ for all datasets and perform a grid-search of the parameters α, β and λ that control the ℓ_1, ℓ_F-norm regularizations and BDR, respectively. We visualize the results with heat maps in Figure 7, where α is shown on the x-axis and β on the y-axis, and different settings of λ are shown with different facets.

Specifically, for the Amazon dataset (the first row of Figure 7), as we have mentioned, when $\beta = 100$, the performance of BISM is insensitive to α and λ. However, when β is relatively small, larger value of α generally leads to better performance. BDR can show its effectiveness on the Amazon dataset as long as we set larger value for β. A similar result is shown on the BookX dataset (the second row of Figure 7), which also prefers larger value of β but BISM is less sensitive to λ, the best value of α is around 1. Similar heat map distributions are shown for the MovieLens dataset (the third row of Figure 7). Different from the Amazon and BookX datasets, on the MovieLens dataset the performance of BISM varies slightly when we change the value of λ: a large value λ with a small value of α or a small value of λ with a large value of α generally works better. This is understandable because both α and λ control the regularization of sparsity. The last row of Figure 7 of the Yahoo dataset shows a small difference compared with the Amazon, BookX and MovieLens datasets. BISM shows better performance when $\beta = 10$ instead of $\beta = 100$.

In short, the parameter spaces of all the datasets have shown similar patterns: larger values of β generally lead to better performance and when β is large enough, BISM is insensitive to α and λ. The insensitivity of λ can easily be understood. BDR is dynamically optimized along with the learning of item similarities. This means that no matter what prior value is set for λ, BDR can adapt to the right scale for regularization.

7 RELATED WORK

To better appreciate our research findings, we position them w.r.t. the literature.

7.1 Item collaborative filtering

ICF methods are widely studied for the top-N recommendation. ISMs learn item similarities from data to demonstrate strong performance. Ning and Karypis [42] have proposed SLIM, by learning a sparse item similarity matrix. Low-rankness has been introduced to SLIM in order to recover transitive relations. To achieve low-rankness while ensuring sparsity, Cheng et al. [10] proposed LorSLIM, which introduces a rank regularization term to SLIM. Kang and Cheng [31] improves LorSLIM with a better rank approximation.

However, LorSLIM is challenging to be optimized due to the rank constraint. In comparison, by factorizing item similarity matrix into low-rank matrices, the low-rankness is naturally captured [30, 33]. Kabbur et al. [30] have proposed FISM to factorize the item similarity matrix into two low-dimensional matrices. Due to the successful application of deep learning in information retrieval [51], recent works propose to extend FISM by neural networks. He et al. [25] proposed NAIS to aggregate item similarities by the attention mechanism. Xue et al. [57] studied DeepICF to model non-linear and higher-order relations among items.

A recent trend is to extend linear ICF to non-linear by using auto-encoders. The auto-encoders are item-side: they encode from and decode to user rating vectors of all items, which can be regarded as a generalization of ICF. Wu et al. [53] learn to recover the rating matrix through denoising auto-encoders. Liang et al. [37] introduce variational auto-encoders for top-N recommendations. However, the recommendations generated by these models have weak interpretability. Similar to FISMs, they also failed to achieve sparsity.

Other ICF methods consider different aspects to improve top-N recommendations. Ning and Karypis [43] and Chen et al. [6, 8] utilize side information to overcome rating sparsity. Kang et al. [32] and Hu et al. [28] address rating sparsity for top-N recommendation by leveraging graphs [9]. Wang et al. [52] and Zhao and Guo [61] investigate ranking loss functions.

7.2 Local models

Clustering has been well studied for CF models [4, 7, 22, 35, 44, 56, 58, 60]. These methods cluster users or items based on user ratings into subgroups and estimate a local model for each cluster. Results from all subgroups are aggregated to generate recommendations. Christakopoulou and

Karypis [12] propose local latent factor models, where the assignments of the users to subsets are constantly updated. Wang et al. [50] propose a probabilistic model to cluster items as topics. Wu et al. [54] propose a mixture model to infer memberships of users or items to subgroups. Lee et al. [36] describe an iterative way for estimation where first the latent factors representing the anchor points are estimated and then based on the similarities of observed entries to the anchor points, the latent factors are re-estimated.

