This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2021.3079581, IEEE Transactions on Knowledge and Data Engineering

Multi-modal Discrete Collaborative Filtering for Efficient Cold-start Recommendation

Yang Xu, Lei Zhu, Zhiyong Cheng, Jingjing Li, Zheng Zhang, Huaxiang Zhang

Abstract—Hashing is an effective technique to improve the efficiency of large-scale recommender system by representing both users and items into binary codes. However, existing hashing-based recommendation methods still suffer from two important problems: 1) *Cold-start*. They employ the user-item interactions and single auxiliary information to learn the binary hash codes. But the full interaction history is not always available and the single auxiliary information may be missing. 2) *Efficient optimization*. They learn the hash codes with two-step relaxed optimization or one-step discrete hash optimization based on the discrete cyclic coordinate descent, which results in significant quantization loss or still consumes considerable computation time. In this paper, we propose a *Multi-modal Discrete Collaborative Filtering* (MDCF) for efficient cold-start recommendation. We map the multi-modal features of users and items to a consensus Hamming space based on the matrix factorization framework. Specifically, a low-rank self-weighted multi-modal fusion module is designed to adaptively fuse the multi-modal features into binary hash codes. Additionally, to support large-scale recommendation, a fast discrete optimization method based on augmented Lagrangian multiplier is developed to directly compute the binary hash codes with simple operations. Experiments show the superior performance of the proposed method over state-of-the-art baselines.

Index Terms—Multi-modal fusion, Discrete collaborative filtering, Cold-start, Efficient recommendation

1 Introduction

With the development of E-commerce, recommender systems have been widely adopted by many online services for helping their customers find desirable products to purchase. However, the ever-growing scales of products and users render recommendation more challenging than ever before [1]. For example, there are more than 0.82 billion active Taobao[1] users and over one billion products for sale till now. Consequently, it is challenging to make immediate response to match products for potential customers accurately and efficiently, by analyzing large-scale yet sparse user interaction history.

As a critical class of recommendation methods, Collaborative Filtering (CF), as exemplified by Matrix Factorization (MF) algorithms have demonstrated great success in both academia and industry. MF factorizes an $n \times m$ user-item rating matrix to project both users and items into a r-dimensional latent feature space, where the user's preference scores for items are predicted by the inner product between their latent features. However, the time complexity for generating top-k items recommendation for all users is $\mathcal{O}(nmr + nm \log k)$ [2]. Therefore, MF-based methods are often computationally expensive and inefficient when handling the large-scale recommendation applications [3, 4].

Hashing-based recommendation algorithms [5, 6] are promising to tackle the efficiency challenge by mapping both users and items into the same k-dimension binary Hamming space. Each user and item are then represented by k-bit binary codes. The Hamming similarity between them can be computed very efficiently by Hamming distance (using an Exclusive-Or operation). However, learning binary hash codes is generally NP-hard [7] due to the discrete constraints. To tackle this problem, the researchers resort to a two-step hash learning procedure [6, 8], where continuous representations are first computed by the relaxed optimization, and subsequently the hash codes are generated by binary quantization. This learning strategy indeed simplifies the optimization challenge. However, it inevitably suffers from significant quantization loss [5, 9]. Hence, several solutions are developed to directly optimizing the binary hash codes with Discrete Cyclic Coordinate descent (DCC) that one hash bit is optimized in each iteration step [10–12].

Despite much progress has been achieved, existing hashing-based recommendation methods still suffer from two important problems: 1) *Cold-start*. Most hashing-based recommendation methods mainly rely on the user-item interactions and single specific content feature. Multi-modal features of users and items are not taken into account. Under such circumstances, they cannot provide meaningful recommendations for new users and items (e.g. for the new items who have no interaction history with the users or lack of the particular auxiliary feature), thus cold-start problems cannot be well handled. 2) *Efficient optimization*. Most the state-of-the-art hashing-based recommendation methods learn the hash codes bit-by-bit with DCC. Thus, learning all hashing bits requires lots of iterations. Although DCC scales linearly with the size of data, a lot of additional computation cost and information loss are generated in the

- *Y. Xu, L. Zhu and H. Zhang are with the School of Information Science and Engineering, Shandong Normal University, Jinan 250358, China. Code is available at https://github.com/zzmylq/MDCF.*
- *Z. Cheng is with Shandong Computer Science Center (National Supercomputer Center in Jinan), Qilu University of Technology (Shandong Academy of Sciences).*
- *J. Li is with the School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China.*
- *Z. Zhang is with Bio-Computing Research Center, Harbin Institute of Technology, Shenzhen 518055, China*

[1] https://www.taobao.com/

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2021.3079581, IEEE Transactions on Knowledge and Data Engineering

optimization process.

To alleviate these problem, in this paper, we propose a *Multi-modal Discrete Collaborative Filtering* (MDCF) method for fast cold-start recommendation. We extract multi-modal features from the cold-start objects, and simultaneously map them into the compact binary hash codes by sufficiently exploiting their complementarity. More importantly, different from existing cold-start recommendation solutions [13–16], we propose a self-weighted multi-modal binary mapping method to adaptively fuse the multi-modal features into hash codes with automatically generated fusion weights. Besides, in real-world large-scale recommender systems, data sparsity is also a significant challenge. To solve this problem, we additionally impose low-rank constraint on multi-modal fusion module, which handles the extremely sparse user-item interaction data and helps highlight the latent shared features across different users and items. To support large-scale recommender systems, we develop an efficient discrete optimization approach based on augmented Lagrangian multiplier to directly solve binary hash codes by simple and efficient operations with alleviating the quantization errors. Moreover, in the online recommendation stage, the proposed method can efficiently fuse multi-modal features by using dynamic modality weights and adaptively generate the hash codes for cold-start users and items.

Finally, we evaluate the proposed method on three public datasets, and demonstrate its superior performance over the state-of-the-art baselines. The proposed optimization approach yields much lower time and space costs and higher recommendation performance than DCC. Since real-world recommender systems tend to use a two-stage recommendation framework, consisting of an efficient item-recalling stage and a highly accurate fine-ranking stage. To better show the performance of the hash codes, we further design a two-stage recommendation framework. The experimental results of two-stage recommendation framework show that our proposed method can substantially improve the performance of the recommender systems with a small information loss. The main technique contributions of this paper are summarized as follows:

- We propose a Multi-modal Discrete Collaborative Filtering (MDCF) method for efficient cold-start recommendation. MDCF transforms multi-modal features of users and items into binary hash codes by sufficiently exploiting the complementarity. To the best of our knowledge, there is still no similar work.
- We propose an efficient discrete optimization strategy based on augmented Lagrangian multiplier to directly learn the user and item hash codes with simple efficient operations. This strategy avoids the great storage cost of huge interaction matrix and the performance penalty in existing discrete cyclic coordinate descent based hash optimization process.
- Instead of adopting the fixed modality fusion weights to generate hash codes, we propose a modality-adaptive and self-weighted online hashing module to generate hash codes of cold-start users and items. Specifically, the online hashing module is hyperparameter-free. It could avoid time-consuming and inaccurate parameter adjustment in the online recommendation process.

This paper is an extension of our preliminary paper [17]. In this paper, we further deliver the following contributions:

- Different from [17] that fuses multi-modal information of users and only supports cold-start user recommendation, we reformulate the objective function and transform multi-modal features of users and items into binary hash codes separately. Our new formulation enables MDCF to take full advantage of multi-modal auxiliary information of both users and items, and can be applied directly in both cold-start user and cold-start item recommendation tasks.
- We develop an efficient discrete optimization strategy of MDCF, and analyze its efficiency theoretically and experimentally. Additionally, we propose an initialization module to further improve the learning performance of MDCF. The evaluation results show the advantage of initialization module on accelerating the model training and improving the recommendation performance.
- We conduct more extensive experiments on larger public datasets, report more performance metrics of recommendation, and compare the proposed method with more competing baselines. Additionally, we design a two-stage recommender system, consisting of an efficient item-recalling stage and a highly accurate fine-ranking stage, to evaluate the performance of our proposed method in practice. The evaluation results show the advantage of MDCF on accelerating the recommendation of practical recommender systems with acceptable accuracy loss.

2 Related Work

In this paper, we investigate the hashing-based collaborative filtering at the presence of multi-modal features for fast cold-start recommendation. Hence, in this section, we mainly review the recommender systems with auxiliary information and the recent advanced hashing-based recommendation methods.

2.1 Recommendation with Auxiliary Information

Collaborative filtering is one of the most widely used techniques in recommender systems. However, in the cold-start scenario, since the collaborations between users or items are not available, CF-based models become ineffective. To alleviate the cold-start problem, one of the main strategies is to use auxiliary information such as demographic data, trust relations or user reviews beside the collaborative filtering method [14, 18, 19]. In general, a large amount of descriptive information about items and users is available in real-world applications, such as visual descriptions and textual descriptions of movie. Making full use of multi-modal auxiliary information can improve our understanding of items and users [20, 21]. Due to the success of hybrid methods which incorporate the auxiliary information and the collaborative filtering, most current multi-modal recommendation algorithms are based on the hybrid models [22–25]. For instance, [22] proposes a deep users' multimodal preferences-based recommendation method to capture the textual and visual matching of users and items for recommendation. [23] learns the modal-specific representations of users and items by utilizing information interchange between users and items in multiple modalities for micro-video recommendation. [25] adopts deep matrix factorization architecture to learn the concept representation of multi-model data. However, in the cold-start scenario, both interaction and auxiliary information may be missing, which can lead to a significant degradation of recommendation performance. In addition, there are still few works that apply hashing techniques to multi-modal cold-start recommender systems.

2.2 Hashing-based Recommendation

As discussed in Section 1, the two-step hashing-based recommendation framework consists of relaxed optimization step and binary quantization step. The real-valued representations of users and items are first obtained by the relaxed optimization, and hash codes of them are then generated by the quantization. A pioneer work of this kind [26] is proposed to exploit Locality-Sensitive Hashing (LSH) [27] to generate hash codes for Google new readers based on their item-sharing history similarity. Based on this, [28, 29] follow the idea of Iterative Quantization [30] to generate binary codes from real-valued user/item latent factors. To enhance discriminative capability of hash codes, the decorrelated constraint [8] is imposed on user/item real-valued latent factors before quantization. However, due to the metric loss of latent factors induced by quantization, the hash codes only preserve the similarity between user and item, rather than preference based on inner product. Therefore, [6] propose to impose Constant Feature Norm (CFN) constraint on real-valued latent factors of user and item, and then quantize the metric values and similarity separately. [31] proposes a collaborative hashing model and corresponding distributed optimization method to learn user and item hash codes. As indicated by [5], this two-step approach will lead to significant quantization loss.

To alleviate quantization loss, direct binary code learning by discrete optimization is proposed [32]. In the recommendation area, Discrete Collaborative Filtering (DCF) [5] is the first binarized collaborative filtering method, which directly learns binary hash codes in matrix factorization with binary constraint by DCC. On the basis of DCF, Discrete Deep Learning (DDL) [13] applies Deep Belief Network (DBN) to extract real-valued representation of items from auxiliary information, and generates hash codes by combining the DBN with DCF. Content-aware discrete matrix factorization methods [14, 15] develop discrete optimization algorithms to learn binary codes for users and items at the presence of their respective auxiliary information. Discrete Factorization Machines (DFM) [16] learns hash codes for any auxiliary feature and models the pair-wise interactions between feature codes. Discrete trust-aware matrix factorization (DTMF) [33] and discrete social recommendation (DSR) [34] learn binary representation of users and items by reconstructing the rating and social relationship between users and items. The recommendation process of the above algorithms mainly relies on the user-item interactions and single auxiliary feature. When the part of interaction history is not available or the single auxiliary feature is missing, their performance will be seriously deteriorated. Besides, since the above approaches solve the hash codes with bit-by-bit discrete optimization, they still consume considerable computation time.

TABLE 1
Main notations used in this paper.

Notation	Description
$\boldsymbol{B}$	binary hash code matrix of n users
$\boldsymbol{D}$	binary hash code matrix of m items
$\boldsymbol{S}$	the user-item rating matrix
$\boldsymbol{X}^{(k)}/\boldsymbol{Y}^{(k)}$	feature matrix of the k-th modality data of users/items
$\phi(\boldsymbol{X}^{(k)})/\phi(\boldsymbol{Y}^{(k)})$	nonlinear transformed representation of $\boldsymbol{X}^{(k)}/\boldsymbol{Y}^{(k)}$
$\boldsymbol{H}_x/\boldsymbol{H}_y$	multi-modal shared factor representation of users/items
$\boldsymbol{W}_x^{(k)}/\boldsymbol{W}_y^{(k)}$	mapping matrix of the k-th modality data of users/items
$\boldsymbol{R}_x, \boldsymbol{R}_y$	rotation matrix
$\boldsymbol{Z}_{R_x}, \boldsymbol{Z}_{R_y}$	auxiliary discrete variable
$\mu_x^{(k)}/\mu_y^{(k)}$	weight of the k-th modality data of users/items
n	the number of users
m	the number of items
p	the number of anchors
r	hash code length

Different from existing hashing-based recommendation algorithms, the proposed MDCF method has the following advantages. First, the proposed multi-modal binary mapping strategy is low-rank, self-weighted, and efficient. It can support cold-start recommendation well. Second, we propose an efficient discrete optimization method to directly learn the binary hash codes, which has better hash learning efficiency than the widely-used DCC-based discrete hash optimization method. Third, the online cold-start recommendation problem is based on efficient online hashing, which can efficiently fuse multi-modal data of cold-start users and items and adaptively generate hash codes by using dynamic modality weights. Finally, we design a two-stage recommender system based on MDCF to better illustrate the advantages of the binary representation for the practical recommender systems.

