Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17)

Discrete Personalized Ranking for Fast Collaborative Filtering from Implicit Feedback

Yan Zhang, Defu Lian,* Guowu Yang
Big Data Research Center, University of Electronic Science and Technology of China
yixianqianzy@gmail.com, dove@uestc.edu.cn, guowu@uestc.edu.cn

Abstract

Personalized ranking is usually considered as an ultimate goal of recommendation systems, but it suffers from efficiency issues when making recommendations. To this end, we propose a learning-based hashing framework called Discrete Personalized Ranking (DPR), to map users and items to a Hamming space, where user-item affinity can be efficiently calculated via Hamming distance. Due to the existence of discrete constraints, it is possible to exploit a two-stage learning procedure for learning binary codes according to most existing methods. This two-stage procedure consists of relaxed optimization by discarding discrete constraints and subsequent binary quantization. However, such a procedure has been shown resulting in a large quantization loss, so that longer binary codes would be required. To this end, DPR directly tackles the discrete optimization problem of personalized ranking. And the balance and un-correlation constraints of binary codes are imposed to derive compact but informatics binary codes. Based on the evaluation on several datasets, the proposed framework shows consistent superiority to the competing baselines even though only using shorter binary code.

Introduction

Recommender systems have been recently used in a growing number of e-commerce websites for helping their customers find desirable products to purchase. The ultimate goal of such systems is to present personalized ranking list for each user. However, a growing scale of users and products renders todays's recommendation much more challenging. Taking a Taobao user as an example, a recommendation system should make fast response to recommend products from a billion-scale collection by analyzing her browsing, purchasing and searching history.

Dimension reduction techniques such as matrix factorization has been shown to balance perfectly between accuracy and efficiency in recommendation. Such matrix factorization methods factorize a $n \times m$ user-item matrix to map both users and items into a joint r-dimensional latent space. Then users' preference for items could be efficiently predicted by inner product between them. To align with the ultimate goal of recommender systems, matrix factorization has been equipped with ranking based objective functions, such as WR-MF (Hu, Koren, and Volinsky 2008), BPR-MF (Rendle et al. 2009), CofiRank (Weimer et al. 2007), LA-LDA (Yin et al. 2015) and ListCF (Shi, Larson, and Hanjalic 2010), to directly generating a preference ranking of items for each user. To extract top-k preferred items for each user, recommendation systems need compute users' preference for all items and rank them by preference descendingly. Thus recommendation of time complexity $\mathcal{O}(nmr + nm \log k)$ is a critical efficiency bottleneck.

Fortunately, hashing techniques, encoding real-valued vectors into compact binary codes, could a promising approach to tackle this challenge, since inner product in this case could be efficiently achieved by bit operations, i.e., Hamming distance. One can even use a fast and accurate indexing methods for finding approximate top-K preferred items with sublinear or logarithmic time complexity (Wang, Kumar, and Chang 2012; Muja and Lowe 2009). Due to challenging discrete constraints, the learning of the compact binary codes is generally NP-hard (Håstad 2001), but can resort to a two-stage procedure (Liu et al. 2014b; Zhou and Zha 2012; Zhang et al. 2014), which consists of relaxed optimization via discarding the discrete constraints, and subsequent binary quantization. But according to (Zhang et al. 2016), these two-stage approaches oversimplify original discrete optimization, resulting in a large quantization loss, thus a principle hashing framework called Discrete Collaborative Filtering was proposed for direct discrete optimization. Unfortunately, the objective function of this framework is for rating prediction instead of for personalized item ranking, thus without aligning with the ultimate goal of recommendation system.

To this end, we propose a learning-based hashing framework called Discrete Personalized Ranking, which directly addresses the discrete optimization problem of personalized ranking from implicit feedback. In particular, DPR replaces user/item latent representation with binary codes and optimizes the objective – the Area Under ROC (Receiver Operating Characteristics) Curve (Rendle et al. 2009). The reason why targeting for implicit feedback is that personalized ranking in this case is more challenging due to the necessity of comparing the preference of each user's positively-preferred items with others. Additionally, DPR imposes the

*Corresponding author

balance and un-correlation constraints to derive the compact but informative codes. For the tractable discrete optimization of DPR, we develop an efficient alternating optimization method consisting of iteratively solving mixed-integer programming subproblems. Finally, we evaluate the proposed framework on three different sizes of datasets and show its consistent superiority to the competing baselines.