A few number of research specifically investigate clustering for ICF methods. Christakopoulou and Karypis [11] explore user subsets to learn user-specific local ISMs, which is combined with a global ISM. Al-Ghossein et al. [2] study online recommendation, where a user's membership is adaptively updated during incremental learning. However, these models only investigate user subsets rather than item groups. Clustering and the estimation of local models in these methods are also treated as separate tasks.

Unlike these methods, we propose to cluster items for ICF. We introduce BDR to encourage a block-diagonal structure to ICF methods, which embeds the clustering into the learning.

7.3 Subspace clustering

Learning block-diagonal representations has originally been studied for subspace clustering [20, 38, 55]. While these methods can be utilized to generate a block-diagonal item similarity matrix, they fail to provide desirable item similarities for the top-N recommendation task. This is because the ultimate goal of these methods is for subspace clustering. These methods rely on the *self-expressiveness* property [20, 38, 55], which states that each data point in a union of subspaces can be well represented by a linear combination of other points in the dataset, i.e., $R = RS$. In comparison, ISMs address top-N recommendation. Rather than perfectly expressing R by RS under the self-expressiveness constraint, ISMs minimize $\|R - \tilde{R}\|_F^2$ and generate the prediction $\tilde{R}$ by RS.

In this paper, we apply BDR to ISMs and propose the BISM. Besides learning block-diagonal representations, BISM improves over these methods for top-N recommendations in the following manner: (1) BISM makes up a combination of local and global similarity matrices to overcome the adversarial effect on top-N recommendation caused by BDR (discussed in Section 3.2); (2) the optimization by these subspace clustering methods requires intermediate terms, which can introduce bias for the learned item similarity matrix; in comparison, BISM directly penalizes the item similarity matrix by BDR;

8 CONCLUSION

In this paper, we have proposed a block-diagonal regularization (BDR) to capture the block-diagonal structure in item similarities for item-based collaborative filtering (ICF) methods, so as to improve the top-N recommendation performance. We have applied BDR to item similarity models (ISMs) and formulate the proposed block-aware item similarity model (BISM), with a theoretical guarantee of block-diagonality, Besides, our method theoretically ensures that the learned item similarities are sparse and capture transitive relations within blocks. Experimental evaluations on a large number of datasets show the effectiveness of BDR for ICF methods.

Despite its effectiveness, one limitation of BDR is that it can only be applied to item similarity model currently. Since item similarity models is not scalable when there is a large number of items, in future work, we will extend BDR to factored item similarity model, which is more scalable.

CODE AND DATA

To facilitate the reproducibility of the reported results, this work only made use of publicly available data and our experimental implementation is publicly available at https://github.com/yifanclifford/BISM.

REFERENCES

[1] Fabio Aiolli. 2013. A Preliminary Study on a Recommender System for the Million Songs Dataset Challenge. In *Proceedings of the 4th Italian Information Retrieval Workshop (IIR '13)*. Pisa, Italy, 73–83. http://ceur-ws.org/Vol-964/paper12.pdf

[2] Marie Al-Ghossein, Talel Abdessalem, and Anthony Barré. 2018. Dynamic Local Models for Online Recommendation. In *Companion of the 27th World Wide Web Conference (WWW '18)*. Lyon, France, 1419–1423. https://doi.org/10.1145/3184558.3191586

[3] Immanuel Bayer, Xiangnan He, Bhargav Kanagal, and Steffen Rendle. 2017. A Generic Coordinate Descent Framework for Learning from Implicit Feedback. In *Proceedings of the 26th International Conference on World Wide Web (WWW '17)*. Perth, Australia, 1341–1350. https://doi.org/10.1145/3038912.3052694

[4] Alex Beutel, Ed Huai-hsin Chi, Zhiyuan Cheng, Hubert Pham, and John R. Anderson. 2017. Beyond Globally Optimal: Focused Learning for Improved Recommendations. In *Proceedings of the 26th International Conference on World Wide Web (WWW '17)*. Perth, Australia, 203–212. https://doi.org/10.1145/3038912.3052713