3 The Proposed Method

3.1 Notations and Problem Formulation

Assume that there are n users and m items, and the user-item rating matrix $\boldsymbol{S}$ is of size $n \times m$, where each entry $s_{ij} \in \mathbb{R}$ indicates rating of a user i for an item j. Suppose that $O_x = o_{xi}|_{i=1}^{n}$ is the user training dataset, which contains multi-modal auxiliary information of n users represented with M_x different modality features (e.g. demographic information, location and interaction preference extracted from tags and reviews), and $O_y = o_{yi}|_{i=1}^{m}$ is the item training dataset, which contains multi-modal auxiliary information of m items represented with M_y different modality features (e.g. audiovisual materials of items, descriptions, tags and reviews). The k-th modality feature of users and items are denoted as $\boldsymbol{X}^{(k)} = [x_1^{(k)}, x_2^{(k)}, ..., x_n^{(k)}] \in \mathbb{R}^{d_k \times n}$ and $\boldsymbol{Y}^{(k)} = [y_1^{(k)}, y_2^{(k)}, ..., y_m^{(k)}] \in \mathbb{R}^{d_k \times m}$ respectively, where d_k is the feature dimension of the k-th modality. Our proposed method aims at learning hash codes $\boldsymbol{B} \in \{-1, 1\}^{r \times n}$ for users and $\boldsymbol{D} \in \{-1, 1\}^{r \times m}$ for items to represent their latent factors in the offline training stage, where r is the hash code length. The hash codes of the cold-start users and items that have no collaborative information or few collaborative information are obtained by efficient online hashing, and recommendation results are quickly generated by calculating the Hamming distance between hash codes

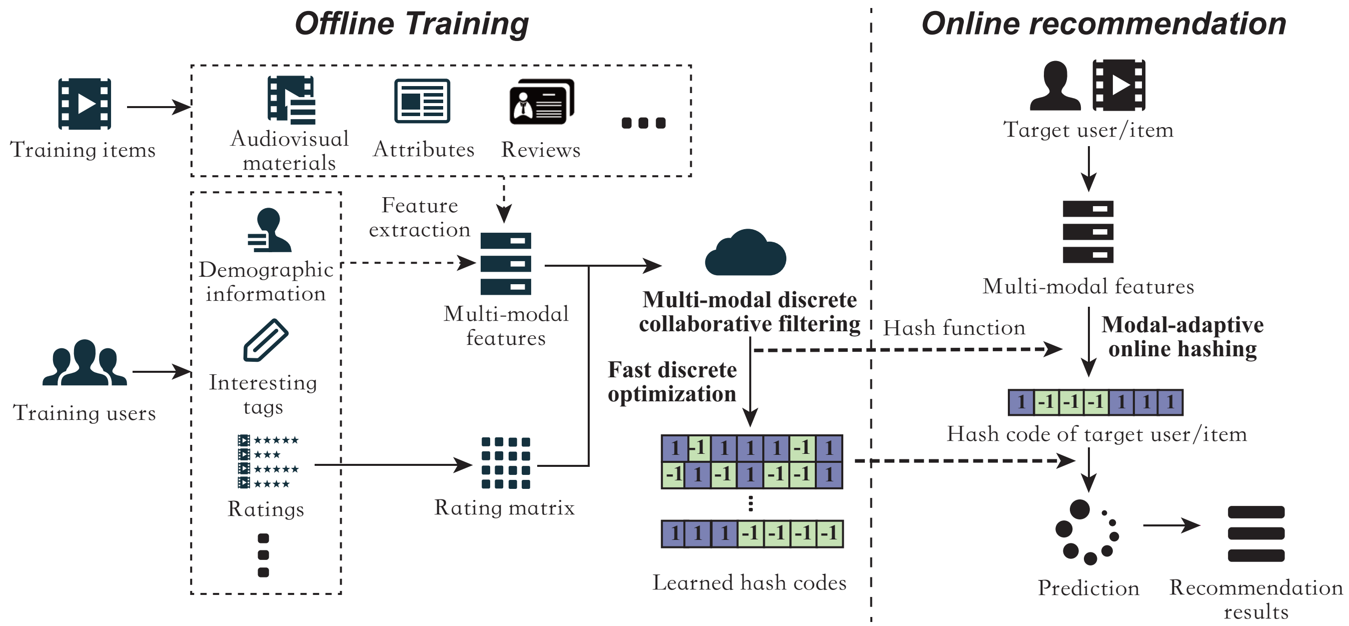


Fig. 1. The basic framework of the proposed MDCF. The framework consists of two main parts: offline training and online recommendation. The main task of the offline training stage is to learn hash functions and generate hash codes for users and items by fusing multi-modal auxiliary information of them. In the online recommendation stage, when cold-start objects arrive, the binary hash codes can be quickly generated by the proposed modality-adaptive hashing method with dynamic modality weights and the learned hash functions in the offline training stage.

in the online recommendation stage. The basic framework of the proposed MDCF is illustrated in Figure 1.

Throughout this paper, we use bold lowercase letters to represent vectors and bold uppercase letters to represent matrices. All of the vectors in this paper denote column vectors. Non-bold letters represent scalars. We denote $tr(\cdot)$ as the trace of a matrix and $\|\cdot\|_F$ as the Frobenius norm of a matrix. $\Delta_m \overset{def}{=} \{x \in \mathbb{R}^m | x_i \geq 0, 1^\top x = 1\}$ is the probabilistic simplex. $sgn(\cdot) : R \rightarrow \pm 1$ is the sign function which returns -1 for $x < 0$ and 1 for $x \geq 0$. Main notations used in this paper are listed in Table 1.

3.2 Low-rank Self-weighted Multi-modal Fusion

Given a training dataset $O = o_i|_{i=1}^{l}$, which contains l multi-modal auxiliary information represented with M different modality features. The k-th modality feature is $\mathbf{X}^{(k)} = [\mathbf{x}_1^{(k)}, ..., \mathbf{x}_l^{(k)}] \in \mathbb{R}^{d_k \times l}$. We first obtain the nonlinear transformed representation $\phi(\mathbf{x}_i^{(k)})$ as $[exp(\frac{||\mathbf{x}_i^{(k)} - \mathbf{a}_1^{(k)}||_F^2}{2\sigma_k^2}), ..., exp(\frac{||\mathbf{x}_i^{(k)} - \mathbf{a}_p^{(k)}||_F^2}{2\sigma_k^2})]$ where $\{\mathbf{a}_1^{(k)}, ..., \mathbf{a}_p^{(k)}\}$ are p anchors that are randomly selected from the training samples in the m-th modality, σ_k is the Gaussian kernel parameter. $\phi(\mathbf{X}^{(k)}) = [\phi(\mathbf{x}_1^{(k)}), ..., \phi(\mathbf{x}_l^{(k)})] \in \mathbb{R}^{p \times l}$ preserves the modality-specific sample correlations by characterizing correlations between the sample and certain anchors. Since the heterogeneous modality gap and inter-modality redundancy in multi-modal data are detrimental to hashing learning. In this paper, we aim at adaptively mapping the nonlinear transformed representation $\phi(\mathbf{X}^{(k)})|_{k=1}^{M}$ into a consensus shared multi-modal representation $\mathbf{H} \in \mathbb{R}^{r \times l}$ (r is the hash code length) in a shared homogeneous space. Specifically, considering that the complementarity of multi-modal features and the generalization ability of the fusion module are very important, we formulate this part as:

$$\min_{\mu^{(k)},\mathbf{W}^{(k)},\mathbf{H}} \sum_{k=1}^{M} \mu^{(k)} ||\boldsymbol{H} - \boldsymbol{W}^{(k)}\phi(\boldsymbol{X}^{(k)})||_F^2 + \zeta||\mu||_F^2, \quad (1)$$
$$s.t. \mu = [\mu^{(1)}, \mu^{(2)}, ..., \mu^{(M)}]^\top, \mu \in \Delta_M,$$

where $\boldsymbol{W}^{(k)} \in \mathbb{R}^{r \times p}, k = 1, ..., M$ is the mapping matrix of the k-th modality feature, $\mu^{(k)}$ is the weight of the k-th modality and it measures the importance of modality feature. By setting weights, the complementarity of multi-modal features can be fully exploited. Similar to previous multi-modal fusion method [11], we introduce the second term in Eq.(1) to smooth the weight distribution. ζ is a hyper-parameter used to balance the fusion weights of each modality in the multi-modal fusion process. If there is no ζ or if ζ tends to 0, the weight of the optimal modality with the minimum reconstruction loss will be assigned to 1, while the weights of the other modalities will be assigned to 0. On the other hand, If ζ approaches infinity, each modality will be assigned an equal weight. Under such circumstance, the fusion approach will not be able to exploit the complementarity of multi-modal features. Therefore, it is advisable to involve an additional parameter ζ in this parameter-weighted hash learning, whose optimal value is confirmed to be data related. In practice, introducing additional parameter means that more time will be consumed on parameter adjustment in the offline training process, and the parameter adjustment requirement is also contradictory to the fact that we cannot manually assign an appropriate parameter value for each cold-start user and item in the online recommendation process.

To address this problem, in this paper, we introduce a virtual weight and propose a self-weighted multi-modal mapping approach which can achieve the same goal as Eq.(1) without additional hyper-parameter. The formula is

$$\min_{\mathbf{W}^{(k)},\mathbf{H}} \sum_{k=1}^{M} ||\boldsymbol{H} - \boldsymbol{W}^{(k)}\phi(\boldsymbol{X}^{(k)})||_F, \quad (2)$$

where $||\cdot||_F$ is the Frobenius norm of the matrix. We can derive the following theorem.

Theorem 1 *Eq.(2) is equivalent to*

$$\min_{\boldsymbol{\mu}\in\Delta_M,\boldsymbol{W}^{(k)},\boldsymbol{H}}\sum_{k=1}^{M}\frac{1}{\mu^{(k)}}||\boldsymbol{H}-\boldsymbol{W}^{(k)}\phi(\boldsymbol{X}^{(k)})||_F^2. \tag{3}$$

Proof *Note that,*

$$\begin{aligned}&\sum_{k=1}^{M}\frac{1}{\mu^{(k)}}||\boldsymbol{H}-\boldsymbol{W}^{(k)}\phi(\boldsymbol{X}^{(k)})||_F^2\\ &\overset{(a)}{=}(\sum_{k=1}^{M}\frac{1}{\mu^{(k)}}||\boldsymbol{H}-\boldsymbol{W}^{(k)}\phi(\boldsymbol{X}^{(k)})||_F^2)(\sum_{k=1}^{M}\mu^{(k)})\\ &\overset{(b)}{\geq}(\sum_{k=1}^{M}||\boldsymbol{H}-\boldsymbol{W}^{(k)}\phi(\boldsymbol{X}^{(k)})||_F)^2,\end{aligned} \tag{4}$$

where (a) holds since $\sum_{k=1}^{M}=\mu^{(k)}=1$ and (b) holds according to the Cauchy-Schwarz inequality. This equation indicates

$$(\sum_{k=1}^{M}||\boldsymbol{H}-\boldsymbol{W}^{(k)}\phi(\boldsymbol{X}^{(k)})||_F)^2=\min_{\boldsymbol{\mu}\in\Delta_M}\sum_{k=1}^{M}\frac{1}{\mu^{(k)}}||\boldsymbol{H}-\boldsymbol{W}^{(k)}\phi(\boldsymbol{X}^{(k)})||_F^2, \tag{5}$$

It is easy to derive

$$\begin{aligned}&\min_{\boldsymbol{W}^{(k)},\boldsymbol{H}}\sum_{k=1}^{M}||\boldsymbol{H}-\boldsymbol{W}^{(k)}\phi(\boldsymbol{X}^{(k)})||_F\\ &\Leftrightarrow\min_{\boldsymbol{W}^{(k)},\boldsymbol{H}}(\sum_{k=1}^{M}||\boldsymbol{H}-\boldsymbol{W}^{(k)}\phi(\boldsymbol{X}^{(k)})||_F)^2\\ &\Leftrightarrow\min_{\boldsymbol{\mu}\in\Delta_M,\boldsymbol{W}^{(k)},\boldsymbol{H}}\sum_{k=1}^{M}\frac{1}{\mu^{(k)}}||\boldsymbol{H}-\boldsymbol{W}^{(k)}\phi(\boldsymbol{X}^{(k)})||_F^2,\end{aligned} \tag{6}$$

which completes the proof.

As shown in Eq.(3), if the k-th modality feature is discriminative, then the value of $||\boldsymbol{H}-\boldsymbol{W}^{(k)}\phi(\boldsymbol{X}^{(k)})||_F$ should be small and the corresponding $\frac{1}{\mu^{(k)}}$ should be large. Accordingly, if the modality feature is indiscriminative, it should have a small $\frac{1}{\mu^{(k)}}$. Therefore, $\frac{1}{\mu^{(k)}}$ can be considered as a virtual weight of the k-th modality feature, and it measures the importance of this modality.