Related work

Below we mainly review recent advance of hashing-based collaborative filtering. For comprehensive reviews of hashing techniques, please refer to (Wang et al. 2016).

A pioneer work was to exploit Locality-Sensitive Hashing to generate hash codes for Google News readers based on their click history (Das et al. 2007). Following this, (Karatzoglou, Smola, and Weimer 2010) randomly projected user/item latent representation learned from regularized matrix factorization into the Hamming space, to obtain hash codes for users and items. Similar to this, (Zhou and Zha 2012) followed the idea of Iterative Quantization to generate binary code from rotated user/item latent representation. In order to derive more compact binary codes, before producing binary codes, the uncorrelated bit constraints was imposed on user/item latent representation in regularized matrix factorization (Liu et al. 2014b). However, according to analysis in (Zhang et al. 2014), hashing essentially only preserves similarity rather than inner product based preference, since user/item's magnitudes are lost in the subsequent binary quantization. Thus they imposed Constant Feature Norm (CFN) constraint when learning user/item latent representation, and then quantized their magnitudes and similarity respectively.

To summarize the aforementioned work, hashing codes are generated by two independent stages: relaxed learning of user/item latent representation and binary quantization. Since such two-stage methods suffer from a large quantization loss according to (Zhang et al. 2016), direct optimization for regularized matrix factorization with discrete constraints was proposed. To derive compact hash codes, the balance and uncorrelation constraints were further imposed. However, their algorithm was designed for rating prediction instead of for personalized ranking, so it is inconsistent with the ultimate goal of recommendation systems: providing a personalized ranking list of items.

Preliminary

In this section we will introduce some notations related to this paper. All of the vectors in this paper represent column vectors. Uppercase bold and lowercase bold letters denote matrices and vectors, respectively. Non-bold letters represent scalars.

Notations

Let user and item sets are denoted by $U = \{1, \cdots, n\}$ and $I = \{1 \cdots, m\}$ respectively. The implicit feedback S used in this paper is defined as a subset of the Cartesian product of U and I: $S \subseteq U \times I$. Examples for such feedback are product purchase/click/collection history in online shops, location visit history and music listening records. The task of personalized collaborative filtering is to provide a personalized total ranking of all items for each user. For convenience, following BPR (Rendle et al. 2009), we define the positive items for user u (on which the user has actions) as $I_u^+ = \{i \in I : (u, i) \in S\}$; and other items as $I_u^- = I \backslash I_u^+$. Users who have actions on item i are denoted as $U_i^+ = \{u \in U : (u, i) \in S\}$; other users are denoted as $U_i^- = U \backslash U_i^+$

AUC Objective

As introduced above, only positive instances are observed in implicit feedback datasets. The rest data is a mixture of actually negative and potentially positive data. In order to obtain a personalized total ranking of all items, one common approach is to predict a user's personalized preference scores $\hat{x}_{ui}$ for each item. Then the total ranking of items are determined by these scores. We denote $\hat{x}_{uij} = \hat{x}_{ui} - \hat{x}_{uj}$ as the comparison of $\hat{x}_{ui}$ and $\hat{x}_{uj}$. If $\hat{x}_{uij} > 0$, we can conclude that user u prefers item i over j; otherwise, user u prefers item j over i. For a particular user u, we expect that the goal that positive items will get higher scores than other items could be satisfied as much as possible, so that AUC is a commonly used objective. According to (Rendle et al. 2009), AUC per user is defined as

$$\text{AUC}(u) = \frac{1}{\left|I_u^+\right| \cdot \left|I_u^-\right|} \sum_{i \in I_u^+} \sum_{j \in I_u^-} \mathbb{I}\left(\hat{x}_{uij} > 0\right), \quad (1)$$

where $\mathbb{I}(t)$ is a delta function, which returns 1 if t is true and 0 otherwise. It ranges from zero to one, and it is larger if more pairs of items are preserved for comparative preference. The overall AUC averaged over all users is

