Discrete Social Recommendation

Chenghao Liu,[1,2*] **Xin Wang,**[3*] **Tao Lu,**[2,4] **Wenwu Zhu,**[3] **Jianling Sun,**[2,4] **Steven C.H. Hoi**[1]
[1]School of Information Systems, Singapore Management University, Singapore
[2]School of Computer Science and Technology, Zhejiang University, China
[3]Department of Computer Science and Technology, Tsinghua University, China
[4]Alibaba-Zhejiang University Joint Institute of Frontier Technologies, China
{twinsken, 3140102441, sunjl}@zju.edu.cn, {xin_wang,wwzhu}@tsinghua.edu.cn, chhoi@smu.edu.sg,

Abstract

Social recommendation, which aims at improving the performance of traditional recommender systems by considering social information, has attracted broad range of interests. As one of the most widely used methods, matrix factorization typically uses continuous vectors to represent user/item latent features. However, the large volume of user/item latent features results in expensive storage and computation cost, particularly on terminal user devices where the computation resource to operate model is very limited. Thus when taking extra social information into account, precisely extracting K most relevant items for a given user from massive candidates tends to consume even more time and memory, which imposes formidable challenges for efficient and accurate recommendations. A promising way is to simply binarize the latent features (obtained in the training phase) and then compute the relevance score through Hamming distance. However, such a two-stage hashing based learning procedure is not capable of preserving the original data geometry in the real-value space and may result in a severe quantization loss. To address these issues, this work proposes a novel discrete social recommendation (DSR) method which learns binary codes in a unified framework for users and items, considering social information. We further put the balanced and uncorrelated constraints on the objective to ensure the learned binary codes can be informative yet compact, and finally develop an efficient optimization algorithm to estimate the model parameters. Extensive experiments on three real-world datasets demonstrate that DSR runs nearly 5 times faster and consumes only with $1/37$ of its real-value competitor's memory usage at the cost of almost no loss in accuracy.

Introduction

Online social networks have become an important source to provide a huge amount of timely and valuable information for understanding social behaviors and inferring personal preferences. Recommender systems, though are designed to help users discover interesting products or services, always suffer from the lack of useful information for understanding user preferences. Therefore, when recommender systems (which urge for data) meet online social networks (which offer useful data), their combination naturally gives birth to *social recommendation* (Ma et al. 2011; Ma et al. 2008; Tang, Hu, and Liu 2013; Wang et al. 2016) that aims at improving the accuracy of traditional recommendation methods through mitigating the data sparsity issue.

However, the ever-growing scales of various web services bring social recommendation a serious concern on the computational efficiency in terms of time and space. More concretely, matrix factorization based collaborative filtering (CF), one of the most widely adopted techniques in real-world applications, factorizes an $n \times m$ user-item rating matrix and maps both users and items into a r-dimensional ($r \ll \min(n, m)$) real-value latent space. A user's preference for an item can be approximated by the inner product of their real-valued latent vectors. As a result, the number of parameters to be stored grows linearly with the number of users or items. This becomes very challenging when the recommender systems have millions or even billions of users/items, which prevents the model from being deployed to mobile devices. Moreover, to further generate the top-K preferred items for each user, it's necessary to first compute users' preferences for all items, and then sort the preference scores by a descending order, whose computation complexity, i.e., $\mathcal{O}(mnr + mn \log K)$, becomes a critical bottleneck in terms of efficiency especially when m or n is quite large. Both issues become even more severe on social recommendation tasks which need extra storage and computation to aggregate social information from social ties.

One simple solution is to distribute the calculations with parallel computation techniques (Zhou et al. 2008), which is not our focus in this work. Another promising option is to encode real-valued latent vectors into compact binary codes as they cost less memory than real-valued vectors and it is much more efficient to compute the inner product by bitwise operations such as Hamming distance. Furthermore, by exploiting special data structures for indexing all items, the computational complexity of generating top-K preferred items is sub-linear or even constant (Wang, Kumar, and Chang 2010; Muja and Lowe 2009). This triggers us to propose a novel framework of Discrete Social Recommendation (DSR) which learns binary codes by taking social information into consideration. However, learning the binary codes is generally NP-hard (Håstad 2001) due to its discrete constraints. Given this NP-hardness, a two-stage optimization procedure, which first solves a relaxed optimization problem

*equal contribution

through ignoring the discrete constraints and then binarizes the results by thresholding, becomes a compromising solution. Nevertheless, as pointed out by Zhang et al. (Zhang et al. 2016), this solution suffers from a large quantization loss for failing to preserve the original data geometry (user-item relevance and user-user relationship) in the continuous real-valued vector space. To this end, the proposed DSR adopts a unified framework which optimizes the user/item binary codes in a way that can preserve the intrinsic user-item relevance and user-user relationship. Moreover, DSR is able to obtain informative yet compact binary codes in a limited size through satisfying additional balanced and uncorrelated constraints. To solve the discrete optimization with balanced and uncorrelated bits in a tractable way, we develop an efficient alternative optimization algorithm which solves several mixed-integer programming subproblems in an iterative process. Extensive experiments are carried out on real-world datasets to validate the recommendation accuracy and retrieval time of our proposed DSR method.

As a summary, we make the following contributions:

- We propose a novel discrete social recommendation (DSR) for compact representations. DSR is able to learn informative yet compact user/item binary codes for efficient user-item relevance and user-user relationship calculation and significantly reduce the storage cost in the scenario of large-scale social recommendation.
- We develop an efficient learning algorithm to solve the discrete optimization problem in a tractable way, with balanced and uncorrelated constraints being considered simultaneously.
- We conduct extensive experiments on three real-world datasets to demonstrate that DSR runs roughly 5 times faster and consumes down to $1/37$ of its real-value competitor's memory usage at the cost of almost no loss in accuracy.

To the best of our knowledge, DSR is the first work that learns binary representations in social recommendation. The proposed method is potentially applicable to a wide range of other social recommendation tasks.

Related Work

In this section, we briefly review related work on social recommendation and binary code collaborative filtering.