In this paper, we focus on enabling the recommendation task for cold-start users and items. Given the training nonlinear transformed representations $\phi(\boldsymbol{X}^{(k)})|_{k=1}^{M_x}$ and $\phi(\boldsymbol{Y}^{(k)})|_{k=1}^{M_y}$. We aim at mapping them into corresponding consensus shared multi-modal representations H_x and H_y by the form of Eq.(2), respectively. We formulate this part as

$$\begin{aligned}\min_{\Theta_H}\boldsymbol{\mathcal{L}}_H=\min_{\Theta_H}&\sum_{k=1}^{M_x}\frac{1}{\mu_x^{(k)}}||\boldsymbol{H}_x-\boldsymbol{W}_x^{(k)}\phi(\boldsymbol{X}^{(k)})||_F^2\\ &+\sum_{k=1}^{M_y}\frac{1}{\mu_y^{(k)}}||\boldsymbol{H}_y-\boldsymbol{W}_y^{(k)}\phi(\boldsymbol{Y}^{(k)})||_F^2,\end{aligned} \tag{7}$$

where Θ_H denotes the variables to be learned.

In practical recommender systems, such as Taobao[2] and Amazon[3], there are a huge number of users and items, which have rich and diverse auxiliary information. However, a specific user only has a small number of interactions with limited items, and the auxiliary information is also complex and varied. Consequently, we need to map large mount of heterogeneous and high-dimensional sparse feature into a homogeneous shared space. To avoid spurious correlations caused by the mapping matrix, we impose a low-rank constraint on $\boldsymbol{W}_x$ and $\boldsymbol{W}_y$:

$$\min_{\Theta_H}\boldsymbol{\mathcal{L}}_H+\gamma(\sum_{k=1}^{M_x}rank(\boldsymbol{W}_x^{(k)})+\sum_{k=1}^{M_y}rank(\boldsymbol{W}_y^{(k)})), \tag{8}$$

where γ is a balance parameter and $rank(\cdot)$ is the rank operator of a matrix. The low-rank constraint on mapping matrix helps highlight the latent shared features across different users or items and handles the extremely spare observations. Meanwhile, the low-rank constraint makes the optimization more difficult for the reason that low-rank optimization is a well-known NP-hard problem. As an alternative method, the nuclear norm is well-known to be a convex surrogate to the matrix rank, and is widely used to encourage low-rankness in previous work [35, 36]. However, the nuclear norm optimizes the singular values of the matrix, but the changes of the singular values are not always lead to a change of the rank. To tackle this problem, we adopt an explicit form of low-rank constraint as follows:

$$\min\sum_{k=1}^{M}rank(\boldsymbol{W}^{(k)})\Leftrightarrow\min\sum_{k=1}^{M}\sum_{i=d+1}^{l_k}(\sigma_i(\boldsymbol{W}^{(k)}))^2, \tag{9}$$

where $\sigma_i(\boldsymbol{W^{(k)}})$ denotes the i-th singular value of $\boldsymbol{W^{(k)}}$. l_k is the total number of singular values of $\boldsymbol{W^{(k)}}$. Note that

$$\sum_{i=d+1}^{l_k}(\sigma_i(\boldsymbol{W}^{(k)}))^2=tr({\boldsymbol{V}^{(k)}}^\top\boldsymbol{W}^{(k)}{\boldsymbol{W}^{(k)}}^\top\boldsymbol{V}^{(k)}), \tag{10}$$

where $\boldsymbol{V^{(k)}}$ consists of singular vectors which correspond to the (l_k-d)-smallest singular values of $\boldsymbol{W}^{(k)}{\boldsymbol{W}^{(k)}}^\top$. Based on Eq.(9) and Eq.(10), we can rewrite the low-rank constraint on $\boldsymbol{W_x}$ and $\boldsymbol{W_y}$ as follows:

$$\begin{aligned}\min_{\boldsymbol{V}_x^{(k)},\boldsymbol{V}_y^{(k)}}\boldsymbol{\mathcal{L}}_R=\min_{\boldsymbol{V}_x^{(k)},\boldsymbol{V}_y^{(k)}}&\sum_{k=1}^{M_x}tr({\boldsymbol{V}_x^{(k)}}^\top\boldsymbol{W}_x^{(k)}{\boldsymbol{W}_x^{(k)}}^\top\boldsymbol{V}_x^{(k)})\\ &+\sum_{k=1}^{M_y}tr({\boldsymbol{V}_y^{(k)}}^\top\boldsymbol{W}_y^{(k)}{\boldsymbol{W}_y^{(k)}}^\top\boldsymbol{V}_y^{(k)}).\end{aligned}$$

The multi-modal fusion module can be rewritten as: (11)

$$\min_{\Theta_H,\boldsymbol{V}_x^{(k)},\boldsymbol{V}_y^{(k)}}\boldsymbol{\mathcal{L}}_H+\gamma\boldsymbol{\mathcal{L}}_R. \tag{12}$$

3.3 Multi-modal Discrete Collaborative Filtering

To obtain feature representations applicable to efficient cold-start recommendation task, in this paper, we propose to preserve multi-modal shared factors into binary hash codes with matrix factorization, which can support large-scale collaborative filtering problems.

Given a rating matrix $\boldsymbol{S}$ of size $n\times m$, where n and m are the number of users and items, respectively. Let $b_i\in\{\pm1\}^r$ denote the binary hash codes for the i-th user, and $d_j\in\{\pm1\}^r$ denote the binary hash codes for the j-th item, the rating of user i for item j is approximated by Hamming similarity $(\frac{1}{2}+\frac{1}{2r}b_i^\top d_j)$. Thus, our goal is to learn binary hash code matrix $\boldsymbol{B}=[\boldsymbol{b_1},...,\boldsymbol{b_n}]\in\{\pm1\}^{r\times n}$ and $\boldsymbol{D}=[\boldsymbol{d_1},...,\boldsymbol{d_m}]\in\{\pm1\}^{r\times m}$for users and items respectively, where $r\ll min(n,m)$ is the hash code length. Similar to the problem of conventional collaborative filtering, the basic discrete collaborative filtering can be formulated as:

$$\begin{aligned}&\min_{\boldsymbol{B},\boldsymbol{D}}||\boldsymbol{S}-\boldsymbol{B}^\top\boldsymbol{D}||_F^2,\\ &s.t.\boldsymbol{B}\in\{\pm1\}^{r\times n},\boldsymbol{D}\in\{\pm1\}^{r\times m}.\end{aligned} \tag{13}$$

To address the sparse and cold-start problem, we integrate auxiliary information of users and items into the above model, by substituting $\boldsymbol{B}$ and $\boldsymbol{D}$ with the rotated multi-modal shared factors $\boldsymbol{R_x H_x}$ and $\boldsymbol{R_y H_y}$ respectively

[2] www.taobao.com

[3] www.amazon.com

($\boldsymbol{R_x}, \boldsymbol{R_y} \in \mathbb{R}^{r\times r}$ are rotation matrices), and keeping their consistency during the optimization process. The formula is given as follows:

$$\begin{aligned}\min_{\boldsymbol{R}_x,\boldsymbol{R}_y,\boldsymbol{B},\boldsymbol{D}} &||\boldsymbol{S}-\boldsymbol{H}_x^\top\boldsymbol{R}_x^\top\boldsymbol{R}_y\boldsymbol{H}_y||_F^2\\ &+\alpha_1||\boldsymbol{B}-\boldsymbol{R}_x\boldsymbol{H}_x||_F^2+\alpha_2||\boldsymbol{D}-\boldsymbol{R}_y\boldsymbol{H}_y||_F^2,\\ s.t.\ &\boldsymbol{B}\in\{\pm1\}^{r\times n},\boldsymbol{D}\in\{\pm1\}^{r\times m},\boldsymbol{R}_x^\top\boldsymbol{R}_x=\boldsymbol{R}_y^\top\boldsymbol{R}_y=\boldsymbol{I}_r.\end{aligned} \tag{14}$$

This formulation has two advantages: 1) All of the decomposed variable are not directly subject to discrete constraint. As shown in the optimization part, the hash codes can be learned with a simple $sgn(\cdot)$ operation instead of bit-by-bit discrete optimization. The second and third terms can guarantee the acceptable information loss. 2) The learned hash codes can reflect the multi-modal features of users and items via $\boldsymbol{H}_x$ and $\boldsymbol{H}_y$ respectively, and involve the latent interactive features in $\boldsymbol{S}$ simultaneously.

3.4 Overall Objective Formulation

By integrating the above two parts into a unified learning framework, we derive the overall objective formulation of Multi-modal Discrete Collaborative Filtering (MDCF) as:

$$\begin{aligned}\min_{\Theta} &||\boldsymbol{S}-\boldsymbol{H}_x^\top\boldsymbol{R}_x^\top\boldsymbol{R}_y\boldsymbol{H}_y||_F^2+\alpha_1||\boldsymbol{B}-\boldsymbol{R}_x\boldsymbol{H}_x||_F^2\\ &+\alpha_2||\boldsymbol{D}-\boldsymbol{R}_y\boldsymbol{H}_y||_F^2+\beta\boldsymbol{\mathcal{L}}_H+\gamma\boldsymbol{\mathcal{L}}_R,\\ s.t.\ &\boldsymbol{B}\in\{\pm1\}^{r\times n},\boldsymbol{D}\in\{\pm1\}^{r\times m},\boldsymbol{R}_x^\top\boldsymbol{R}_x=\boldsymbol{R}_y^\top\boldsymbol{R}_y=\boldsymbol{I}_r,\end{aligned} \tag{15}$$

where $\alpha_1, \alpha_2, \beta, \gamma$ are balance parameters. The first three terms minimize the information loss during the process of preserving multi-modal features and interaction features into hash codes. $\boldsymbol{\mathcal{L}}_H$ projects multi-modal features of users and items into corresponding homogeneous shared space, respectively. $\boldsymbol{\mathcal{L}}_R$ is a low-rank constraint for mapping matrix, which can highlight the latent shared features across different users or items.

3.5 Fast Discrete Optimization

Solving hash codes in Eq.(15) is essentially an NP-hard problem due to the discrete constraint. Existing hashing-based recommendation methods always learn the hash codes bit-by-bit with DCC [5]. Although this strategy alleviates the quantization loss problem caused by conventional relaxing-rounding optimization strategy, it is still time-consuming.

In this paper, with the favorable support of objective formulation, we propose to directly learn the discrete hash codes with fast optimization. Specifically, different from existing hashing-based recommendation methods [5, 13, 14, 16], we avoid explicitly computing the rating matrix $\boldsymbol{S}$, and achieve linear computation and storage efficiency. We propose an effective optimization algorithm based on augmented Lagrangian multiplier (ALM) [37, 38]. In particular, we introduce two auxiliary variables, $\boldsymbol{Z_{R_x}}$ and $\boldsymbol{Z_{R_y}}$, to separate the constraint on $\boldsymbol{R}_x$ and $\boldsymbol{R}_y$ respectively, and transform the objective function Eq.(15) to an equivalent one that can be tackled more easily. Let

$$\boldsymbol{\mathcal{L}_Z}=||\boldsymbol{R}_x-\boldsymbol{Z}_{\boldsymbol{R}_x}+\frac{\boldsymbol{G}_{\boldsymbol{R}_x}}{\lambda}||_F^2+||\boldsymbol{R}_y-\boldsymbol{Z}_{\boldsymbol{R}_y}+\frac{\boldsymbol{G}_{\boldsymbol{R}_y}}{\lambda}||_F^2, \tag{16}$$

where $\boldsymbol{Z}_{\boldsymbol{R}_x}^\top\boldsymbol{Z}_{\boldsymbol{R}_x}=\boldsymbol{Z}_{\boldsymbol{R}_y}^\top\boldsymbol{Z}_{\boldsymbol{R}_y}=\boldsymbol{I}_r$. $\boldsymbol{G}_{\boldsymbol{R}_x},\boldsymbol{G}_{\boldsymbol{R}_y}\in\mathbb{R}^{r\times r}$ measure the difference between the targets and auxiliary variables. Then the Eq.(15) is transformed as:

$$\begin{aligned}\min_{\Theta} &||\boldsymbol{S}-\boldsymbol{H}_x^\top\boldsymbol{R}_x^\top\boldsymbol{R}_y\boldsymbol{H}_y||_F^2+\alpha_1||\boldsymbol{B}-\boldsymbol{R}_x\boldsymbol{H}_x||_F^2\\ &+\alpha_2||\boldsymbol{D}-\boldsymbol{R}_y\boldsymbol{H}_y||_F^2+\beta\boldsymbol{\mathcal{L}}_H+\gamma\boldsymbol{\mathcal{L}}_R+\frac{\lambda}{2}\boldsymbol{\mathcal{L}_Z},\\ s.t.\ &\boldsymbol{B}\in\{\pm1\}^{r\times n},\boldsymbol{D}\in\{\pm1\}^{r\times m},\boldsymbol{R}_x^\top\boldsymbol{R}_x=\boldsymbol{R}_y^\top\boldsymbol{R}_y=\boldsymbol{I}_r,\end{aligned} \tag{17}$$

where Θ denotes the variables that need to be solved in the objective function, $\lambda > 0$ is a balance parameter. With this transformation, we follow the alternative optimization process by updating each of variables, given the others fixed.