$$\text{AUC} = \frac{1}{|U|} \sum_{u \in U} \text{AUC}(u)$$

For simplifying notations, we define D_S as $D_S = \{(u, i, j) | u \in U, i \in I_u^+ \text{ and } j \in I_u^-\}$, and rewrite AUC as

$$\text{AUC} = \sum_{(u,i,j) \in D_S} \frac{1}{|U| \left|I_u^+\right| \left|I_u^-\right|} \mathbb{I}\left(\hat{x}_{ui} > \hat{x}_{uj}\right)$$

However, optimizing AUC directly often leads to an NP-hard problem (Gao et al. 2013). A feasible solution in practice is to minimize some pairwise surrogate losses

$$L = \sum_{(u,i,j) \in D_S} \frac{1}{|U| \left|I_u^+\right| \left|I_u^-\right|} \ell(\hat{x}_{ui} - \hat{x}_{uj})$$

where $\ell : \mathbb{R} \to \mathbb{R}^+$ is a convex function such as exponential loss $\ell(t) = e^{-t}$, hinge loss $\ell(t) = \max(0, 1 - t)$, logistic loss $\ell(t) = \log(1 + e^{-t})$, least square loss $\ell(t) = (1 - t)^2$, etc.

Discrete Personalized Ranking

Problem formulation

In this paper, we propose a novel personalized ranking approach that directly optimizes the personalized ranking objective – AUC to learn effective hash codes for users and

items. As the least square loss is consistent with AUC (Gao et al. 2013) and could lead to efficient and closed forms for updating latent factors without sampling in an either continuous or discrete case. So we propose to use the least square loss $\ell(t) = (1-t)^2$, and to minimize the following pairwise least square loss:

$$\min \sum_{(u,i,j)\in D_S} \frac{1}{|U|\left|I_u^+\right|\left|I_u^-\right|}(1-(\hat{x}_{ui}-\hat{x}_{uj}))^2, \quad (2)$$

In this paper, we are interested in mapping users and items into r-bits binary codes for fast recommendation, where user-item affinity can be efficiently calculated via Hamming distance in the r-d Hamming space. Let $\mathbf{b}_u \in \{\pm 1\}^r$ and $\mathbf{d}_i \in \{\pm 1\}^r$ represent user codes and item codes respectively. We stack them by column into a matrix $\mathbf{B} \in \{\pm 1\}^{r\times n}$ and a $\mathbf{D} \in \{\pm 1\}^{r\times m}$, respectively. The user-item affinity (similarity) can be defined as (Zhou and Zha 2012):

$$\text{sim}(u,i) = \frac{1}{r}\sum_{k=1}^{r}\mathbb{I}(b_{uk}=d_{ik}) = \frac{1}{2}+\frac{1}{2r}\mathbf{b}_u^T\mathbf{d}_i$$

If $\text{sim}(u,i) > \text{sim}(u,j)$, the user u is more affinitive to item i than j. To preserve such affinity, we take the affinity as preference, then the preference of user u to item i is defined as

$$\hat{x}_{ui} = \frac{1}{2}+\frac{1}{2r}\mathbf{b}_u^T\mathbf{d}_i \quad (3)$$

Substituting Eq(3) into Eq (2), we rewrite the objective function

$$\arg\min_{\mathbf{B},\mathbf{D}} \sum_{(u,i,j)\in D_S} \frac{1}{|U|\left|I_u^+\right|\left|I_u^-\right|}\left(2r-\mathbf{b}_u^T(\mathbf{d}_i-\mathbf{d}_j)\right)^2$$