Social Recommendation. The data sparsity and cold start problem are two important reasons deteriorating the performance of traditional recommender systems (Liu et al. 2017). With the booming of data in social media, a large amount of valuable social information can be utilized for effectively solving these problems through modeling the mutual influences between users (Wang et al. 2017b), which motivates the advent of social recommendation that aims to improve conventional recommender systems through utilizing social information. Specifically, Ma et al. (2008) combines collaborative filtering with social information by proposing a probabilistic matrix factorization model which factorizes user-item rating matrix and user-user linkage matrix simultaneously. They later present another probabilistic matrix factorization model which aggregates a user's own rating and her friends' ratings to predict the target user's final rating on an item. Jamali and Ester in their work (Jamali and Ester 2010) introduce a novel probabilistic matrix factorization model based on the assumption that users' latent feature vectors are dependent on their social ties'. Last but not least, Wang et al. (2016; 2017a) propose to distinguish different tie types in social recommendation through presenting a method which can simultaneously classify strong and weak ties in a social network with respect to optimal recommendation accuracy as well as learn the latent vectors for users and items. We remark that all these models are learned in a continuous space, which in practice imposes severe challenges in the storage and computation cost when the number of users or items becomes large enough. The proposed discrete representation method could be easily applied for the above model.

Binary Code Collaborative Filtering . As the scale of social media data becomes larger and larger, the efficiency and scalability of recommendation methods are gradually drawing more attention. Collaborative filtering can essentially be treated as a similarity search process, the methodology of learning the binary codes (Salakhutdinov and Hinton 2009; Grauman and Fergus 2013) can serve as a promising way to enhance the efficiency and scalability of recommendation. Existing works on binary code collaborative filtering can be categorized into two groups: two-stage method and single-stage (i.e., unified framework) method. In general, two-stage method first obtains the user/item real-valued representations in a separate procedure, then performs the process of binarization on the learned representation to get the binary codes for users/items. This method may run the risk of ignoring the original data geometry and producing large quantization loss. Typical models include utilizing Locality-Sensitive Hashing (Gionis, Indyk, and Motwani 1999) to generate binary codes (Das et al. 2007), randomly projecting (Karatzoglou, Smola, and Weimer 2011) or rotating (Zhou and Zha 2012) features to obtain binary codes and imposing uncorrelated bit constraints to get binary codes (Liu et al. 2014) etc. Single-stage method (Zhang et al. 2016; Zhang, Lian, and Yang 2017; Lian et al. 2017; Zhang et al. 2018; Liu et al.), demonstrates its superiority over two-stage method by learning the binary codes for users and items in a unified framework so that the intrinsic geometry in real-valued vector space can be maintained and the quantization error can be dramatically reduced. However, all existing single-stage models do not take the important social information into account and thus fail to handle our problem in this work.

Discrete Social Recommendation

In this section, we introduce some preliminaries, present our proposed discrete social recommendation (DSR) which represents users and items by r-bit binary codes in a social recommendation setting, and then elaborate on the optimization procedure of DSR. Finally, we elucidate the model initialization method which has a large impact on a discrete model and the way to deal with cold-start issue.

Preliminaries

Given a user-item rating matrix $\mathbf{R}$ of size $n \times m$, matrix factorization based collaborative filtering aims to decompose $\mathbf{R}$ into two matrices, i.e., $\mathbf{P}^{r\times n}$ and $\mathbf{Q}^{r\times m}$ which represent the user preferences and the item preferences respectively, such that $\mathbf{R} \approx \mathbf{P}^\intercal\mathbf{Q}$. The i^{th} column of $\mathbf{P}$, denoted as $\mathbf{P}_i$, represents the latent feature vector of user i. Similarly, the j^{th} column of $\mathbf{Q}$, denoted as $\mathbf{Q}_j$, represents the latent feature vector of item j. As such, the rating of user i for item j is approximated by the dot product of $\mathbf{P}_i$ and $\mathbf{Q}_j$, i.e., $R_{ij} \approx \mathbf{P}_i^\intercal\mathbf{Q}_j$.

When considering social relationships among users (Ma et al. 2008), there comes an additional social relation matrix $\boldsymbol{S}$ of size $n \times n$ whose entry for row i and column k, i.e., S_{ik}, measures the relationships (e.g., degree of trust) between user i and user k. The approximation of $\mathbf{S}$ is similar to that of $\mathbf{R}$ through introducing a third latent feature matrix $\mathbf{T}^{r\times n}$, i.e., social factor matrix, satisfying $S_{ik} \approx \mathbf{P}_i^\intercal\mathbf{T}_k$:

$$min \sum_{i=1}^{n} \sum_{\mathbf{T}_k \in N_i} (S_{ik} - \mathbf{P}_i^\intercal\mathbf{T}_k)^2,$$

where N_i is the set of neighbors (direct connections) latent vectors of user i. Thus the overall objective is to minimize the squared loss for observed ratings Ω and social relationships Ψ:

$$\begin{aligned}\underset{\mathbf{P},\mathbf{Q},\mathbf{T}}{\operatorname{argmin}} \sum_{(i,j)\in\Omega} (R_{ij} - \mathbf{P}_i^\intercal\mathbf{Q}_j)^2 + \alpha_0 \sum_{(i,k)\in\Psi} (S_{ik} - \mathbf{P}_i^\intercal\mathbf{T}_k)^2 \\ + \alpha_1\|\mathbf{P}\|_F^2 + \alpha_2\|\mathbf{Q}\|_F^2 + \alpha_3\|\mathbf{T}\|_F^2, \end{aligned} \tag{1}$$

where $\|\cdot\|_F$ denotes the Frobenius norm of a matrix, α_0 is the hyperparameter that controls the relative importance of the rating matrix and the social relation matrix and $\alpha_1, \alpha_2, \alpha_3 > 0$ are hyperparameters which control the trade-off between low loss and low model complexity. In this work, we follow Ma et al. (2008) and factorize user-item rating matrix and user-user relationship matrix simultaneously, assuming that user rating information and user social information should share the same user latent feature vectors.

Discrete Social Recommendation (DSR)

When the dot product of those learned real-valued latent vectors is used for the purpose of similarity search, the total computation complexity is $\mathcal{O}(mnr + mn\log k)$, which may result in a crucial efficiency issue when the number of users n or the number of items m becomes quite large. Thus we propose discrete social recommendation (DSR) model to replace real-valued latent feature vectors with binary codes such that the computation efficiency of dot product between user and item binary codes can be improved through the Hamming distance. Formally, the dot product between the user and item binary codes, $\mathbf{b}_i \in \{\pm1\}^r$ and $\mathbf{d}_j \in \{\pm1\}^r$, can be formulated as

$$\mathbf{b}_i\mathbf{d}_j = 2H(\mathbf{b}_i, \mathbf{d}_j) - r,$$

where $H(\cdot,\cdot) = \sum_{k=1}^{r} \mathbb{I}(b_{ik} = d_{jk})$ is the Hamming distance between two binary codes and $\mathbb{I}(\cdot)$ denotes the indicator function that returns 1 if the element is true and 0 otherwise. Specifically, $H(\mathbf{b}_i, \mathbf{d}_j) = 0$ if all the bits between $\mathbf{b}_i$ and $\mathbf{d}_j$ are different and $H(\mathbf{b}_i, \mathbf{d}_j) = r$ if all the bits are the same. Thanks to the high efficiency of bit operations, similarity search through computing Hamming similarity can be accelerated up to logarithmic or even constant time complexity (Muja and Lowe 2009; Wang, Kumar, and Chang 2012).