Step 1: learning $\mu_x^{(k)}$ and $\mu_y^{(k)}$. For convenience, we denote $||\boldsymbol{H_x}-\boldsymbol{W_x}^{(k)}\phi(\boldsymbol{X}^{(k)})||_F$ as $h_x^{(k)}$. By fixing the other variables, we ignore the term that is irrelevant to $\mu_x^{(k)}$. The original problem can be rewritten as:

$$\min_{\mu_x^{(k)}\geq0,\boldsymbol{1}^\top\boldsymbol{\mu_x}=1}\sum_{k=1}^{M_x}\frac{{h_x^{(k)}}^2}{\mu_x^{(k)}}. \tag{18}$$

With Cauchy-Schwarz inequality, we derive that

$$\sum_{k=1}^{M_x}\frac{{h_x^{(k)}}^2}{\mu_x^{(k)}}\overset{(a)}{=}(\sum_{k=1}^{M_x}\frac{{h_x^{(k)}}^2}{\mu_x^{(k)}})(\sum_{k=1}^{M_x}\mu_x^{(k)})\overset{(b)}{\geq}(\sum_{k=1}^{M_x}h_x^{(k)})^2,$$

where (a) holds since $\boldsymbol{1}^\top\boldsymbol{\mu_x}=1$ and the equality in (b) holds when $\sqrt{\mu_x^{(k)}}\propto\frac{h_x^{(k)}}{\sqrt{\mu_x^{(k)}}}$. Since $(\sum_{k=1}^{M_x}h_x^{(k)})^2=const$, we can obtain the optimal $\mu_x^{(k)}$ in Eq.(18) by

$$\mu_x^{(k)}=\frac{h_x^{(k)}}{\sum_{k=1}^{M_x}h_x^{(k)}}. \tag{19}$$

Similar to $\mu_x^{(k)}$, the optimal $\mu_y^{(k)}$ can be obtained by

$$\mu_y^{(k)}=\frac{h_y^{(k)}}{\sum_{k=1}^{M_y}h_y^{(k)}}. \tag{20}$$

Step 2: learning $\boldsymbol{W_x^k}$ and $\boldsymbol{W_y^k}$. Since $\boldsymbol{W_x^k}$ and $\boldsymbol{W_y^k}$ are learned in a similar way, for convenience, we first introduce the learning method of mapping matrix $\boldsymbol{W_x^k}$. Removing the terms that are irrelevant to the $\boldsymbol{W_x^k}$, the optimization formula is rewritten as

$$\begin{aligned}\min_{\boldsymbol{W}_x^{(k)}}&\sum_{k=1}^{M_x}\frac{\beta}{\mu_x^{(k)}}||\boldsymbol{H}_x-\boldsymbol{W}_x^{(k)}\phi(\boldsymbol{X}^{(k)})||_F^2\\ &+\gamma\sum_{k=1}^{M_x}tr({\boldsymbol{V}_x^{(k)}}^\top\boldsymbol{W}_x^{(k)}{\boldsymbol{W}_x^{(k)}}^\top\boldsymbol{V}_x^{(k)}).\end{aligned} \tag{21}$$

We calculate the derivative of Eq.(21) with respect to $\boldsymbol{W_x}$ and set it to zero,

$$\begin{aligned}&\sum_{k=1}^{M_x}\frac{\beta}{\mu_x^{(k)}}||\boldsymbol{H}_x-\boldsymbol{W}_x^{(k)}\phi(\boldsymbol{X}^{(k)})||_F^2\\ &\qquad+\gamma\sum_{k=1}^{M_x}tr({\boldsymbol{V}_x^{(k)}}^\top\boldsymbol{W}_x^{(k)}{\boldsymbol{W}_x^{(k)}}^\top\boldsymbol{V}_x^{(k)})=0\\ \Rightarrow&\gamma\boldsymbol{V}_x^{(k)}{\boldsymbol{V}_x^{(k)}}^\top\boldsymbol{W}_x^{(k)}+\frac{\beta}{\mu_x^{(k)}}\boldsymbol{W}_x^{(k)}\phi(\boldsymbol{X}^{(k)})\phi(\boldsymbol{X}^{(k)})^\top\\ &\qquad=\frac{\beta}{\mu_x^{(k)}}\boldsymbol{H}_x\phi(\boldsymbol{X}^{(k)})^\top.\end{aligned} \tag{22}$$

By using the following substitutions,

$$\begin{cases}\boldsymbol{A}_x=\gamma\boldsymbol{V}_x^{(k)}{\boldsymbol{V}_x^{(k)}}^\top\\ \boldsymbol{B}_x=\frac{\beta}{\mu_x^{(k)}}\phi(\boldsymbol{X}^{(k)})\phi(\boldsymbol{X}^{(k)})^\top\\ \boldsymbol{C}_x=\frac{\beta}{\mu_x^{(k)}}\boldsymbol{H}_x\phi(\boldsymbol{X}^{(k)})^\top\end{cases}, \tag{23}$$

Eq.(22) can be rewritten as $\boldsymbol{A}_x\boldsymbol{W}_x+\boldsymbol{W}_x\boldsymbol{B}_x=\boldsymbol{C}_x$, which can be efficiently solved by Sylvester operation in Matlab. Similar to $\boldsymbol{W}_x^{(k)}$, the Sylvester equation with respect to $\boldsymbol{W}_y^{(k)}$ is $\boldsymbol{A}_y\boldsymbol{W}_y+\boldsymbol{W}_y\boldsymbol{B}_y=\boldsymbol{C}_y$, where

$$\begin{cases}\boldsymbol{A}_y=\gamma\boldsymbol{V}_y^{(k)}{\boldsymbol{V}_y^{(k)}}^\top\\ \boldsymbol{B}_y=\frac{\beta}{\mu_y^{(k)}}\phi(\boldsymbol{Y}^{(k)})\phi(\boldsymbol{Y}^{(k)})^\top\\ \boldsymbol{C}_y=\frac{\beta}{\mu_y^{(k)}}\boldsymbol{H}_y\phi(\boldsymbol{Y}^{(k)})^\top\end{cases}. \tag{24}$$

Step 3: learning $\boldsymbol{R}_x$ and $\boldsymbol{R}_y$. The optimization formula for updating $\boldsymbol{R}_x$ can be represented as

$$\min_{\boldsymbol{R}_x^\top \boldsymbol{R}_x = I_r} tr(\boldsymbol{R}_x^\top \boldsymbol{R}_y \boldsymbol{H}_y \boldsymbol{H}_y^\top \boldsymbol{R}_y^\top \boldsymbol{R}_x \boldsymbol{H}_x \boldsymbol{H}_x^\top - 2\alpha_1 \boldsymbol{R}_x^\top \boldsymbol{B} \boldsymbol{H}_x^\top \\ - 2\boldsymbol{R}_x^\top \boldsymbol{R}_y \boldsymbol{H}_y \boldsymbol{S}^\top \boldsymbol{H}_x^\top - \lambda \boldsymbol{R}_x^\top (\boldsymbol{Z}_{\boldsymbol{R}_x} - \frac{\boldsymbol{G}_{\boldsymbol{R}_x}}{\lambda})). \tag{25}$$

It is challenging to solve $\boldsymbol{R}_x$ directly due to the orthogonal constraint. In this paper, for the term of $\boldsymbol{R}_x^\top \boldsymbol{R}_y \boldsymbol{H}_y \boldsymbol{H}_y^\top \boldsymbol{R}_y^\top \boldsymbol{R}_x \boldsymbol{H}_x \boldsymbol{H}_x^\top$ in Eq.(25), we use an auxiliary variable $\boldsymbol{Z}_{\boldsymbol{R}_x} \in \mathbb{R}^{r \times r}$ to substitute the second $\boldsymbol{R}_x$, and simultaneously keep the equivalence of them during the optimization. With the constraint $\boldsymbol{R}_x^\top \boldsymbol{R}_x = \boldsymbol{I}_r$, the above equation can be transformed as:

$$\max_{\boldsymbol{R}_x^\top \boldsymbol{R}_x = \boldsymbol{I}_r} tr(\boldsymbol{R}_x^\top \boldsymbol{C}_x) ,$$

$$\begin{aligned} \boldsymbol{C}_x = & - \boldsymbol{R}_y \boldsymbol{H}_y \boldsymbol{H}_y^\top \boldsymbol{R}_y^\top \boldsymbol{Z}_{\boldsymbol{R}_x} \boldsymbol{H}_x \boldsymbol{H}_x^\top + 2\alpha_1 \boldsymbol{B} \boldsymbol{H}_x^\top \\ & + 2\boldsymbol{R}_y \boldsymbol{H}_y \boldsymbol{S}^\top \boldsymbol{H}_x^\top + \lambda \boldsymbol{Z}_{\boldsymbol{R}_x} - \boldsymbol{G}_{\boldsymbol{R}_x} . \end{aligned} \tag{26}$$

With transformation, the optimal $\boldsymbol{R}_x$ is defined as $\boldsymbol{R}_x = \boldsymbol{P}_x \boldsymbol{Q}_x^\top$, where $\boldsymbol{P}_x$ and $\boldsymbol{Q}_x$ are comprised of left-singular and right-singular vectors of $\boldsymbol{C}_x$, respectively [39]. The objective function with respect to $\boldsymbol{R}_y$ can be represented as

$$\min_{\boldsymbol{R}_y^\top \boldsymbol{R}_y = \boldsymbol{I}_r} tr(\boldsymbol{R}_y^\top \boldsymbol{R}_x \boldsymbol{H}_x \boldsymbol{H}_x^\top \boldsymbol{R}_x^\top \boldsymbol{R}_y \boldsymbol{H}_y \boldsymbol{H}_y^\top - 2\alpha_2 \boldsymbol{R}_y^\top \boldsymbol{D} \boldsymbol{H}_y^\top \\ - 2\boldsymbol{R}_y^\top \boldsymbol{R}_x \boldsymbol{H}_x \boldsymbol{S}^\top \boldsymbol{H}_y^\top - \lambda \boldsymbol{R}_y^\top (\boldsymbol{Z}_{\boldsymbol{R}_y} - \frac{\boldsymbol{G}_{\boldsymbol{R}_y}}{\lambda})). \tag{27}$$

We introduce an auxiliary variable $\boldsymbol{Z}_{\boldsymbol{R}_y}$ and substitute $\boldsymbol{R}_y^\top \boldsymbol{R}_x \boldsymbol{H}_x \boldsymbol{H}_x^\top \boldsymbol{R}_x^\top \boldsymbol{R}_y \boldsymbol{H}_y \boldsymbol{H}_y^\top$ with $\boldsymbol{R}_y^\top \boldsymbol{R}_x \boldsymbol{H}_x \boldsymbol{H}_x^\top \boldsymbol{R}_x^\top \boldsymbol{Z}_{\boldsymbol{R}_y} \boldsymbol{H}_y \boldsymbol{H}_y^\top$. Thus, the Eq.(27) can be transformed into the following form

$$\max_{\boldsymbol{R}_y^\top \boldsymbol{R}_y = \boldsymbol{I}_r} tr(\boldsymbol{R}_y^\top \boldsymbol{C}_y) ,$$

$$\begin{aligned} \boldsymbol{C}_y = & - \boldsymbol{R}_x \boldsymbol{H}_x \boldsymbol{H}_x^\top \boldsymbol{R}_x^\top \boldsymbol{Z}_{\boldsymbol{R}_y} \boldsymbol{H}_y \boldsymbol{H}_y^\top + 2\alpha_2 \boldsymbol{D} \boldsymbol{H}_y^\top \\ & + 2\boldsymbol{R}_x \boldsymbol{H}_x \boldsymbol{S}^\top \boldsymbol{H}_y^\top + \lambda \boldsymbol{Z}_{\boldsymbol{R}_y} - \boldsymbol{G}_{\boldsymbol{R}_y} . \end{aligned} \tag{28}$$

The optimal $\boldsymbol{R}_y$ is defined as $\boldsymbol{R}_y = \boldsymbol{P}_y \boldsymbol{Q}_y^\top$, where $\boldsymbol{P}_y$ and $\boldsymbol{Q}_y$ are comprised of left-singular and right-singular vectors of $\boldsymbol{C}_y$, respectively.

Note that, the user-item rating matrix $\boldsymbol{S} \in \mathbb{R}^{n \times m}$ is included in the terms $2\boldsymbol{R}_y \boldsymbol{H}_y \boldsymbol{S}^\top \boldsymbol{H}_x^\top$ and $2\boldsymbol{R}_x \boldsymbol{H}_x \boldsymbol{S}^\top \boldsymbol{H}_y^\top$ when updating $\boldsymbol{R}_x$ and $\boldsymbol{R}_y$, respectively. In real-world retail giants, such as Taobao and Amazon, there are hundreds of millions of users and even more items. In consequence, the user-item rating matrix $\boldsymbol{S}$ would be pretty enormous and sparse. If we use $\boldsymbol{S}$ directly, the computational complexity will be $\mathcal{O}(mnr)$ and it is extremely expensive to calculate and store $\boldsymbol{S}$. In this paper, we apply the singular value decomposition to obtain the left singular and right singular vectors as well as the corresponding singular values of $\boldsymbol{S}$. We utilize a diagonal matrix Σ_S to store the o-largest ($o \ll min\{m, n\}$) singular values, and employ an $n \times o$ matrix $\boldsymbol{P}_S$, an $o \times m$ matrix $\boldsymbol{Q}_S$ to store the corresponding left singular and right singular vectors respectively. We substitute $\boldsymbol{S}$ with $\boldsymbol{P}_S \Sigma_S \boldsymbol{Q}_S$ and the computational complexity can be reduced to $\mathcal{O}(max\{mor, nor\})(r, o \ll min\{m, n\})$. With transformation, both the computation and storage cost can be decreased with the guarantee of accuracy.