[5] Deng Cai, Xiaofei He, Xiaoyun Wu, and Jiawei Han. 2008. Non-negative Matrix Factorization on Manifold. In *Proceedings of the 8th IEEE International Conference on Data Mining (ICDM '08)*. IEEE, Pisa, Italy, 63–72. https://doi.org/10.1109/ICDM.2008.57

[6] Yifan Chen, Pengjie Ren, Yang Wang, and Maarten de Rijke. 2019. Bayesian Personalized Feature Interaction Selection for Factorization Machines. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '19)*. ACM, Paris, France, 665–674. https://doi.org/10.1145/3331184.3331196

[7] Yifan Chen, Yang Wang, Xiang Zhao, Hongzhi Yin, Ilya Markov, and Maarten de Rijke. 2020. Local Variational Feature-Based Similarity Models for Recommending Top-*N* New Items. *ACM Trans. Inf. Syst.* 38, 2 (2020), 12:1–12:33. https://doi.org/10.1145/3372154

[8] Yifan Chen, Xiang Zhao, and Maarten de Rijke. 2017. Top-N Recommendation with High-Dimensional Side Information via Locality Preserving Projection. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '17)*. ACM, Shinjuku, Tokyo, Japan, 985–988. https://doi.org/10.1145/3077136.3080697

[9] Yifan Chen, Xiang Zhao, Xuemin Lin, Yang Wang, and Deke Guo. 2019. Efficient Mining of Frequent Patterns on Uncertain Graphs. *IEEE Trans. Knowl. Data Eng.* 31, 2 (2019), 287–300. https://doi.org/10.1109/TKDE.2018.2830336

[10] Yao Cheng, Li'ang Yin, and Yong Yu. 2014. LorSLIM: Low Rank Sparse Linear Methods for Top-N Recommendations. In *Proceedings of the 14th IEEE International Conference on Data Mining (ICDM '14)*. IEEE, Shenzhen, China, 90–99. https://doi.org/10.1109/ICDM.2014.112

[11] Evangelia Christakopoulou and George Karypis. 2016. Local Item-Item Models For Top-N Recommendation. In *Proceedings of the 10th ACM Conference on Recommender Systems (RecSys '16)*. ACM, Boston, MA, USA, 67–74. https://doi.org/10.1145/2959100.2959185

[12] Evangelia Christakopoulou and George Karypis. 2018. Local Latent Space Models for Top-N Recommendation. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (SIGKDD '18)*. ACM, London, UK, 1235–1243. https://doi.org/10.1145/3219819.3220112

[13] Fan RK Chung and Fan Chung Graham. 1997. *Spectral graph theory.* Number 92. American Mathematical Soc.

[14] Andrzej Cichocki, Rafal Zdunek, Anh Huy Phan, and Shun-ichi Amari. 2009. *Nonnegative Matrix and Tensor Factorizations - Applications to Exploratory Multi-way Data Analysis and Blind Source Separation.* Wiley.

[15] Paolo Cremonesi, Yehuda Koren, and Roberto Turrin. 2010. Performance of recommender algorithms on top-n recommendation tasks. In *Proceedings of the 4th ACM Conference on Recommender Systems (RecSys '10)*. ACM, Barcelona, Spain, 39–46. https://doi.org/10.1145/1864708.1864721

[16] Maurizio Ferrari Dacrema, Paolo Cremonesi, and Dietmar Jannach. 2019. Are we really making much progress? A worrying analysis of recent neural recommendation approaches. In *Proceedings of the 13th ACM Conference on Recommender Systems, RecSys 2017, Copenhagen, Denmark, September 16-20, 2019.* 101–109. https://doi.org/10.1145/3298689.3347058

[17] Mukund Deshpande and George Karypis. 2004. Item-based top-N Recommendation Algorithms. *ACM Trans. Inf. Syst.* 22, 1 (Jan. 2004), 143–177. https://doi.org/10.1145/963770.963776

[18] Travis Ebesu, Bin Shen, and Yi Fang. 2018. Collaborative Memory Network for Recommendation Systems. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR 2018, Ann Arbor, MI, USA, July 08-12, 2018.* 515–524. https://doi.org/10.1145/3209978.3209991