Similar to the problem of conventional social recommendation in Equation (1), we can use the above similarity score to reconstruct the observed user-item ratings and user-user links. In order to maximize the information each bit carries and to assure the compactness of the learned binary codes, we impose the de-correlated constraint on the binary codes to guarantee the independence among different bits and place balance constraint on the learning process to ensure that the each bit carries as much information as possible. Given the observed user-item ratings index Ω and user-user links index Φ, the objective function of DSR model could be formulated as:

$$\underset{\mathbf{B},\mathbf{D},\mathbf{F}}{\operatorname{argmin}} \sum_{(i,j)\in\Omega} (R_{ij} - \boldsymbol{b}_i^\intercal\boldsymbol{d}_j)^2 + \alpha_0 \sum_{(i,k)\in\Psi} (S_{ik} - \boldsymbol{b}_i^\intercal\boldsymbol{f}_k)^2 \tag{2}$$

$$\textit{s.t. } \mathbf{B} \in \{\pm1\}^{r\times n}, \mathbf{D} \in \{\pm1\}^{r\times m}, \mathbf{F} \in \{\pm1\}^{r\times n}$$

$$\mathbf{B}\mathbf{1}_n = 0, \mathbf{D}\mathbf{1}_m = 0, \mathbf{F}\mathbf{1}_n = 0 \tag{3}$$

$$\mathbf{B}\mathbf{B}^\intercal = n\mathbf{I}_r, \mathbf{D}\mathbf{D}^\intercal = m\mathbf{I}_r, \mathbf{F}\mathbf{F}^\intercal = n\mathbf{I}_r, \tag{4}$$

where $\boldsymbol{b}_i \in \{\pm1\}^r$, $\boldsymbol{d}_j \in \{\pm1\}^r$ and $\boldsymbol{f}_k \in \{\pm1\}^r$ represent user binary codes, item binary codes and social binary codes respectively. We further stack the binary codes by column to form matrices, i.e., $\boldsymbol{B} \in \{\pm1\}^{r\times n}$, $\boldsymbol{D} \in \{\pm1\}^{r\times m}$, $\boldsymbol{F} \in \{\pm1\}^{r\times n}$. Due to the binary constraint, the regularization $\|\mathbf{B}\|_F^2 + \|\mathbf{D}\|_F^2 + \|\mathbf{F}\|_F^2$ in Equation (1) becomes an constant and hence is canceled. Equation (3) and (4) guarantees the informativeness and compactness of the learned binary codes. It is obvious that the proposed DSR model follows the same assumption that rating matrix $\mathbf{R}$ and social relation matrix $\mathbf{S}$ share the same user binary codes.

Optimizing DSR is essentially a challenging discrete optimization problem, since it is generally NP-hard. Specifically, finding the global optimum solution needs to involve $O(2^{(2rn+rm)})$ combinatorial search for the binary codes (Håstad 2001). Accordingly, we minimize this objective function in a computationally tractable way by softening the balance and de-correlated constraints (without discarding the discrete constraints $\mathbf{B} \in \{\pm1\}^{r\times n}, \mathbf{D} \in \{\pm1\}^{r\times m}, \mathbf{F} \in \{\pm1\}^{r\times n}$). To achieve this, we first define the delegate continuous variables $\mathbf{X}, \mathbf{Y}, \mathbf{Z}$, where $\mathbf{X} \in \mathcal{B} = \{\mathbf{X} \in \mathbb{R}^{r\times n} | \mathbf{X}\mathbf{1} = 0, \mathbf{X}\mathbf{X}^\intercal = n\mathbf{I}\}$, $\mathbf{Y} \in \mathcal{D} = \{\mathbf{Y} \in \mathbb{R}^{r\times m} | \mathbf{Y}\mathbf{1} = 0, \mathbf{Y}\mathbf{Y}^\intercal = m\mathbf{I}\}$, $\mathbf{Z} \in \mathcal{F} = \{\mathbf{Z} \in \mathbb{R}^{r\times n} | \mathbf{Z}\mathbf{1} = 0, \mathbf{Z}\mathbf{Z}^\intercal = n\mathbf{I}\}$. Then the balanced and de-correlated constraints on users and items could be softened by $d(\mathbf{B}, \mathcal{B}) = min_{\mathbf{X}\in\mathcal{B}}||\mathbf{B} - \mathbf{X}||_F$, $d(\mathbf{D}, \mathcal{D}) = min_{\mathbf{Y}\in\mathcal{D}}||\mathbf{D} - \mathbf{Y}||_F$, and $d(\mathbf{F}, \mathcal{F}) = min_{\mathbf{Z}\in\mathcal{F}}||\mathbf{F} - \mathbf{Z}||_F$ respectively. Thus, we for-

mulate the tractable objective function of DSR as follows:

$$\begin{aligned}\underset{\mathbf{B},\mathbf{D},\mathbf{F}}{\operatorname{argmin}} & \sum_{(i,j)\in\Omega}(R_{ij}-\mathbf{b}_i^\intercal\mathbf{d}_j)^2+\alpha_0\sum_{(i,k)\in\Sigma}(S_{ik}-\mathbf{b}_i^\intercal\mathbf{f}_k)^2 \\ & +\beta_1 d^2(\mathbf{B},\mathcal{B})+\beta_2 d^2(\mathbf{D},\mathcal{D})+\beta_3 d^2(\mathbf{F},\mathcal{F}) \quad (5) \\ \textit{s.t. } & \mathbf{B}\in\{\pm1\}^{r\times n},\mathbf{D}\in\{\pm1\}^{r\times m},\mathbf{F}\in\{\pm1\}^{r\times n},\end{aligned}$$