Step 4: learning $\boldsymbol{H}_x$ and $\boldsymbol{H}_y$. We calculate the derivative of objective function with respect to $\boldsymbol{H}_x$ and $\boldsymbol{H}_y$ and set them to zero, then we get

$$\begin{aligned} \boldsymbol{H}_x = & (\sum_{k=1}^{M_x} \frac{\beta}{\mu_x^{(k)}} \boldsymbol{I}_r + \alpha_1 \boldsymbol{I}_r + \boldsymbol{R}_x^\top \boldsymbol{R}_y \boldsymbol{H}_y \boldsymbol{H}_y^\top \boldsymbol{R}_y^\top \boldsymbol{R}_x)^{-1} \\ & (\sum_{k=1}^{M_x} \frac{\beta}{\mu_x^{(k)}} \boldsymbol{W}_x^{(k)} \phi(\boldsymbol{X}^{(k)}) + \alpha_1 \boldsymbol{R}_x^\top \boldsymbol{B} + \boldsymbol{R}_x^\top \boldsymbol{R}_y \boldsymbol{H}_y \boldsymbol{S}^\top), \end{aligned} \tag{29}$$

$$\begin{aligned} \boldsymbol{H}_y = & (\sum_{k=1}^{M_y} \frac{\beta}{\mu_y^{(k)}} \boldsymbol{I}_r + \alpha_2 \boldsymbol{I}_r + \boldsymbol{R}_y^\top \boldsymbol{R}_x \boldsymbol{H}_x \boldsymbol{H}_x^\top \boldsymbol{R}_x^\top \boldsymbol{R}_y)^{-1} \\ & (\sum_{k=1}^{M_y} \frac{\beta}{\mu_y^{(k)}} \boldsymbol{W}_y^{(k)} \phi(\boldsymbol{Y}^{(k)}) + \alpha_2 \boldsymbol{R}_y^\top \boldsymbol{D} + \boldsymbol{R}_y^\top \boldsymbol{R}_x \boldsymbol{H}_x \boldsymbol{S}), \end{aligned} \tag{30}$$

where $\boldsymbol{S}$ is substituted with $\boldsymbol{P}_S \Sigma_S \boldsymbol{Q}_S$, and the time complexity of computing $\boldsymbol{R}_x^\top \boldsymbol{R}_y \boldsymbol{H}_y \boldsymbol{S}^\top$ and $\boldsymbol{R}_y^\top \boldsymbol{R}_x \boldsymbol{H}_x \boldsymbol{S}$ are reduced to $\mathcal{O}(max\{mor, nor\})$.

Step 5: learning $\boldsymbol{B}$ and $\boldsymbol{D}$. We can obtain the closed solutions of $\boldsymbol{B}, \boldsymbol{D}$ as

$$\boldsymbol{B} = sgn(\boldsymbol{R}_x \boldsymbol{H}_x), \quad \boldsymbol{D} = sgn(\boldsymbol{R}_y \boldsymbol{H}_y). \tag{31}$$

Step 6: learning $\boldsymbol{V}_x^{(k)}$ and $\boldsymbol{V}_y^{(k)}$. As described in section 3.2, $\boldsymbol{V}_x^{(k)}$ and $\boldsymbol{V}_y^{(k)}$ are stacked by the singular vectors which correspond to the $(l_k - d)$-smallest singular values of $\boldsymbol{W}_x^{(k)} \boldsymbol{W}_x^{(k)\top}$ and $\boldsymbol{W}_y^{(k)} \boldsymbol{W}_y^{(k)\top}$, respectively. Thus we can solve the eigen-decomposition problem to get $\boldsymbol{V}_x^{(k)}, \boldsymbol{V}_y^{(k)}$:

$$\boldsymbol{V}_x^{(k)} \leftarrow svd(\boldsymbol{W}_x^{(k)} \boldsymbol{W}_x^{(k)\top}), \boldsymbol{V}_y^{(k)} \leftarrow svd(\boldsymbol{W}_y^{(k)} \boldsymbol{W}_y^{(k)\top}). \tag{32}$$

Step 7: learning $\boldsymbol{Z}_{\boldsymbol{R}_x}$ and $\boldsymbol{Z}_{\boldsymbol{R}_y}$. The objective function with respect to $\boldsymbol{Z}_{\boldsymbol{R}_x}$ and $\boldsymbol{Z}_{\boldsymbol{R}_y}$ can be transformed as

$$\max_{\boldsymbol{Z}_{\boldsymbol{R}_x}^\top \boldsymbol{Z}_{\boldsymbol{R}_x} = \boldsymbol{I}_r} tr(\boldsymbol{Z}_{\boldsymbol{R}_x}^\top \boldsymbol{C}_{zrx}),$$

$$\boldsymbol{C}_{zrx} = \lambda \boldsymbol{R}_x - \boldsymbol{R}_y \boldsymbol{H}_y \boldsymbol{H}_y^\top \boldsymbol{R}_y^\top \boldsymbol{R}_x \boldsymbol{H}_x \boldsymbol{H}_x^\top , \tag{33}$$

$$\max_{\boldsymbol{Z}_{\boldsymbol{R}_y}^\top \boldsymbol{Z}_{\boldsymbol{R}_y} = \boldsymbol{I}_r} tr(\boldsymbol{Z}_{\boldsymbol{R}_y}^\top \boldsymbol{C}_{zry}),$$

$$\boldsymbol{C}_{zry} = \lambda \boldsymbol{R}_y - \boldsymbol{R}_x \boldsymbol{H}_x \boldsymbol{H}_x^\top \boldsymbol{R}_x^\top \boldsymbol{R}_y \boldsymbol{H}_y \boldsymbol{H}_y^\top , \tag{34}$$

where the optimal $\boldsymbol{Z}_{\boldsymbol{R}_x}$ and $\boldsymbol{Z}_{\boldsymbol{R}_y}$ can be solved with theorem 2.

Step 8: learning $\boldsymbol{G}_{\boldsymbol{R}_x}$ and $\boldsymbol{G}_{\boldsymbol{R}_y}$. By fixing other variables, the update rules of $\boldsymbol{G}_{\boldsymbol{R}_x}$ and $\boldsymbol{G}_{\boldsymbol{R}_y}$ are

$$\boldsymbol{G}_{\boldsymbol{R}_x} = \boldsymbol{G}_{\boldsymbol{R}_x} + \lambda (\boldsymbol{R}_x - \boldsymbol{Z}_{\boldsymbol{R}_x}), \tag{35}$$

$$\boldsymbol{G}_{\boldsymbol{R}_y} = \boldsymbol{G}_{\boldsymbol{R}_y} + \lambda (\boldsymbol{R}_y - \boldsymbol{Z}_{\boldsymbol{R}_y}). \tag{36}$$

3.6 Modality-adaptive Online Hashing for Cold-start Recommendation

In the online recommendation stage, we aim to map multimodal features of the target users and items into binary hash codes with the learned hash projection matrix $\{\boldsymbol{W}_x^{(k)}\}_{k=1}^{M_x}$ and $\{\boldsymbol{W}_y^{(k)}\}_{k=1}^{M_y}$, respectively. When cold-start users and items have no rating history in the training set and are only associated with auxiliary information of certain modality, the fixed modality weights obtained from offline hash learning cannot address the modality-missing problem, and may fail to effectively capture the variations of cold-start objects.

In this paper, with the support of online hash learning, we propose to generate hash codes for cold-start objects with a self-weighting scheme. The objective functions for cold-start users and items are formulated as

$$\min_{\hat{\boldsymbol{B}}, \boldsymbol{\mu_{x'}} \in \Delta_{M_x}} \sum_{k=1}^{M_x} \frac{1}{\mu_{x'}^{(k)}} ||\hat{\boldsymbol{B}} - \boldsymbol{W}_x^{(k)} \phi(\hat{\boldsymbol{X}}^{(k)})||_F^2, \tag{37}$$

$$\min_{\hat{\boldsymbol{D}},\boldsymbol{\mu_{y'}}\in\Delta_{M_y}} \sum_{k=1}^{M_y} \frac{1}{\mu_{y'}^{(k)}} ||\hat{\boldsymbol{D}} - \boldsymbol{W}_y^{(k)}\phi(\hat{\boldsymbol{Y}}^{(k)})||_F^2, \tag{38}$$

where $\boldsymbol{W}_x^{(k)}$ and $\boldsymbol{W}_y^{(k)}$ are the mapping matrixes from Eq.(15). $\hat{\boldsymbol{X}}^{(k)}$ and $\hat{\boldsymbol{Y}}^{(k)}$ are k-th modality features of target users and items, respectively. $\hat{\boldsymbol{B}}$ and $\hat{\boldsymbol{D}}$ are binary feature matrices of target users and items, respectively. $\mu_{x'}^{(k)}$ and $\mu_{y'}^{(k)}$ are the dynamic modality weights to be learned.

We employ alternating optimization to update $\hat{\boldsymbol{B}}$, $\hat{\boldsymbol{D}}$, $\mu_{x'}^{(k)}$ and $\mu_{y'}^{(k)}$. The update rules are

$$\begin{aligned}
&\mu_{x'}^{(k)} = \frac{h_{x'}^{(k)}}{\sum_{k=1}^{M_x} h_{x'}^{(k)}}\ ,\ h_{x'}^{(k)} = ||\hat{\boldsymbol{B}} - \boldsymbol{W}_x^{(k)}\phi(\hat{\boldsymbol{X}}^{(k)})||_F,\\
&\mu_{y'}^{(k)} = \frac{h_{y'}^{(k)}}{\sum_{k=1}^{M_y} h_{y'}^{(k)}}\ ,\ h_{y'}^{(k)} = ||\hat{\boldsymbol{D}} - \boldsymbol{W}_y^{(k)}\phi(\hat{\boldsymbol{Y}}^{(k)})||_F,\\
&\hat{\boldsymbol{B}} = sgn(\sum_{k=1}^{M_x} \frac{1}{\mu_{x'}^{(k)}} \boldsymbol{W}_x^{(k)}\phi(\hat{\boldsymbol{X}}^{(k)})),\\
&\hat{\boldsymbol{D}} = sgn(\sum_{k=1}^{M_y} \frac{1}{\mu_{y'}^{(k)}} \boldsymbol{W}_y^{(k)}\phi(\hat{\boldsymbol{Y}}^{(k)})).
\end{aligned} \tag{39}$$

3.7 Initialization

Since MDCF deals with the discrete matrix factorization problem, initialization is important for fast convergence. In this paper, we propose an effective initialization strategy, which first solves relaxed problem of Eq.(15), and then quantizes the real-valued representation to obtain the initial hash code for users and items. We also use alternate optimization to optimize the relaxed problem of Eq.(15). The objective function for optimization is the same as Eq.(17) except the binary constraints and the updating rules are the same except $\boldsymbol{B}$ and $\boldsymbol{D}$ because the binary constraints are removed. With the favorable support of Eq.(15), we can obtain real-valued feature matrices of users and items, $\boldsymbol{B^*}$ and $\boldsymbol{D^*}$, by removing the sign function in Eq.(31), respectively. Then, the optimization can be done alternatively. $\boldsymbol{B}$ and $\boldsymbol{D}$ are initialized to feasible solutions $sgn(\boldsymbol{B^*})$ and $sgn(\boldsymbol{D^*})$ respectively, and the other variables are initialized as solutions of the relaxed problem of Eq.(15). The effectiveness of the proposed initialization is described in subsection 4.4.2.

3.8 Complexity Analysis

This section provides space and time complexity analysis of MDCF. Directly adopting the $n \times m$ rating matrix $\boldsymbol{S}$ will bring the space complexity $\mathcal{O}(nm)$, which is unacceptable in large-scale recommender systems. In our method, we substitute $\boldsymbol{S}$ with $\boldsymbol{P}_S\Sigma_S\boldsymbol{Q}_S$ where Σ_S is a diagonal matrix consist of the o-largest singular values, $\boldsymbol{P}_S$ and $\boldsymbol{Q}_S$ are the corresponding left singular and right singular vectors respectively. Assuming that $M_x = M_y = M$. Since the space complexity of storing $\boldsymbol{P}_S$ and $\boldsymbol{Q}_S$ is $\mathcal{O}(o(m+n))$ and that of storing modality features of users and items is $\mathcal{O}(Mp(m+n))$, the space complexity of offline hash code learning stage can be reduced to $\mathcal{O}((o+Mp)(m+n))$.

The overall time cost of constructing $\phi(\boldsymbol{X})$ and $\phi(\boldsymbol{Y})$ is $\mathcal{O}(Mp(m+n))$. The time cost total $\mathcal{O}(Mpr(m+n))$ for updating μ_x and μ_y. The overall time cost of computing $\boldsymbol{W}_x$ and $\boldsymbol{W}_y$ is $\mathcal{O}(2Mr^2(d-l_k) + Mp(p+r)(m+n))$. The overall time cost of updating $\boldsymbol{R}_x$ and $\boldsymbol{R}_y$ is $\mathcal{O}((7r^2 + 2ro)(m+n) + 6r^3 + 2ro^2)$. Updating $\boldsymbol{H}_x$ and $\boldsymbol{H}_y$ require $\mathcal{O}((5r^2 + 2ro + Mpr)(m+n) + 2ro^2 + 8r^3)$ in total. Updating $\boldsymbol{B}$ and $\boldsymbol{D}$ require $\mathcal{O}(r^2(m+n))$ in total. The time cost total $\mathcal{O}(4r^2(m+n) + 4r^3)$ for updating $\boldsymbol{Z}_{R_x}$ and $\boldsymbol{Z}_{R_y}$. The overall time cost of updating $\boldsymbol{V}_x$ and $\boldsymbol{V}_y$ is $\mathcal{O}(2r^2(r+p))$. Updating $\boldsymbol{G}_{R_x}$ and $\boldsymbol{G}_{R_y}$ only require $\mathcal{O}(2r)$ in total. Suppose the entire algorithm requires $iter$ iterations for convergence, the overall time complexity of optimization process is $\mathcal{O}(iter \times Mpr(m+n))$, where r is the hash code length and the convergence experiment results in Fig. 6(a) indicate that $iter$ is usually $5 \sim 10$. In summary, training MDCF is efficient since it scales linearly with $m+n$. In the online stage, it takes $\mathcal{O}(Mpr\hat{n})$ to generate binary hash code for $\hat{n}$ cold-start users/items.