[19] Ky Fan. 1949. On a theorem of Weyl concerning eigenvalues of linear transformations I. *Proceedings of the National Academy of Sciences* 35, 11 (1949), 652–655. MISSING

[20] Jiashi Feng, Zhouchen Lin, Huan Xu, and Shuicheng Yan. 2014. Robust Subspace Segmentation with Block-Diagonal Prior. In *Proceedings of the 27th IEEE Conference on Computer Vision and Pattern Recognition (CVPR '14)*. IEEE, Columbus, OH, USA, 3818–3825. https://doi.org/10.1109/CVPR.2014.482

[21] Xue Geng, Hanwang Zhang, Jingwen Bian, and Tat-Seng Chua. 2015. Learning Image and User Features for Recommendation in Social Networks. In *ICCV*. IEEE, 4274–4282. MISSING

[22] Thomas George and Srujana Merugu. 2005. A Scalable Collaborative Filtering Framework Based on Co-Clustering. In *Proceedings of the 5th IEEE International Conference on Data Mining (ICDM '05)*. IEEE, Houston, Texas, USA, 625–628. https://doi.org/10.1109/ICDM.2005.14

[23] Luigi Grippo and Marco Sciandrone. 2000. On the convergence of the block nonlinear Gauss–Seidel method under convex constraints. *Operations research letters* 26, 3 (2000), 127–136. MISSING

[24] Guibing Guo, Jie Zhang, Zhu Sun, and Neil Yorke-Smith. 2015. LibRec: A Java Library for Recommender Systems. In *UMAP (CEUR Workshop Proceedings, Vol. 1388)*. CEUR-WS.org. MISSING

[25] Xiangnan He, Zhankui He, Jingkuan Song, Zhenguang Liu, Yu-Gang Jiang, and Tat-Seng Chua. 2018. NAIS: Neural Attentive Item Similarity Model for Recommendation. *IEEE Trans. Knowl. Data Eng.* 30, 12 (2018), 2354–2366. MISSING

[26] 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)*. Perth, Australia, 173–182. https://doi.org/10.1145/3038912.3052569

[27] Xiangnan He, Hanwang Zhang, Min-Yen Kan, and Tat-Seng Chua. 2016. Fast Matrix Factorization for Online Recommendation with Implicit Feedback. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval, SIGIR 2016, Pisa, Italy, July 17-21, 2016*, Raffaele Perego, Fabrizio Sebastiani, Javed A. Aslam, Ian Ruthven, and Justin Zobel (Eds.). ACM, 549–558. https://doi.org/10.1145/2911451.2911489

[28] Binbin Hu, Chuan Shi, Wayne Xin Zhao, and Philip S. Yu. 2018. Leveraging Meta-path based Context for Top-N Recommendation with A Neural Co-Attention Model. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (SIGKDD '18)*. ACM, London, UK, 1531–1540. https://doi.org/10.1145/3219819.3219965

[29] Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative Filtering for Implicit Feedback Datasets. In *Proceedings of the 8th IEEE International Conference on Data Mining (ICDM '08)*. IEEE, Pisa, Italy, 263–272. https://doi.org/10.1109/ICDM.2008.22

[30] 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 (SIGKDD '13)*. ACM, Chicago, IL, USA, 659–667. https://doi.org/10.1145/2487575.2487589

[31] Zhao Kang and Qiang Cheng. 2016. Top-N Recommendation with Novel Rank Approximation. In *Proceedings of the 2016 SIAM International Conference on Data Mining (SDM '16)*. SIAM, Miami, Florida, USA, 126–134. https://doi.org/10.1137/1.9781611974348.15

[32] Zhao Kang, Chong Peng, Ming Yang, and Qiang Cheng. 2016. Top-N Recommendation on Graphs. In *Proceedings of the 25th ACM International Conference on Information & Knowledge Management (CIKM '16)*. ACM, 2101–2106. https://doi.org/10.1145/2983323.2983649

[33] Yehuda Koren. 2008. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (SIGKDD '08)*. ACM, Las Vegas, Nevada, USA, 426–434. https://doi.org/10.1145/1401890.1401944