where $\beta_1, \beta_2, \beta_3$ are tuning parameters. We can find that large values of β_1, β_2 and β_3 will force $d^2(\mathbf{B},\mathcal{B}) = d^2(\mathbf{D},\mathcal{D}) = d^2(\mathbf{F},\mathcal{F}) = 0$ when DSR can be optimized under some feasible constraints. On the other side, the comparative small values of β_1, β_2 and β_3 provide a certain discrepancy between $\mathbf{B}$ and $\mathbf{X}$, $\mathbf{D}$ and $\mathbf{Y}$, $\mathbf{F}$ and $\mathbf{Z}$, making Equation (5) more flexible. Note that the norm of binary codes are constants and do not take any effect on regularization terms, we can replace the original regularization $d^2(\mathbf{B},\mathcal{B}), d^2(\mathbf{D},\mathcal{D}), d^2(\mathbf{F},\mathcal{F})$ with $tr(\mathbf{B}^\top\mathbf{X}), tr(\mathbf{D}^\top\mathbf{Y}), tr(\mathbf{F}^\top\mathbf{Z})$ for the ease of optimization, where $tr(\cdot)$ denotes the trace of a matrix. Besides, the above Equation (5) dose not discard the discrete constraint and still perform a discrete optimization over $\mathbf{B}, \mathbf{D}, \mathbf{F}$.

Optimizing DSR model

We employ alternating optimization strategy to solve the problem. Specifically, we take turns to update $\mathbf{B}$,$\mathbf{D}$,$\mathbf{F}$ and $\mathbf{X}$,$\mathbf{Y}$,$\mathbf{Z}$, given others fixed. Next we elaborate on how to solve each of the subproblems. Algorithm 1 depicts the details of the proposed DSR model.

Algorithm 1 Discrete Social Recommendation

Input:
$\{R_{ij}|i,j\in\mathcal{V}\}, \{S_{ij}|i,j\in\mathcal{N}\}$: observed user-item ratings and user-user similarity in trust relationship.

Initialize $\mathbf{B}$, $\mathbf{D}$, $\mathbf{F}$ and $\mathbf{X}$, $\mathbf{Y}$, $\mathbf{Z}$ by Equation (4)
repeat
  for $i = 1$ to n **do**
    repeat
    for $k = 1$ to r **do**
      Update b_{ik} via the Equation (7).
    end for
    until converge
end for
Update $\mathbf{D}$ and $\mathbf{F}$ according to Equation (10) and Equation (12) respectively.
Update $\mathbf{X}$ according to Equation (14).
Update of $\mathbf{Y}$ and $\mathbf{Z}$ in a similar way with $\mathbf{X}$.
until converge
return $\mathbf{B}, \mathbf{D}, \mathbf{F}$.

Optimizing $\mathbf{B}$. We update $\mathbf{B}$ via fixing $\mathbf{D}$, $\mathbf{F}$, $\mathbf{X}$, $\mathbf{Y}$, $\mathbf{Z}$. As the objective function sums over users independently, the updating procedure can be conducted through updating each $\mathbf{b}_i$ in parallel. Ignoring the unrelated terms such as $\sum_{i,j} R_{ij}^2$ in Equation (2), we need to solve the following optimization problem:

$$\begin{aligned}\underset{\mathbf{b}_i\in\{\pm1\}^r}{\operatorname{argmin}} & \ \mathbf{b}_i^\intercal(\sum_{j\in\mathcal{V}_i}\mathbf{d}_j\mathbf{d}_j^\intercal)\mathbf{b}_i-2(\sum_{j\in\mathcal{V}_i}R_{ij}\mathbf{d}_j^\intercal)\mathbf{b}_i-2\beta_1\mathbf{x}_i^\intercal\mathbf{b}_i, \\ & +\alpha_0\mathbf{b}_i^\intercal(\sum_{j\in\mathcal{N}_i}\mathbf{f}_j\mathbf{f}_j^\intercal)\mathbf{b}_i-2\alpha_0(\sum_{j\in\mathcal{N}_i}S_{ij}\mathbf{f}_j^\intercal)\mathbf{b}_i \quad (6)\end{aligned}$$

where $\mathcal{V}_i$ denotes the set of items rated by user i and $\mathcal{N}_i$ is the set of users having social relationships with user i. Due to the discrete constraints, we adopt a bitwise learning method, i.e., Discrete Coordinate Descent (Shen et al. 2015). In particular, denoting b_{ik} as the *k*-th bit of $\mathbf{b}_i$ and $\mathbf{b}_{i\bar{k}}$ as the other codes excluding b_{ik}, we update b_{ik} bit by bit with $\mathbf{b}_{i\bar{k}}$ fixed. Then, we update b_{ik} based on the following rule:

$$b_{ik} = sgn\Big(K(\hat{b}_{ik}, b_{ik})\Big), \quad (7)$$

where $\hat{b}_{ik} = \sum_{j\in\mathcal{V}_i}(R_{ij}-\mathbf{d}_{j\bar{k}}^\intercal\mathbf{b}_{i\bar{k}})d_{jk}+\alpha_0\sum_{j\in\mathcal{N}_i}(S_{ij}-\mathbf{f}_{j\bar{k}}^\intercal\mathbf{b}_{i\bar{k}})f_{jk}+\beta_1 x_{ik}$. and $K(i,j)=i$ if $i\neq 0$, and $K(i,j)=j$ otherwise and $sgn(\cdot)$ is a sign function mapping its input value to 1 or -1. Note that b_{ik} should only be updated when $\hat{b}_{ik}$ is not zero. To efficiently compute $\hat{b}_{ik}$, we rewrite its updating rule as:

$$\begin{aligned}\hat{b}_{ik} = & \sum_{j\in\mathcal{V}_i}R_{ij}d_{jk}-\sum_{j\in\mathcal{V}_i}\mathbf{d}_j^\intercal\mathbf{b}_i d_{jk}+\sum_{j\in\mathcal{V}_i}b_{ik}+\beta_1 x_{ik} \quad (8)\\ & +\alpha_0\sum_{j\in\mathcal{N}_i}S_{ij}f_{jk}-\alpha_0\sum_{j\in\mathcal{N}_i}\mathbf{f}_j^\intercal\mathbf{b}_i f_{jk}+\alpha_0\sum_{j\in\mathcal{N}_i}b_{ik},\end{aligned}$$

Optimizing $\mathbf{D}$. Similar to the procedure of optimizing $\boldsymbol{B}$, we optimize $\mathbf{D}$ by fixing $\mathbf{B}$, $\mathbf{F}$, $\mathbf{X}$, $\mathbf{Y}$ and $\mathbf{Z}$. Discarding terms irrelevant to $\mathbf{d}_i$, we can rewrite the objective function as follows:

$$\underset{\mathbf{d}_i}{\operatorname{argmin}} \quad \mathbf{d}_i^\intercal(\sum_{j\in\mathcal{V}_i}\mathbf{b}_j\mathbf{b}_j^\intercal)\mathbf{d}_i-2(\sum_{j\in\mathcal{V}_i}R_{ji}\mathbf{b}_j^\intercal)\mathbf{d}_i-2\beta_2\mathbf{y}_i^\intercal\mathbf{d}_i. \quad (9)$$