Based on the above analysis, both space and time complexity of MDCF is linear with the size of the dataset $(m+n)$, which is scalable for large-scale recommender systems.

3.9 Convergence Analysis

The objective function Eq.(17) is convex to one variable by fixing the others. Thus, optimizing one variable in each step will cause the value of the objective function to decrease or equal. Our iterative update rule will monotonically reduce the objective function value. After several iterations, the optimization process will eventually reach a local minimum. In addition, we will empirically verify the convergence of the proposed MDCF on three datasets in experiments.

4 Experiments

4.1 Evaluation Datasets

We evaluate the proposed method on three widely used public datasets: MovieLens-1M[4], MovieLens-10M[4] and BookCrossing[5]. In these three datasets, each user has only one rating for an item.

- **MovieLens-1M**: This dataset is collected from the MovieLens website by GroupLens Research. It originally includes 1,000,000 ratings from 6,040 users for 3,952 movies. The rating score is from 1 to 5 with 1 granularity. The users in this dataset are associated with demographic information, and the movies are related to 3-5 genre labels.
- **MovieLens-10M**: This dataset contains 10,000,054 ratings and 95580 tags applied to 10,681 movies by 71,567 users of the MovieLens. Unlike MovieLens-1M dataset, demographic information is not included in this dataset and ratings are made on a 5-star scale, with half-star increments.
- **BookCrossing**: This dataset is collected by Cai-Nicolas Ziegler from the Book-Crossing community. It contains 278,858 users providing 1,149,780 ratings about 271,379 books. The rating score is from 1 to 10 with 1 interval. Most users in this dataset are associated with demographic information.

Considering the extreme sparsity of the original BookCrossing dataset, we remove the users with less than 20 ratings and the items rated by less than 20 users. After the filtering, there are 2,151 users, 6,830 items, and 180,595

[4] https://grouplens.org/datasets/movielens/
[5] https://grouplens.org/datasets/book-crossing/

TABLE 2
Statistics of experimental datasets.

Dataset	#User	#Item	#Rating	Sparsity
BookCrossing	2,151	6,830	180,595	98.77%
MovieLens-1M	6,040	3,952	1,000,209	95.81%
MovieLens-10M	71,567	10,681	10,000,054	98.69%

ratings left in the BookCrossing dataset. For the MovieLens-1M and MovieLens-10M datasets, we keep all users and items without any filtering. The statistics of the datasets are summarized in Table 2. To obtain multi-modal information about items, we crawl the item-related information from the web to extend the original datasets. For the BookCrossing dataset, we crawl descriptions and reviews on Amazon.com based on the book IDs provided in the dataset. For the MovieLens-1M and MovieLens-10M datasets, we crawl directors, writers, cast, storyline, plot keywords, genres and reviews on IMDB.com based on the movie links provided in the datasets. Since some of the users lack multi-modal auxiliary information, we use the user-item interaction data and auxiliary information of users to generate the user's multi-modal preference features, which reflect the user's preference for different modality features of items, and apply them as the user's multi-modal features for model learning.

We refer to all reviews written by a user as user document. Also, an item document can be formed by merging all reviews written for the item. Then, LDA is used to extract the auxiliary features from descriptions, storylines and reviews. The one-hot encoding approach is adopted to generate feature representation for directors, writers, cast, plot keywords, and genres.

In our experiments, we randomly select 20% users as cold-start users, while 20% items as cold-start items, and randomly keep their θ ratings and other ratings are removed from the training set and transferred to the testing set. We repeat the experiments with 5 random splits and report the average values as the experimental results.

4.2 Evaluation Baselines and Evaluation Metrics

In this paper, we compare our approach with three continuous value based recommendation methods and three hashing-based recommendation methods.

- **Zero-shot recommendation (ZSR)** [40] considers cold-start recommendation problem as a zero-shot learning problem [41]. It extracts real-valued representation of user preference for each item from user attribute. The parameters λ and β for the relax and low-rank constraints are tuned within $\{10^{-5}, 10^{-3}, 10^{-2}, 10^{-1}, 1, 5, 10, 100\}$.
- **Collaborative topic regression (CTR)** [18] is a hybrid recommendation algorithm that combines topic model, collaborative filtering and probabilistic matrix factorization (PMF) [42]. CTR generates real-valued latent representations of users and items by exploiting user's collection data and content data of items. The coefficient for the balanced regularization λ_u, λ_v and topic parameter β are tuned within $\{10, 100, 1000, 10000\}$.
- **Collaborative deep learning (CDL)** [19] is a probabilistic model that learns a probabilistic SDAE [43] and CF jointly. CDL leverages an effective deep learning framework to learn real-valued latent representation from interaction data and content data. The parameter λ_u, λ_v, λ_n and λ_w are tuned within $\{10^{-4}, 10^{-2}, 1, 10^2, 10^4\}$, and the layer structure of SDAE is set as [8000, 200, 50] according to the results of their experiments.
- **Discrete Collaborative Filtering (DCF)** [5] is the first binarized CF method that can directly optimize the binary codes for users and items.
- **Discrete content-aware matrix factorization (DCMF)** [14] is the state-of-the-art binarized method for CF with auxiliary information. It is the extension of DMF based on the regression-based modeling. The parameters λ_1 and λ_2 for modeling user and item auxiliary features are tuned within $\{1, 10, 50, 100, 500, 1000\}$.
- **Discrete factorization machines (DFM)** [16] is the first binarized factorization machines method that learns the hash codes for any auxiliary feature and models the pair-wise interaction between feature codes. In DFM, the parameter β for the softened de-correlation constraint is tuned within $\{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 1, 10, 100\}$ according to the results of their sensitive analysis.
- **Discrete deep learning (DDL)** [13] is a binary deep recommendation approach. It adopts Deep Belief Network to extract item representation from item auxiliary information, and combines the DBN with DCF to solve the cold-start recommendation problem. The parameters α, β and λ are tuned within $\{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 1, 10, 100\}$. The layer structure of DBN is set as [8000,800,30].
- **Multi-feature discrete collaborative filtering (MFDCF)** [17] is the method we proposed earlier.

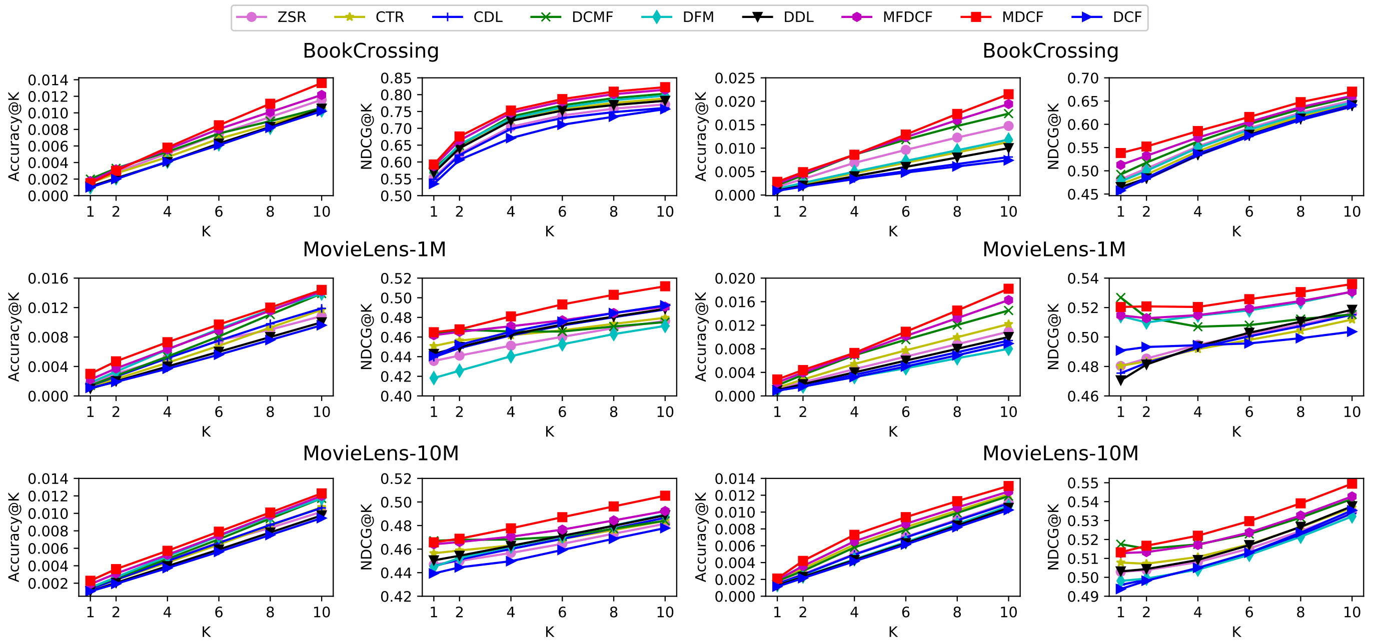
In the experiments, we adopt 5-fold cross validation method on random split of training data to tune the optimal hyperparameters of all compared approaches. All the best hyperparameters are found by grid search.

The goal of our proposed method is to find out the top-k items that the cold-start user may be interested in, or top-k users who are most likely to interact with the cold-start item. In our experiments, we adopt two common ranking evaluation methods: Accuracy@k and Normalized Discounted Cumulative Gain (NDCG), to evaluate the quality of the recommendation list. NDCG is a widely used measure for evaluating recommendation algorithms [44, 45], owing to its comprehensive consideration of both ranking precisions and the position of ratings. Accuracy@k is widely used as a metric for previous ranking based recommender systems [13, 46] to test whether the target user's favorite items that appears in the top-k recommendation list.

4.3 Results and Analysis

4.3.1 Comparison with the State-of-the-art

In this subsection, we evaluate the recommendation performance of MDCF and the baselines in cold-start recommendation scenario when the cold-start threshold θ is set as 1. Fig. 2 demonstrates the recommendation performance, including Accuracy@k and NDCG@k of MDCF and competing baselines on three real-world datasets for the both cold-start item (Fig. 2(a)) and cold-start user (Fig. 2(b)) recommendation tasks when the hash code length r is set as 8.



(a) Cold-start item recommendation

(b) Cold-start user recommendation

Fig. 2. Comparison of MDCF with baseline algorithms on the BookCrossing, MovieLens-1M and MovieLens-10M datasets in both cold-start item recommendation (a) and cold-start user recommendation scenarios (b).

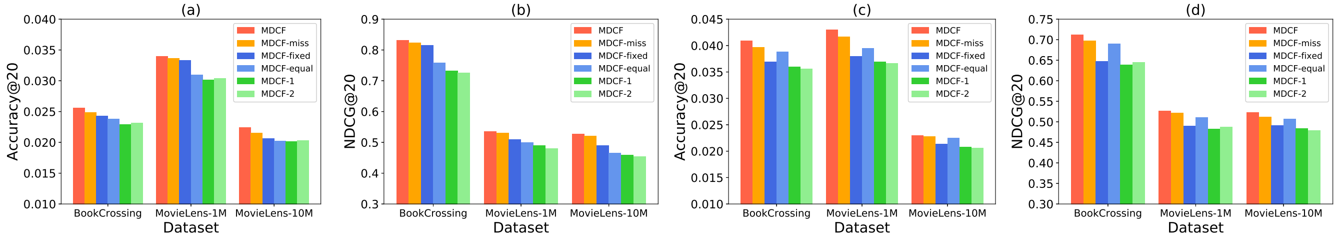


Fig. 3. Effects of our proposed multi-modal self-weight learning strategy in both cold-start item recommendation (a-b) and cold-start user recommendation scenarios (c-d).

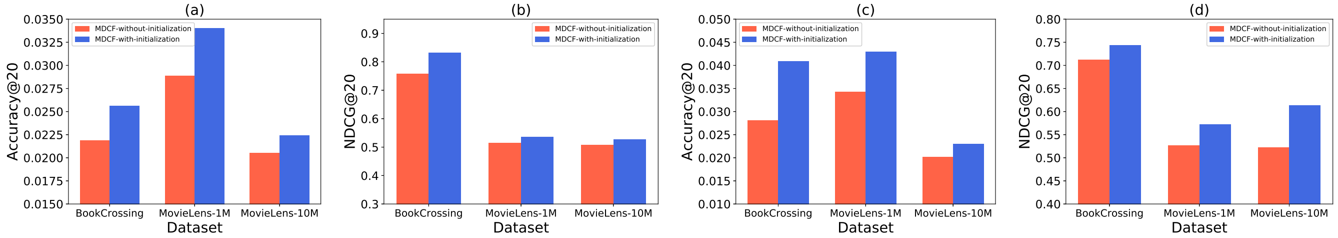


Fig. 4. Accuracy@20 and NDCG@20 using MDCF with/without initializations on three datasets ($r = 8$) in both cold-start item (a-b) and cold-start user (c-d) recommendation scenarios. We can see that the proposed initialization strategy helps to achieve higher performance.