In oder to maximize the entropy of each binary bit, a balance constraint need be imposed, so that each bit carries as much information as possible (Zhou and Zha 2012). In order to learn compact binary codes, a un-correlation constraints is also imposed, so that each bit should be as independent as possible, that is, to remove the redundancy among the bits. Together with these two additional constraints on $\mathbf{B}$ and $\mathbf{D}$ respectively to maximize information load in short code (Weiss, Torralba, and Fergus 2009), we can formulate the objective function of the proposed Discrete Personalized Ranking (DPR) as follows:

$$\begin{aligned}\arg\min_{\mathbf{B},\mathbf{D}} &\sum_{(u,i,j)\in D_S} \frac{1}{|U|\left|I_u^+\right|\left|I_u^-\right|}\left(2r-\mathbf{b}_u^T(\mathbf{d}_i-\mathbf{d}_j)\right)^2 \\ &s.t.\ \mathbf{B}\in\{\pm1\}^{r\times n}, \mathbf{D}\in\{\pm1\}^{r\times m} \\ &\underbrace{\mathbf{B}\mathbf{1}_n=0, \mathbf{D}\mathbf{1}_m=0,}_{\text{Balance}}\ \underbrace{\mathbf{B}\mathbf{B}^T{=}n\mathbf{I}_r, \mathbf{D}\mathbf{D}^T{=}m\mathbf{I}_r}_{\text{Un-correlation}} \quad (4)\end{aligned}$$

Learning Model

Next we will introduce a learning model that can solve DPR in a computationally tractable manner. DPR in Eq(4) is essentially a discrete optimization problem, which has been proved as an NP-hard problem (Håstad 2001). Therefore, we adopt a strategy to solve DPR by softening the balance and decorrelation constraints (Zhang et al. 2016; Liu et al. 2014a). Let us define two sets $\mathcal{B} = \left\{\mathbf{X}\in\mathbb{R}^{r\times n}\middle|\mathbf{X}\mathbf{1}_n=0,\ \mathbf{X}\mathbf{X}^T{=}n\mathbf{I}_r\right\}$, $\mathcal{D} = \left\{\mathbf{Y}\in\mathbb{R}^{r\times m}\middle|\mathbf{Y}\mathbf{1}_m=0,\ \mathbf{Y}\mathbf{Y}^T{=}m\mathbf{I}_r\right\}$. We can soften these balance and un-correlation constraints to derive the following objective function, making Eq (4) computationally tractable,

$$\begin{aligned}\arg\min_{\mathbf{B},\mathbf{D}} &\sum_{(u,i,j)\in D_S} \frac{1}{|U|\left|I_u^+\right|\left|I_u^-\right|}\left(2r-\mathbf{b}_u{}^T(\mathbf{d}_i-\mathbf{d}_j)\right)^2 \\ &+\alpha d^2(\mathbf{B},\mathcal{B})+\beta d^2(\mathbf{D},\mathcal{D}) \\ &s.t.\ \mathbf{B}\in\{\pm1\}^{r\times n}, \mathbf{D}\in\{\pm1\}^{r\times m}, \quad (5)\end{aligned}$$

where $d(\mathbf{B},\mathcal{B}) = \min_{\mathbf{X}\in\mathcal{B}}\|\mathbf{B}-\mathbf{X}\|_F$ represents distance from a matrix $\mathbf{B}$ from the corresponding set $\mathcal{B}$, and $d(\mathbf{D},\mathcal{D}) = \min_{\mathbf{Y}\in\mathcal{D}}\|\mathbf{D}-\mathbf{Y}\|_F$ represents distance of a matrix $\mathbf{D}$ from the corresponding set $\mathcal{D}$. And $\alpha > 0$ and $\beta > 0$ are tuning parameters so that the second and third term of Eq (5) respectively allow certain discrepancy between the binary codes $\mathbf{B}$ and delegated values $\mathbf{X}$, and between $\mathbf{D}$ and $\mathbf{Y}$. Since $\text{tr}(\mathbf{B}\mathbf{B}^T) = \text{tr}(\mathbf{X}\mathbf{X}^T) = nr$ and $\text{tr}(\mathbf{D}\mathbf{D}^T) = \text{tr}(\mathbf{Y}\mathbf{Y}^T) = mr$, Eq (5) can be equivalently transformed to the following problem.