We can derive the updating rule of d_{ik} as follows:

$$\begin{aligned}\hat{d}_{ik} &= \sum_{j\in\mathcal{V}_i}R_{ij}b_{jk}-\sum_{j\in\mathcal{V}_i}\mathbf{b}_j^\intercal\mathbf{d}_i b_{jk}+\sum_{j\in\mathcal{V}_i}d_{ik}+\beta_2 y_{ik}, \\ d_{ik} &= sgn\Big(K(\hat{d}_{ik}, d_{ik})\Big). \quad (10)\end{aligned}$$

Optimizing $\mathbf{F}$. Similarly, we can learn binary code for $\mathbf{F}$ by solving:

$$\underset{\mathbf{f}_i}{\operatorname{argmin}} \quad \mathbf{f}_i^\intercal(\sum_{j\in\mathcal{N}_i}\mathbf{b}_j\mathbf{b}_j^\intercal)\mathbf{f}_i-2(\sum_{j\in\mathcal{N}_i}S_{ji}\mathbf{b}_j^\intercal)\mathbf{f}_i-2\beta_3\mathbf{z}_i^\intercal\mathbf{f}_i, \quad (11)$$

Based on the coordinate-descent approach, we update f_{ik} according to:

$$\begin{aligned}\hat{f}_{ik} &= \sum_{j\in\mathcal{V}_i}S_{ij}b_{jk}-\sum_{j\in\mathcal{V}_i}\mathbf{b}_j^\intercal\mathbf{f}_i b_{jk}+\sum_{j\in\mathcal{V}_i}f_{ik}+\beta_3 z_{ik} \\ f_{ik} &= sgn\Big(K(\hat{f}_{ik}, f_{ik})\Big). \quad (12)\end{aligned}$$

Optimizing $\mathbf{X}$. When fixing $\mathbf{B}$, learning $\mathbf{X}$ could be solved via optimizing the following problem:

$$\underset{\mathbf{x}}{\operatorname{argmax}} \quad tr(\mathbf{B}^\intercal \mathbf{X}), \quad s.t., \quad \mathbf{X1} = 0, \mathbf{X}\mathbf{X}^\intercal = n\mathbf{I}. \quad (13)$$

An analytic solution could be obtained via the aid of Singular Value Decomposition (SVD). Denote $\bar{\mathbf{B}}$ as a row-wise zero-mean matrix, where $\bar{B}_{ij} = B_{ij} - \frac{1}{n}\sum_{j=1}^{n} B_{ij}$. As we can decompose $\bar{\mathbf{B}}$ through $\bar{\mathbf{B}} = \mathbf{P}_b \sum_b \mathbf{Q}_b^\intercal$, where $\mathbf{P}_b$ and $\mathbf{Q}_b$ are the left and right singular vectors of $\bar{\mathbf{B}}$, then the closed-form updating rule of $\mathbf{X}$ can be written as follows:

$$\mathbf{X} = \sqrt{n}[\mathbf{P}_b \hat{\mathbf{P}}_b][\mathbf{Q}_b \hat{\mathbf{Q}}_b]^\intercal, \quad (14)$$

where $\hat{\mathbf{P}}_b$ are eigenvectors corresponding to zero eigenvalues and $\hat{\mathbf{Q}}_b$ can be obtained by Gram-Schmidt orthogonalization based on $[\mathbf{Q}_b \mathbf{1}]$. We could update $\mathbf{Y}$ and $\mathbf{Z}$ in a similar way with $\mathbf{X}$.

Computational Complexity. The proposed method converges quickly in practice, which took about $2 \sim 4$ iterations. For each iteration, the overall computational complexity of $\mathbf{B}$, $\mathbf{D}$ and $\mathbf{F}$ subproblem is $O\big(\#iter(m+n)r^2\big)$. In practice, $\#iter$ is usually $2 \sim 5$. The overall complexity of $\mathbf{X}$, $\mathbf{Y}$ and $\mathbf{Z}$ is $O\big(r^2 m\big)$ and $O\big(r^2 n\big)$, respectively.

Initialization. Since DSR solves a mixed-integer non-convex optimization problem, initialization plays an important role in achieving a good convergence and ideal local optimum. Here we introduce an efficient initialization strategy inspired by (Zhang et al. 2016). We relax the binary constraints in Equation (5) as follows:

$$\begin{aligned}\underset{\mathbf{U},\mathbf{V},\mathbf{T}}{\operatorname{argmin}} & \sum_{i,j}(R_{ij} - \mathbf{u}_i^T \mathbf{v}_j)^2 + \alpha \sum_{i,j}(S_{ij} - \mathbf{u}_i^T \mathbf{t}_j)^2 \\ & + \beta_1 ||\mathbf{U}||_F^2 + \beta_2 ||\mathbf{V}||_F^2 + \alpha\beta_3 ||\mathbf{T}||_F^2 \\ & + \beta_1 d^2(\mathbf{U}, \mathcal{U}) + \beta_2 d^2(\mathbf{V}, \mathcal{V}) + \beta_3 d^2(\mathbf{T}, \mathcal{T}),\end{aligned} \quad (15)$$

This initialization is quite similar to the traditional social recommendation in equation (1) except the balanced and decorrelated constraints imposed on real-valued $\mathbf{U}$, $\mathbf{V}$ and $\mathbf{T}$. To solve it, we first randomly initialize real-valued $\mathbf{U}$, $\mathbf{V}$, $\mathbf{T}$ and find feasible initializations for $\mathbf{X}$, $\mathbf{Y}$ and $\mathbf{Z}$ by optimizing them respectively. Assuming the solution is ($\mathbf{U}^*$, $\mathbf{V}^*$, $\mathbf{T}^*$, $\mathbf{X}^*$, $\mathbf{Y}^*$, $\mathbf{Z}^*$), then we can initialize the parameters in Equation (5) as:

$$\begin{aligned}\mathbf{B} = sgn(\mathbf{U}^*), \mathbf{D} = sgn(\mathbf{V}^*), \mathbf{F} = sgn(\mathbf{T}^*), \\ \mathbf{X} = \mathbf{X}^*, \mathbf{Y} = \mathbf{Y}^*, \mathbf{Z} = \mathbf{Z}^*.\end{aligned} \quad (16)$$

The effectiveness of the proposed initialization algorithm is illustrated in Figure 16(left).