Compared with existing hashing-based recommendation methods, the proposed MDCF algorithm consistently and significantly outperforms the state-of-the-arts with respect to Accuracy@k and NDCG@k in the both cold-start recommendation scenarios. Since DCF does not make use of content information, it does not support cold-start recommendation sufficiently. DFM exploits the factorization machine to model the potential relevance between user characteristics and item features. However, it ignores the collaborative interaction. DDL is based on the discrete collaborative filtering. It adopts DBN to generate item feature representation from their auxiliary information. Nevertheless, the structure of DBN is independent with the overall optimization process, which limits the learning capability of DDL. DCMF integrates auxiliary information about users and items on the basis of matrix factorization. However, it does not take into account the weight of multi-modal auxiliary information and the online learning of cold-start objects, which reduce the cold-start recommendation performance. Moreover, these experimental results show that the proposed MDCF outperforms the compared continuous value based hybrid recommendation methods under the same cold-start settings. The better performance of MDCF than ZSR, CTR and CDL validates the effectiveness of the proposed multi-modal fusion strategy, and demonstrates that the hash codes with discriminative feature representation capability can be learned by our proposed online multi-modal hashing method. Additionally, it can be easily observed that the proposed MDCF consistently outperforms MFDCF on three datasets, showing the effectiveness of the initialization module and MDCF for modeling auxiliary information of users and items.

4.3.2 Run Time Comparison

In these experiments, we compare the computation efficiency of our proposed MDCF with three state-of-the-art hashing-based recommendation methods DMF, DCMF

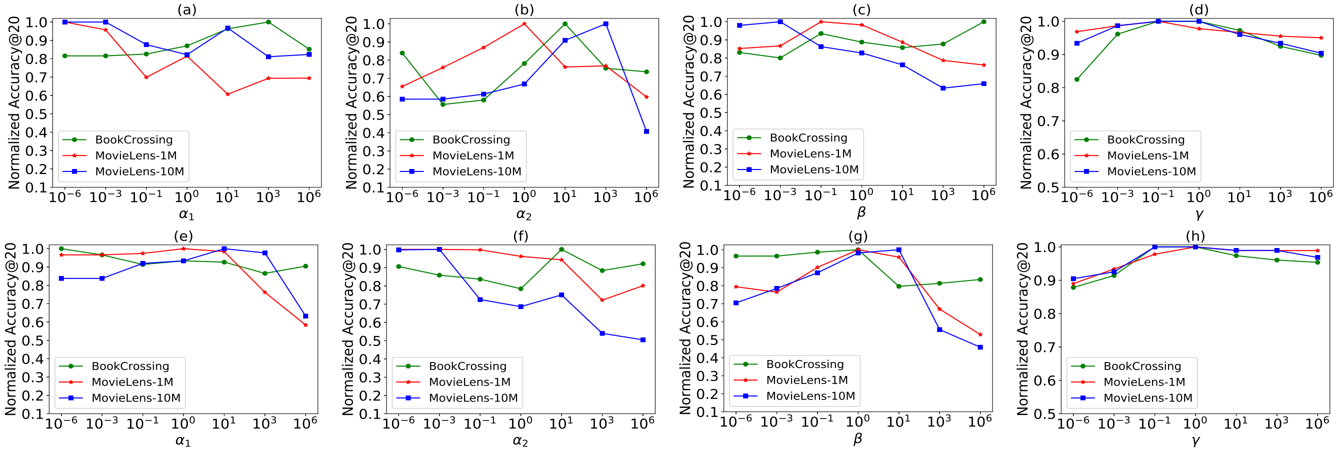


Fig. 5. Sensitive analysis of the MDCF on the three datasets in both cold-start item recommendation (a-d) and cold-start user recommendation scenarios (e-h), where Normalized Accuracy@20 is obtained from dividing each Accuracy@20 by the maximum with respect to the parameter.

TABLE 3
Efficiency comparison between MDCF and the other hashing-based recommendation methods where the code length ranges from 8 to 256 on the BookCrossing dataset.

Methods	Initialization time/iteration (s)						Training time/iteration (s)					
	8 bits	16 bits	32 bits	64 bits	128 bits	256 bits	8 bits	16 bits	32 bits	64 bits	128 bits	256 bits
DDL	2.3398	3.0564	3.4605	3.6305	3.9811	7.7034	56.0339	80.8218	131.7272	349.6231	889.4721	6675.9737
DFM	2.4203	2.9404	4.0259	8.0024	23.0475	102.7623	2.0538	2.3503	2.7921	4.0821	9.1515	26.8376
DCMF	0.1876	0.2234	0.3486	0.8306	2.6959	6.6021	0.1734	0.1957	0.2338	0.4333	0.7743	3.1239
MDCF	**0.0256**	**0.0332**	**0.0512**	**0.0995**	**0.2011**	**0.6981**	**0.0261**	**0.0331**	**0.0504**	**0.1038**	**0.2361**	**0.7436**

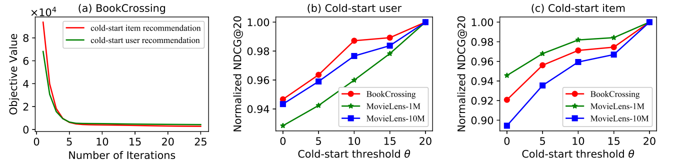


Fig. 6. Variations of objective function value with the number of iterations on BookCrossing dataset (a), Normalized NDCG@20 of MDCF in both cold-start user (b) and cold-start item (c) recommendation scenarios given different cold-start threshold θ.

and DDL on BookCrossing dataset. Similar results can be found on other datasets. The algorithms are implemented via MATLAB. Table 3 demonstrates the efficiency of these methods in both initialization stage and training stage on BookCrossing dataset using a 2.0GHz Intel® Core(TM) i7-4750HQ CPU.

From the Table 3, we can see that MDCF achieves significant speedups. Compared with DFM, our proposed MDCF is about 73 to 139 times faster than DFM in the initialization stage and about 34 to 77 times faster in the training stage. DDL needs to pre-train the DBN once during the initialization stage, and updates the DBN in each round of iteration during the training stage, which slows down the model training considerably. Compared with DDL, MDCF is about 10 to 88 times faster than DDL in the initialization stage and thousands of times faster in the training stage. Specifically, DCMF has a faster training speed compared to other baselines, because it is a recommendation framework based on the matrix factorization, just like our proposed MDCF. However, from the Table 3, we can see that our proposed MDCF is about 6 to 12 times faster than DCMF in the initialization stage and about 3 to 7 times faster in the training stage. This is because DCMF is also based on the DCC optimization strategy, similar to DDL and DFM. Since DCC optimizes hash codes by the bit-wise learning, the update rule is applied among bits iteratively until convergence. Denoting the number of the bit-wise iteration as T_b and the number of entire algorithm iteration as T. The overall time complexity for DCC-based algorithms is $\mathcal{O}(Tr^2(T_b|\mathcal{V}|+m+n))$[5] where $\mathcal{V}$ is observed rating set. As discussed in section 3.8, the overall time complexity of training MDCF is $\mathcal{O}(TMpr(m+n))$. Since $M, p, r \ll min(m,n)$ in practice, the time complexity of DCC-based algorithms is $\mathcal{O}(TT_br^2|\mathcal{V}|)$ more than that of MDCF. It is shown that our proposed fast optimization strategy is more efficient than discrete cyclic coordinate descent theoretically and experimentally.

4.3.3 Ablation Analysis

In this paper, we propose a self-weighted multi-modal fusion strategy to preserve the multi-modal auxiliary information into hash codes while exploring their complementarity, and the hash codes of cold-start objects are adaptively learned in an online mode. The self-weighted scheme addresses the modality-missing problem for cold-start objects and learns adaptive modality weights to dynamically fuse the multi-modal features of cold-start objects. The effectiveness of low-rank constraint is detailed in the subsection 4.3.4. Thus, in this subsection, we design five variants of our method to evaluate the effectiveness of the proposed self-weighted scheme: 1) MDCF-fixed: It adopts fixed modality fusion weights obtained from the offline learning to generate the hash codes of cold-start objects. 2) MDCF-equal: It fixes the weight of each modality to 1 at both the offline learning and online hashing phases. 3) MDCF-1 and MDCF-2: They extract content features from only the first and the second modality of auxiliary information, respectively. 4) MDCF-miss: It randomly removes 50% auxiliary information of users and items in the test set, then tests the performance of MDCF in the modality-missing case. Fig. 3 shows the comparison of the cold-start recommendation performance. From the figures, we can observe that the performance of our method is obviously higher

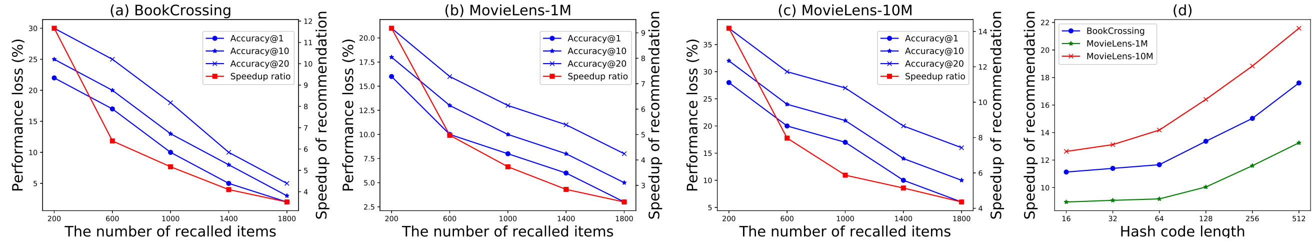


Fig. 7. Results of evaluating the two-stage model on the three datasets (a-c) and speedup ratio with respect to hash code length (d).

than that of the variants in both cold-start recommendation scenarios. In addition, the use of more auxiliary information can be helpful on improving recommendation performance, and the performance of MDCF is minimally affected by the modality-missing problem. These results validate that the proposed self-weighted strategy can indeed explore the complementarity and consistency of multi-modal auxiliary information, adaptively fuse multiple modalities and improve the recommendation accuracy.

Additionally, we design experiments to test the performance of the proposed initialization module. In the section 4.3.2, we discussed the efficiency of the initialization module. Here, we focus on evaluating its effectiveness. The results are shown in Fig. 4. It is easily observed that the MDCF with initialization consistently outperforms MDCF without initialization on the three datasets in both cold-start recommendation scenarios, showing the effectiveness of initialization module in MDCF.

4.3.4 Parameter Sensitivity Analysis and Convergency

We conduct experiments to observe the performance variations with the involved parameters $\alpha_1, \alpha_2, \beta$ and γ. We fix the hash code length as 8 bits and report results on three datasets in both cold-start recommendation scenarios. Since $\alpha_1, \alpha_2, \beta$, and γ are equipped in the same objective function, we change their values from the range of $\{10^{-6}, 10^{-3}, 10^{-1}, 10^{0}, 10^{1}, 10^{3}, 10^{6}\}$ while fixing other parameters. Detailed experimental results are presented in Fig. 5. α_1 is the balance parameter used to control the multi-modal fusion of the user's auxiliary information. From the Fig. 5, we can observe that, in the cold-start item recommendation scenario, $\alpha_1 \in [10^{-6}, 10^{-3}]$ can lead to better recommendation performance on both MovieLens datasets and α_1 should be set to a larger value on BookCrossing dataset. Specifically, a similar pattern is also found in the cold-start user recommendation scenario for α_2, which is the balance parameter used to control the multi-modal fusion of items. This figure also shows that the performance is relatively better when β is in the range of $[10^{-3}, 10^{0}]$ for the cold-start item recommendation, while in the range of $[10^{-1}, 10^{0}]$ for the cold-start user recommendation. In addition, MDCF obtains better performance when the value of γ is from 10^{-1} to 10^{1} in both cold-start scenarios. The performance variations with γ shows that the low-rank constraint can take effect on preserving more discriminative binary codes.

MDCF is suitable for the cold-start recommendation where we want to compute predictions for users or items that have no collaborative information ($\theta = 0$) or few collaborative information. The results of cold-start threshold analysis are shown in Fig. 6(b-c). With the increase of cold-start threshold θ from 0 (no collaborative information) to 20, the recommendation performance gradually improves in all of the three datasets and more rapidly in the sparser datasets, indicating MDCF can work well in different cold-start scenarios.

To evaluate the convergency of the proposed method, we further perform experimental analysis on three datasets with the hash code length fixed as 8 bits. The convergence curves recording the variations of objective function with the number of iterations are shown on the BookCrossing dataset in Fig. 6(a). As shown in the figure, the updating of variables monotonically decreases the objective function value at each iteration. When the number of iterations is less than 5, the objective function value drops sharply. Specifically, when the initialization stage is over and entering the training stage, the binarization of $\boldsymbol{B}$ and $\boldsymbol{D}$ matrices leads to a negligible rise in the objective function value, after which it will rapidly converge. From the Fig. 6(a), we can observe that MDCF achieves a stable minimum within 10 iterations on BookCrossing dataset. Similar convergence results were found on the MovieLens-1M and MovieLens-10M datasets. These results indicate that our proposed method can converge effectively.

4.3.5 Evaluating the Two-Stage Recommender Systems

Similar to [15], we design a two-stage recommender system, consisting of a hashing-based recalling stage and a fine-ranking stage, to help us better understand how MDCF can accelerate practical recommender systems. It is easy to note that our proposed MDCF can be changed to a continuous value based recommendation algorithm with the design idea of initialization module. Thus, in this evaluation, the first stage is to exploit MDCF for recalling the top-K potentially candidates, and the second stage is to use real-valued variant of MDCF for fine-ranking. Moreover, we use the real-valued variant of MDCF as a baseline to evaluate the performance of the proposed two-stage recommender system.