[34] Daniel D. Lee and H. Sebastian Seung. 2000. Proceedings of the 14th Advances in Neural Information Processing Systems. In *NIPS (NIPS '00)*. MIT Press, Denver, CO, USA, 556–562. http://papers.nips.cc/paper/1861-algorithms-for-non-negative-matrix-factorization

[35] Joonseok Lee, Samy Bengio, Seungyeon Kim, Guy Lebanon, and Yoram Singer. 2014. Local collaborative ranking. In *Proceedings of the 23rd International World Wide Web Conference (WWW '14)*. Seoul, Republic of Korea, 85–96. https://doi.org/10.1145/2566486.2567970

[36] Joonseok Lee, Seungyeon Kim, Guy Lebanon, Yoram Singer, and Samy Bengio. 2016. LLORMA: Local Low-Rank Matrix Approximation. *J. Mach. Learn. Res.* 17 (2016), 15:1–15:24. MISSING

[37] Dawen Liang, Rahul G. Krishnan, Matthew D. Hoffman, and Tony Jebara. 2018. Variational Autoencoders for Collaborative Filtering. In *Proceedings of the 27th World Wide Web Conference (WWW '18)*. Lyon, France, 689–698. https://doi.org/10.1145/3178876.3186150

[38] C. Lu, J. Feng, Z. Lin, T. Mei, and S. Yan. 2019. Subspace Clustering by Block Diagonal Representation. *IEEE Trans. Pattern Anal. Mach. Intell.* 41, 2 (2019), 487–501. https://doi.org/10.1109/TPAMI.2018.2794348

[39] Can-Yi Lu, Hai Min, Zhong-Qiu Zhao, Lin Zhu, De-Shuang Huang, and Shuicheng Yan. 2012. Robust and Efficient Subspace Segmentation via Least Squares Regression. In *Proceeding of the 12th European Conference on Computer Vision (ECCV '12)*. Florence, Italy, 347–360. https://doi.org/10.1007/978-3-642-33786-4_26

[40] Julian J. McAuley and Jure Leskovec. 2013. Hidden factors and hidden topics: understanding rating dimensions with review text. In *Proceedings of the 7th ACM Conference on Recommender Systems (RecSys '13)*. ACM, Hong Kong, China, 165–172. https://doi.org/10.1145/2507157.2507163

[41] Bojan Mohar. 1991. The Laplacian Spectrum of Graphs. In *Graph Theory, Combinatorics, and Applications*. Vol. 2. Wiley, 871–898. MISSING

[42] Xia Ning and George Karypis. 2011. SLIM: Sparse Linear Methods for Top-N Recommender Systems. In *Proceedings of the 11th IEEE International Conference on Data Mining (ICDM '11)*. IEEE, Vancouver, BC, Canada, 497–506. https://doi.org/10.1109/ICDM.2011.134

[43] Xia Ning and George Karypis. 2012. Sparse linear methods with side information for top-n recommendations. In *RecSys*. ACM, 155–162. https://doi.org/10.1145/2365952.2365983

[44] Mark O'Connor and Jon Herlocker. 1999. Clustering Items for Collaborative Filtering. In *SIGIR workshop on Recommender Systems*. ACM. MISSING

[45] Steffen Rendle. 2019. Evaluation Metrics for Item Recommendation under Sampling. *CoRR* abs/1912.02263 (2019). arXiv:1912.02263 http://arxiv.org/abs/1912.02263

[46] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. In *UAI*. 452–461. MISSING

[47] Francesco Ricci, Lior Rokach, and Bracha Shapira (Eds.). 2015. *Recommender Systems Handbook*. Springer.

[48] Badrul Munir Sarwar, George Karypis, Joseph A. Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th International World Wide Web Conference (WWW '10)*. Hong Kong, China, 285–295. https://doi.org/10.1145/371920.372071

[49] Harald Steck. 2019. Embarrassingly Shallow Autoencoders for Sparse Data. In *Proceedings of the 28th World Wide Web Conference (WWW '19)*. San Francisco, CA, USA, 3251–3257. https://doi.org/10.1145/3308558.3313710

[50] Keqiang Wang, Wayne Xin Zhao, Hongwei Peng, and Xiaoling Wang. 2016. Bayesian Probabilistic Multi-Topic Matrix Factorization for Rating Prediction. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI '16)*. New York, NY, USA, 3910–3916. MISSING