Learning Binary Codes in Cold-start Problem. By taking social influence into consideration, our proposed DSR model can handle cold-start problem when users have no rating history in the training set but are associated with social link information. For a new user i, we could learn the binary codes according to its social tie j's ($j \in \mathcal{N}_i$) binary codes $\mathbf{f}_j$. The objective could be formulated as:

$$\underset{\boldsymbol{b}_i}{\operatorname{argmin}} \sum_i \sum_{j \in \mathcal{N}_i} (S_{ij} - \mathbf{b}_i^\intercal \mathbf{f}_j)^2, \quad s.t.\ \mathbf{B1}_n = 0, \mathbf{B}\mathbf{B}^\intercal = n\mathbf{I}_r.$$

.

Experiments

In this section, we carry out extensive experiments on several real-world datasets and compare our proposed model with several state-of-the-art algorithms to demonstrate the advantages of the proposed DSR approach.

Dataset	Rating#	User#	Item#	Ties(edges)#
FilmTrust	35497	1508	2071	1853
CiaoDVD	72700	17615	16121	40000
Epinions	664824	49290	139738	487181

Table 1: Detailed statistics of the datasets.

Datasets The evaluations are conducted on three public datasets from different real-world websites.

Epinions: This dataset is collected in a 5-week crawl (November/December 2003) from a product review website [1]. Users in Epinions are allowed to specify scores from 1 to 5 to rate items, and they can also establish relations with others.

FilmTrust: This dataset is extracted from the entire FilmTrust website [2] in June, 2011. The dataset contains user-user friendships and user-movie ratings.

CiaoDVD: This dataset is crawled from the entire category of DVDs on a UK DVD community website [3] in December, 2013. The dataset contains trust relationships among users as well as their ratings on DVDs.

The detailed statistics of the datasets is summarized in Table 1. When a user rates an item multiple times, we merge them into one rating by averaging the duplicate rating scores. For the in-matrix recommendation task, we randomly sample 80% ratings as training and the rest 20% as testing for each user. For the out-matrix recommendation task, we randomly sample 80% users and put all ratings made by them into the training set, and then put the ratings made by the rest users into the test set. We repeat for 5 random splits and report the average results of each algorithms.

Evaluation Metrics To validate the performance of top-K recommendation, we truncate the ranking list at position K and then evaluate the ranking list using Normalized Discounted Cumulative Gain (NDCG) and Recall. NDCG is a weighted sum of the degree of relevance for the ranked items. This metric takes into account both ranking precision and the positions of items in a ranking list. NDCG is position-sensitive, and it assigns higher score to "hits" at higher positions. The larger value of NDCG indicates higher accuracy of recommendation performance. We adopt the average NDCG with different cut off over all users as the final metric (denoted as NDCG@K). Recall, on the other hand, quantifies the fraction of consumed items that are in the top-K ranking list sorted in a descending order by their estimated relevance scores. We also test the proposed model and other comparative methods in term of the Recall with different cut off over all users (denoted as recall@K).

[1] http://www.epinions.com/

[2] http://trust.mindswap.org/FilmTrust/

[3] http:// dvd.ciao.co.uk

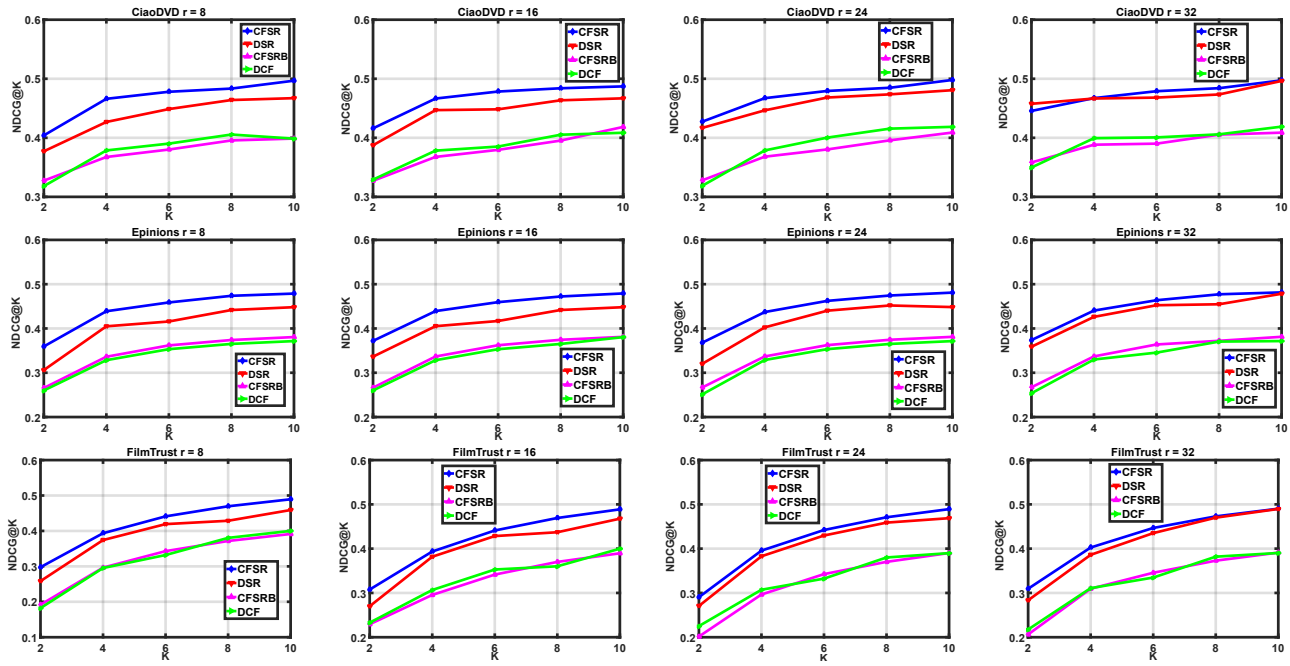

Figure 1: Performance of NDCG@K w.r.t., code length ranging from 8 to 32 in the in-matrix recommendation task.