The evaluation results on the three datasets in the cold-start user recommendation scenario are shown in Fig. 7. We can see that when the number of recalled items is small, the two-stage recommender system will have a large performance loss, but the recommendation can be accelerated by more than 20 times, and even up to around 38 in the MovieLens-10M dataset with the largest number of items. Specifically, when the number of recalled items increases, the performance loss will decrease rapidly, but the speedup ratio is also decreasing. Similar results can be found in the cold-start item recommendation scenario. Therefore, in a practical recommender system, we usually need to find a balance between efficiency and effectiveness. Additionally,

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2021.3079581, IEEE Transactions on Knowledge and Data Engineering

from Fig. 7(a-c), we observe that the speedup ratio is usually larger in the datasets with more items, because the recommendation method based on continuous values takes more time to retrieve large item sets, and the efficiency advantage of hash codes will be more obvious. From the description of Section 3.5, we know that the hash code length is an important factor that influences the efficiency of MDCF. Therefore, we conduct experiments to observe the efficiency variations with respect to the hash code length for the two-stage recommender system on three datasets in cold-start user recommendation scenario. The evaluation results are shown in Fig. 7(d). We can easily observe that as the length of the hash code increases, the speedup ratio gets larger and larger. This is because increasing the length of the continuous value features makes the similarity computation time-consuming, while the efficiency of calculating Hamming distance is less affected by the hash code length.

5 Conclusion

In this paper, we propose a multi-modal discrete collaborative filtering method that projects multi-modal auxiliary information of users and items into the binary hash codes to support efficient cold-start recommendation. The proposed method can handle the data sparsity problem with low-rank constraint, enhance the discriminative capability of hash codes with self-weighted multi-modal binary fusing, generate hash codes for the cold-start objects online by modality-adaptive hashing, and achieve computation and storage efficient with discrete binary optimization. The evaluation on three real-words datasets show that the proposed method outperforms the state-of-the-art significantly. Moreover, MDCF is used as the first stage of the two-stage recommender system for recalling the top-K potentially candidates, and demonstrate its advantages for finding a balance between efficiency and effectiveness of practical recommender systems. Moreover, benefiting from the proposed directly discrete optimization method, the efficiency of MDCF is substantially improved compared to the baseline methods.

References

[1] Z. Batmaz, A. Yurekli, A. Bilge, and C. Kaleli, "A review on deep learning for recommender systems: challenges and remedies," *Artif. Intell. Rev.*, vol. 52, no. 1, pp. 1–37, 2019.

[2] Y. Zhang, D. Lian, and G. Yang, "Discrete personalized ranking for fast collaborative filtering from implicit feedback," in *AAAI*, 2017, pp. 1669–1675.

[3] Z. Cheng, Y. Ding, X. He, L. Zhu, X. Song, and M. S. Kankanhalli, "Aˆ3ncf: An adaptive aspect attention model for rating prediction," in *IJCAI*, 2018, pp. 3748–3754.

[4] Z. Cheng, X. Chang, L. Zhu, R. C. Kanjirathinkal, and M. S. Kankanhalli, "MMALFM: explainable recommendation by leveraging reviews and images," *ACM Trans. Inf. Syst.*, vol. 37, no. 2, pp. 1–28, 2019.

[5] H. Zhang, F. Shen, W. Liu, X. He, H. Luan, and T. Chua, "Discrete collaborative filtering," in *SIGIR*, 2016, pp. 325–334.

[6] Z. Zhang, Q. Wang, L. Ruan, and L. Si, "Preference preserving hashing for efficient recommendation," in *SIGIR*, 2014, pp. 183–192.

[7] J. Håstad, "Some optimal inapproximability results," *J. ACM*, vol. 48, no. 4, pp. 798–859, 2001.

[8] X. Liu, J. He, C. Deng, and B. Lang, "Collaborative hashing," in *CVPR*, 2014, pp. 2147–2154.

[9] H. Cui, L. Zhu, J. Li, Y. Yang, and L. Nie, "Scalable deep hashing for large-scale social image retrieval," *IEEE Transactions on Image Processing*, vol. 29, pp. 1271–1284, 2020.

[10] X. Lu, L. Zhu, Z. Cheng, J. Li, X. Nie, and H. Zhang, "Flexible online multi-modal hashing for large-scale multimedia retrieval," in *MM*, 2019, pp. 1129–1137.

[11] X. Lu, L. Zhu, Z. Cheng, L. Nie, and H. Zhang, "Online multi-modal hashing with dynamic query-adaption," in *SIGIR*, 2019, pp. 715–724.

[12] C. Zheng, L. Zhu, X. Lu, J. Li, Z. Cheng, and H. Zhang, "Fast discrete collaborative multi-modal hashing for large-scale multimedia retrieval," *IEEE Trans. Knowl. Data Eng.*, vol. 32, no. 11, pp. 2171–2184, 2020.

[13] Y. Zhang, H. Yin, Z. Huang, X. Du, G. Yang, and D. Lian, "Discrete deep learning for fast content-aware recommendation," in *WSDM*, 2018, pp. 717–726.

[14] D. Lian, R. Liu, Y. Ge, K. Zheng, X. Xie, and L. Cao, "Discrete content-aware matrix factorization," in *KDD*, 2017, pp. 325–334.

[15] D. Lian, X. Xie, and E. Chen, "Discrete matrix factorization and extension for fast item recommendation," *IEEE Trans. Knowl. Data Eng.*, vol. DOI: 10.1109/TKDE.2019.2951386, 2019.

[16] H. Liu, X. He, F. Feng, L. Nie, R. Liu, and H. Zhang, "Discrete factorization machines for fast feature-based recommendation," in *IJCAI*, 2018, pp. 3449–3455.

[17] Y. Xu, L. Zhu, Z. Cheng, J. Li, and J. Sun, "Multi-feature discrete collaborative filtering for fast cold-start recommendation," in *AAAI*, 2020, pp. 270–278.

[18] C. Wang and D. M. Blei, "Collaborative topic modeling for recommending scientific articles," in *KDD*, 2011, pp. 448–456.

[19] H. Wang, N. Wang, and D. Yeung, "Collaborative deep learning for recommender systems," in *KDD*, 2015, pp. 1235–1244.

[20] W. Zhao, Z. Guan, L. Chen, X. He, D. Cai, B. Wang, and Q. Wang, "Weakly-supervised deep embedding for product review sentiment analysis," *IEEE Trans. Knowl. Data Eng.*, vol. 30, no. 1, pp. 185–197, 2018.

[21] W. Zhao, S. Tan, Z. Guan, B. Zhang, M. Gong, Z. Cao, and Q. Wang, "Learning to map social network users by unified manifold alignment on hypergraph," *IEEE Trans. Neural Networks Learn. Syst.*, vol. 29, no. 12, pp. 5834–5846, 2018.

[22] C. Xu, Z. Guan, W. Zhao, Q. Wu, M. Yan, L. Chen, and Q. Miao, "Recommendation by users' multimodal preferences for smart city applications," *IEEE Trans. Ind. Informatics*, vol. 17, no. 6, pp. 4197–4205, 2021.

[23] Y. Wei, X. Wang, L. Nie, X. He, R. Hong, and T. Chua, "MMGCN: multi-modal graph convolution network for personalized recommendation of micro-video," in *MM*, 2019, pp. 1437–1445.

[24] J. Li, K. Lu, Z. Huang, and H. T. Shen, "On both cold-start and long-tail recommendation with social data," *IEEE Trans. Knowl. Data Eng.*, vol. 33, no. 1, pp. 194–208, 2021.

[25] W. Zhao, C. Xu, Z. Guan, and Y. Liu, "Multiview concept learning via deep matrix factorization," *IEEE Trans. Neural Networks Learn. Syst.*, vol. 32, no. 2, pp. 814–825, 2021.

[26] A. Das, M. Datar, A. Garg, and S. Rajaram, "Google news personalization: scalable online collaborative filtering," in *WWW*, 2007, pp. 271–280.

[27] A. Gionis, P. Indyk, and R. Motwani, "Similarity search in high dimensions via hashing," in *VLDB*, 1999, pp. 518–529.

[28] A. Karatzoglou, A. J. Smola, and M. Weimer, "Collaborative filtering on a budget," in *AISTATS*, 2010, pp. 389–396.

[29] K. Zhou and H. Zha, "Learning binary codes for collaborative filtering," in *KDD*, 2012, pp. 498–506.

[30] Y. Gong, S. Lazebnik, A. Gordo, and F. Perronnin, "Iterative quantization: A procrustean approach to learning binary codes for large-scale image retrieval," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 12, pp. 2916–2929, 2013.

[31] C. Chen, Z. Liu, P. Zhao, L. Li, J. Zhou, and X. Li, "Distributed collaborative hashing and its applications in ant financial," in *SIGKDD*, 2018, pp. 100–109.

[32] F. Shen, C. Shen, W. Liu, and H. T. Shen, "Supervised discrete hashing," in *CVPR*, 2015, pp. 37–45.

[33] G. Guo, E. Yang, L. Shen, X. Yang, and X. He, "Discrete trust-aware matrix factorization for fast recommendation," in *IJCAI*, 2019, pp. 1380–1386.

[34] C. Liu, X. Wang, T. Lu, W. Zhu, J. Sun, and S. C. H. Hoi, "Discrete social recommendation," in *AAAI*, 2019, pp. 208–215.

[35] J. Li, Y. Wu, J. Zhao, and K. Lu, "Low-rank discriminant embedding for multiview learning," *IEEE Trans. Cybern.*, vol. 47, no. 11, pp. 3516–3529, 2017.

[36] Z. Ding and Y. Fu, "Deep transfer low-rank coding for cross-domain learning," *IEEE Trans. Neural Networks Learn. Syst.*, vol. 30, no. 6, pp. 1768–1779, 2019.

[37] Z. Lin, M. Chen, and Y. Ma, "The augmented lagrange multiplier method for exact recovery of corrupted low-rank matrices," *CoRR*, vol. abs/1009.5055, 2010.

[38] K. G. Murty, "Nonlinear programming theory and algorithms," *Technometrics*, vol. 49, no. 1, p. 105, 2007.

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

[40] J. Li, M. Jing, K. Lu, L. Zhu, Y. Yang, and Z. Huang, "From zero-shot learning to cold-start recommendation," in *AAAI*, 2019, pp. 4189–4196.

[41] J. Li, M. Jing, K. Lu, Z. Ding, L. Zhu, and Z. Huang, "Leveraging the invariant side of generative zero-shot learning," in *CVPR*, 2019, pp. 7402–7411.

[42] R. Salakhutdinov and A. Mnih, "Probabilistic matrix factorization," in *NIPS*, 2007, pp. 1257–1264.

[43] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P. Manzagol, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," *J. Mach. Learn. Res.*, vol. 11, pp. 3371–3408, 2010.

[44] Z. Cheng and J. Shen, "On effective location-aware music recommendation," *ACM Trans. Inf. Syst.*, vol. 34, no. 2, pp. 13:1–13:32, 2016.

[45] C. Li, X. Niu, X. Luo, Z. Chen, and C. Quan, "A review-driven neural model for sequential recommendation," in *IJCAI*, 2019, pp. 2866–2872.

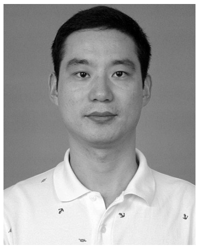
[46] W. Wang, H. Yin, Z. Huang, Q. Wang, X. Du, and Q. V. H. Nguyen, "Streaming ranking based recommender systems," in *SIGIR*, 2018, pp. 525–534.

Yang Xu received the Ph.D. degree at Shandong University, China, in 2018. He is currently a lecture with the School of Information Science and Engineering, Shandong Normal University, China. His research interests are in the area of recommender systems and multimedia computing.

Lei Zhu received the B.S. degree (2009) at Wuhan University of Technology, the Ph.D. degree (2015) at Huazhong University of Science and Technology. He is currently a full Professor with the School of Information Science and Engineering, Shandong Normal University, China. He was a Research Fellow at the University of Queensland (2016-2017), and at the Singapore Management University (2015-2016). His research interests are in the area of large-scale multimedia content analysis and retrieval.

Zhiyong Cheng is currently a Professor with Shandong Artificial Intelligence Institute, Qilu University of Technology (Shandong Academy of Sciences). He received the Ph.D degree in computer science from Singapore Management University in 2016, and then worked as a Research Fellow in National University of Singapore. His research interests mainly focus on large-scale multimedia content analysis and retrieval. His work has been published in a set of top forums, including ACM SIGIR, MM, WWW, TOIS, IJCAI, TKDE, and TCYB. He has served as the PC member for several top conferences such as MM, MMM etc., and the regular reviewer for journals including TKDE, TIP, TMM etc.

Jingjing Li received his MSc and PhD degree in Computer Science from University of Electronic Science and Technology of China in 2013 and 2017, respectively. Now he is a national Postdoctoral Program for Innovative Talents research fellow with the School of Computer Science and Engineering, University of Electronic Science and Technology of China. He has great interest in machine learning, especially transfer learning, subspace learning and recommender systems.



Zheng Zhang received his M.S. degree in Computer Science (2014) and Ph.D. degree in Computer Applied Technology (2018) from the Harbin Institute of Technology, China. Dr. Zhang was a Postdoctoral Research Fellow at The University of Queensland, Australia. He is currently an Assistant Professor at Harbin Institute of Technology, Shenzhen, China. He has published over 50 technical papers at prestigious international journals and conferences, including the IEEE TPAMI, IEEE TNNLS, IEEE TIP, IEEE TCYB, IEEE CVPR, ECCV, AAAI, IJCAI, SIGIR, ACMM, etc.

Huaxiang Zhang received the Ph.D. degree from Shanghai Jiao Tong University, Shanghai, China, in 2004. He was an Associated Professor with the Department of Computer Science, Shandong Normal University, Jinan, China, from 2004 to 2005, where he is currently a Professor with the School of Information Science and Engineering. His current research interests include machine learning, pattern recognition, evolutionary computation, and Web information processing.