[51] Yang Wang, Xuemin Lin, Lin Wu, and Wenjie Zhang. 2017. Effective Multi-Query Expansions: Collaborative Deep Networks for Robust Landmark Retrieval. *IEEE Trans. Image Processing* 26, 3 (2017), 1393–1404. https://doi.org/10.1109/TIP.2017.2655449

[52] Zengmao Wang, Yuhong Guo, and Bo Du. 2018. Matrix completion with Preference Ranking for Top-N Recommendation. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence (IJCAI '18)*. Stockholm, Sweden, 3585–3591. https://doi.org/10.24963/ijcai.2018/498

[53] Yao Wu, Christopher DuBois, Alice X. Zheng, and Martin Ester. 2016. Collaborative Denoising Auto-Encoders for Top-N Recommender Systems. In *Proceedings of the 9th ACM International Conference on Web Search & Data Mining (WSDM '16)*. ACM, San Francisco, CA, USA, 153–162. https://doi.org/10.1145/2835776.2835837

[54] Yao Wu, Xudong Liu, Min Xie, Martin Ester, and Qing Yang. 2016. CCCF: Improving Collaborative Filtering via Scalable User-Item Co-Clustering. In *Proceedings of the 9th ACM International Conference on Web Search and Data Mining (WSDM '16)*. ACM, San Francisco, CA, USA, 73–82. https://doi.org/10.1145/2835776.2835836

[55] Xingyu Xie, Xianglin Guo, Guangcan Liu, and Jun Wang. 2018. Implicit Block Diagonal Low-Rank Representation. *IEEE Trans. Image Processing* 27, 1 (2018), 477–489. https://doi.org/10.1109/TIP.2017.2764262

[56] Bin Xu, Jiajun Bu, Chun Chen, and Deng Cai. 2012. An exploration of improving collaborative recommender systems via user-item subgroups. In *Proceedings of the 21st World Wide Web Conference (WWW '12)*. Lyon, France, 21–30. https://doi.org/10.1145/2187836.2187840

[57] Feng Xue, Xiangnan He, Xiang Wang, Jiandong Xu, Kai Liu, and Richang Hong. 2019. Deep Item-based Collaborative Filtering for Top-N Recommendation. *ACM Trans. Inf. Syst.* 37, 3 (2019), 33:1–33:25. https://doi.org/10.1145/3314578

[58] Gui-Rong Xue, Chenxi Lin, Qiang Yang, Wensi Xi, Hua-Jun Zeng, Yong Yu, and Zheng Chen. 2005. Scalable Collaborative Filtering using Cluster-based Smoothing. In *Proceedings of the 28th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '05)*. ACM, Salvador, Brazil, 114–121. https://doi.org/10.1145/1076034.1076056

[59] Hilmi Yildirim and Mukkai S. Krishnamoorthy. 2008. A Random Walk Method for Alleviating the Sparsity Problem in Collaborative Filtering. In *Proceedings of the 2nd ACM Conference on Recommender Systems (RecSys '08)*. ACM, Lausanne, Switzerland, 131–138. https://doi.org/10.1145/1454008.1454031

[60] Yongfeng Zhang, Min Zhang, Yiqun Liu, Shaoping Ma, and Shi Feng. 2013. Localized matrix factorization for recommendation based on matrix block diagonal forms. In *Proceedings of the 22nd International World Wide Web Conference (WWW '13)*. Rio de Janeiro, Brazil, 1511–1520. https://doi.org/10.1145/2488388.2488520

[61] Feipeng Zhao and Yuhong Guo. 2016. Improving Top-N Recommendation with Heterogeneous Loss. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI '16)*. New York, NY, USA, 2378–2384. http://www.ijcai.org/Abstract/16/339

[62] Cai-Nicolas Ziegler, Sean M. McNee, Joseph A. Konstan, and Georg Lausen. 2005. Improving Recommendation Lists through Topic Diversification. In *Proceedings of the 14th international conference on World Wide Web (WWW '05)*. Chiba, Japan, 22–32. https://doi.org/10.1145/1060745.1060754