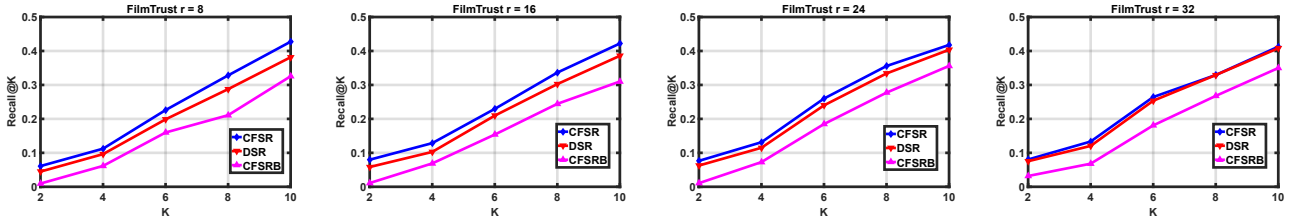

Figure 2: Performance of Recall@K w.r.t., code length ranging from 8 to 32 in the out-matrix recommendation task.

Comparative Methods We compare our method with several state-of-the-art approaches including binary code based collaborative filtering without considering any social information (DCF) and real-valued based social collaborative filtering method (CFSR) as well as its variant.

DCF: The discrete collaborative filtering (DCF) is the first one-stage method directly optimizing the binary codes in collaborative filtering (Zhang et al. 2016). This method outperforms almost all two-stage binary code learning methods for collaborative filtering. We note that DCF directly tackles a discrete optimization problem for seeking informative and compact binary codes without taking social information into account so it is not designed for social recommendation.

CFSR: The collaborative filtering for social recommendation model proposed by (Ma et al. 2008) which factorizes user-item rating matrix and user-user linkage matrix simultaneously. This is a classic social recommendation model widely adopted in many different applications. We use CFSR as a baseline to demonstrate the performance gap between real value representations and binary code representations.

CFSRB: Here binary codes are derived from CFSR through round-off binary quantization, which is a two-stage discrete learning method. We take this method as a baseline to show how quantization loss degrades the model's performance for two-stage binary code representations.

DSR: The proposed discrete social recommendation method which directly learns informative yet compact binary codes for users and items at the presence of social information.

We use grid search and 5-fold cross validation to find the best parameters. The hyper-parameters α_0 is tuned within the range of $[10^{-3}, \ldots, 10^{2}]$, β_1, β_2 are tuned within the range of $[10^{-5}, \ldots, 1]$, and β_3 is tuned within the range of $[10^{-4}, \ldots, 10^{1}]$. For baselines, we either adopt the optimal parameters reported in the original paper or choose the best we can obtain in our experiments. All the experiments are conducted on a computer equipped with an Intel(R) Core(TM) i5-7200U CPU @2.50GHZ, 16GB RAM and 64-bit Windows 10 operating system.

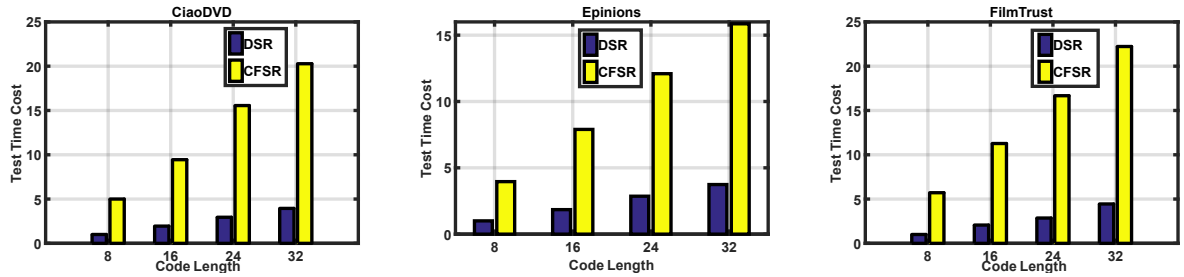

Figure 3: Efficiency comparison between DSR and CFSR w.r.t., TTC (minutes) where the code length ranges from 8 to 32 on three datasets.

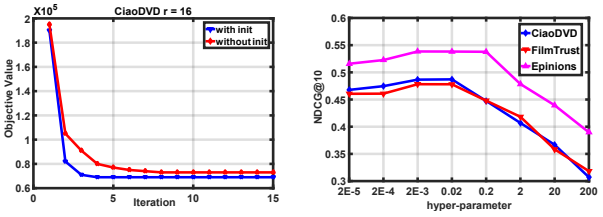

Figure 4: Convergence curve of the overall objective function with/without initialization on the CiaoDVD dataset(left). Sensitivity analysis of social parameter α_0(right).

Dataset	CFSR		DSR
	Memory	Reduction	Memory
CiaoDVD	1.32MB	×36	38KB
FilmTrust	671KB	×34	20KB
Epinions	14.57MB	×37	407KB

Table 2: Memory usage of DSR and CFSR with the same code length (32) on three datasets.

Experimental Results

Comparison with State-of-the-arts. In Figure 1, we compare DSR with the baseline methods whose code lengths vary from 8 to 32 in terms of NDCG@K on three datasets. From this figure, we can find that DSR considerably outperforms CFSRB on all datasets. Compared with CFSRB, DCF can even achieve comparable performance on CiaoDVD and Filmtrust datasets. This is consistent with the findings in (Zhang et al. 2016) that the direct discrete optimization is stronger than two-stage approaches, and the balanced uncorrelated constraints on the binary codes are necessary for the optimization. Besdies, DSR shows very competitive performance compared with CFSR. As the bit size increases, the performance gap between DSR and CFSR shrinks quickly. DSR is able to achieve the same performances with CFSR when the bit size is set to 32. One possible reason is that CFSR suffers from overfitting with the increasing latent feature vector dimensions, whereas binarizing the real-value latent feature vectors can alleviate the model's overfitting problem. This observation again verifies the effectiveness of the proposed DSR. Finally, DSR and CFSR demonstrate consistent improvement over DCF which does not take social information into consideration. This performance boost should be attributed to the benefit of utilizing social information for recommendation.

Performance on Cold-Start Users. We further drill down to the cold-start users. The comparison results with varying lengths of binary codes in the out-matrix task (in which DCF fails to make any recommendations and is therefore ignored) are shown in Figure 2. Similar to the results in in-matrix task, we observe that when code length is 32, DSR is able to achieve the same performance as CFSR.

Sensitivity Analysis of Hyperparameter α_0. Figure 4(right) shows the recommendation performance of DSR on NDCG@10 regarding the relative importance of the rating matrix and the social relation matrix. We can see that the performance continuously improves as we increase α_0 which validates the advantage of utilizing social relationship for recommendation. But the performance significantly drops as we keep increasing the impact of the social relationship matrix. The optimal setting of α_0 varies from different datasets.

Time and Memory Usage. Figure 3 displays the experiments on efficiency comparisons regarding test time cost (TTC) on three datasets. First, we find that DSR achieves significant speedups on all datasets w.r.t. TTC and is particularly much faster than CFSR, indicating the great advantage of binarizing the real-valued parameters adopted in CFSR. Second, the acceleration ratio of DSR against CFSR is stable on all the datasets with code lengths varying from 8 to 32. This experiment indicates that DSR is a suitable model for large-scale social recommender systems where the retrieval time for items is restricted within a limited time quota.

Last but not least, Table 2 shows the memory usage (storing the whole model) of DSR and CFSR with the same code length. It is obvious that DSR significantly reduces the memory storage of model for 37 times within the same dimension. The memory cost of DSR is less than 1M for storing binary codes from all three datasets. These results imply that DSR can adapt to some resource-limited scenarios, e.g., mobile devices.

Conclusion

In this paper, we propose a discrete social recommendation (DSR) model to learn informative yet compact binary

codes for users and items in the context of social recommendation. Given the untractability (NP-hardness) of optimizing DSR, we make some relaxations to the constraints and develop an efficient alternating optimization method to solve DSR through iteratively solving several mixed-integer programming subproblems. Extensive experiments on three real-world datasets demonstrate the advantages of our proposed method against several competitive baselines in terms of recommendation accuracy, retrieval cost and storage cost.

Acknowledgments

This research is supported by the National Research Foundation Singapore under its AI Singapore Programme [AISG-RP-2018-001], and the National Research Foundation, Prime Minister's Office, Singapore under its International Research Centres in Singapore Funding Initiative. It is also supported by China Postdoctoral Science Foundation No. BX201700136, National Program on Key Basic Research Project No. 2015CB352300, National Natural Science Foundation of China Major Project No. U1611461.

References

Das, A. S.; Datar, M.; Garg, A.; and Rajaram, S. 2007. Google news personalization:scalable online collaborative filtering. In *International Conference on World Wide Web*, 271–280.

Gionis, A.; Indyk, P.; and Motwani, R. 1999. Similarity search in high dimensions via hashing. 8(2):518–529.

Grauman, K., and Fergus, R. 2013. Learning binary hash codes for large-scale image search. *Machine learning for computer vision* 411(49-87):1.

Håstad, J. 2001. Some optimal inapproximability results. *Journal of the ACM (JACM)* 48(4):798–859.

Jamali, M., and Ester, M. 2010. A matrix factorization technique with trust propagation for recommendation in social networks. In *Proceedings of the fourth ACM conference on Recommender systems*, 135–142. ACM.

Karatzoglou, A.; Smola, A. J.; and Weimer, M. 2011. Collaborative filtering on a budget. 389–396.

Lian, D.; Liu, R.; Ge, Y.; Zheng, K.; Xie, X.; and Cao, L. 2017. Discrete content-aware matrix factorization. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 325–334. ACM.

Liu, H.; He, X.; Feng, F.; Nie, L.; Liu, R.; and Zhang, H. Discrete factorization machines for fast feature-based recommendation. In *Twenty-Seventh International Joint Conference on Artificial Intelligence ijcai, pages=3449-3455, year=2018,*.

Liu, X.; He, J.; Deng, C.; and Lang, B. 2014. Collaborative hashing. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2147–2154.

Liu, C.; Jin, T.; Hoi, S. C.; Zhao, P.; and Sun, J. 2017. Collaborative topic regression for online recommender systems: an online and bayesian approach. *Machine Learning* 106(5):651–670.

Ma, H.; Yang, H.; Lyu, M. R.; and King, I. 2008. Sorec: social recommendation using probabilistic matrix factorization. In *Proceedings of the 17th ACM conference on Information and knowledge management*, 931–940. ACM.

Ma, H.; Zhou, D.; Liu, C.; Lyu, M. R.; and King, I. 2011. Recommender systems with social regularization. In *Proceedings of the fourth ACM international conference on Web search and data mining*, 287–296. ACM.

Muja, M., and Lowe, D. G. 2009. Fast approximate nearest neighbors with automatic algorithm configuration. *VISAPP (1)* 2(331-340):2.

Salakhutdinov, R., and Hinton, G. 2009. Semantic hashing. *International Journal of Approximate Reasoning* 50(7):969–978.

Shen, F.; Shen, C.; Liu, W.; and Tao Shen, H. 2015. Supervised discrete hashing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 37–45.

Tang, J.; Hu, X.; and Liu, H. 2013. Social recommendation: a review. *Social Network Analysis and Mining* 3(4):1113–1133.

Wang, X.; Lu, W.; Ester, M.; Wang, C.; and Chen, C. 2016. Social recommendation with strong and weak ties. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, 5–14. ACM.

Wang, X.; Hoi, S. C.; Ester, M.; Bu, J.; and Chen, C. 2017a. Learning personalized preference of strong and weak ties for social recommendation. In *Proceedings of the 26th International Conference on World Wide Web*, 1601–1610. International World Wide Web Conferences Steering Committee.

Wang, X.; Hoi, S. C.; Liu, C.; and Ester, M. 2017b. Interactive social recommendation. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, 357–366. ACM.

Wang, J.; Kumar, S.; and Chang, S.-F. 2010. Semi-supervised hashing for scalable image retrieval. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, 3424–3431. IEEE.

Wang, J.; Kumar, S.; and Chang, S.-F. 2012. Semi-supervised hashing for large-scale search. *IEEE Transactions on Pattern Analysis & Machine Intelligence* (12):2393–2406.

Zhang, H.; Shen, F.; Liu, W.; He, X.; Luan, H.; and Chua, T.-S. 2016. Discrete collaborative filtering. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, 325–334. ACM.

Zhang, Y.; Yin, H.; Huang, Z.; Du, X.; Yang, G.; and Lian, D. 2018. Discrete deep learning for fast content-aware recommendation. In *the Eleventh ACM International Conference on Web Search and Data Minig*, 717–726.

Zhang, Y.; Lian, D.; and Yang, G. 2017. Discrete personalized ranking for fast collaborative filtering from implicit feedback. In *AAAI*, 1669–1675.

Zhou, K., and Zha, H. 2012. Learning binary codes for collaborative filtering. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, 498–506.

Zhou, Y.; Wilkinson, D.; Schreiber, R.; and Pan, R. 2008. Large-scale parallel collaborative filtering for the netflix prize. In *International Conference on Algorithmic Applications in Management*, 337–348. Springer.