$$\begin{aligned}\arg\min_{\mathbf{B},\mathbf{D},\mathbf{X},\mathbf{Y}} &\sum_{(u,i,j)\in D_S} \frac{1}{|U|\left|I_u^+\right|\left|I_u^-\right|}\left(2r-\mathbf{b}_u{}^T(\mathbf{d}_i-\mathbf{d}_j)\right)^2 \\ &-2\alpha\text{tr}(\mathbf{B}^T\mathbf{X})-2\beta\text{tr}(\mathbf{D}^T\mathbf{Y}) \\ &s.t.\ \mathbf{B}\in\{\pm1\}^{r\times n}, \mathbf{D}\in\{\pm1\}^{r\times m} \\ &\mathbf{X}\mathbf{1}_n=0, \mathbf{Y}\mathbf{1}_m=0,\ \mathbf{X}\mathbf{X}^T{=}n\mathbf{I}_r, \mathbf{Y}\mathbf{Y}^T{=}m\mathbf{I}_r \quad (6)\end{aligned}$$

It's worth noting that we do not discard the binary constraint and directly optimize the substitution of AUC. Through alternating optimization for $\mathbf{B}$, $\mathbf{D}$, $\mathbf{X}$, and $\mathbf{Y}$, we can obtain nearly balanced and uncorrelated hashing codes for users and items. Next, we will introduce the alternating optimization. Below, for convenience, denote $z_u^+ = \frac{1}{|I_u^+|}$ and $z_u^- = \frac{1}{|I_u^-|}$.

B-subproblem: Fix $\mathbf{D}$, $\mathbf{X}$ and $\mathbf{Y}$, update $\mathbf{B}$.

Since the objective function in the problem (6) sums over users independently, we can update $\mathbf{B}$ by updating $\mathbf{b}_u$ in parallel by solving the following problem,

$$\begin{aligned}\arg\min_{\mathbf{b}_u\in\{\pm1\}^r} \sum_{i,j\in I} z_u^+ z_u^- r_{ui}(1-r_{uj})\Big(&((\mathbf{d}_i-\mathbf{d}_j)^T\mathbf{b}_u)^2 \\ &-4r(\mathbf{d}_i-\mathbf{d}_j)^T\mathbf{b}_u\Big)-2\alpha n\mathbf{x}_u^T\mathbf{b}_u \quad (7)\end{aligned}$$

where r_{ui} represent whether user u has action on item i, that is, returning 1 if $(u,i)\in S$ and 0 otherwise. However, this discrete optimization problem is generally NP-hard, so we adopt the bitwise learning method called Discrete Coordinate Descent (Shen et al. 2015) to update $\mathbf{b}_u$. In particular, denoting b_{uk} as the k-th bit of $\mathbf{b}_u$ and $\mathbf{b}_{u\bar{k}}$ as the rest codes excluding b_{uk}, Discrete Coordinate Descent will update b_{uk}

given $\mathbf{b}_{u\bar{k}}$ fixed. Discarding the terms independent to b_{uk}, the problem (7) could be rewritten as,

$$\arg\min_{b_{uk}\in\{\pm 1\}} b_{uk}\hat{b}_{uk} \tag{8}$$

where $\hat{b}_{uk} = \sum_{i,j} z_u^+ z_u^- r_{ui}(1-r_{uj})((\mathbf{d}_{i\bar{k}} - \mathbf{d}_{j\bar{k}})^T \mathbf{b}_{u\bar{k}}(d_{ik} - d_{jk}) - 2rd_{ik} + 2rd_{jk}) - \alpha n x_{uk}$. For efficient computation of this quantity, please refer to Appendix. Based on this optimization problem, we can easily deduce that the optimal b_{uk} is just the opposite sign of $\hat{b}_{uk}$. However, if $\hat{b}_{uk}$ equals to zero, b_{uk} should not be updated. Therefore, the update rule of $\mathbf{b}_{uk}$ is

$$b_{uk} = \text{sgn}\left(K(-\hat{b}_{uk}, b_{uk})\right) \tag{9}$$

where $K(t,r) = t$ if $t \neq 0$ and $K(t,r) = r$ otherwise. Out of insufficient space, and to keep the paper reasonably concise, the detailed derivation of Eq (8) is not presented in this paper.

D-subproblem: Fix $\mathbf{B}$, $\mathbf{X}$ and $\mathbf{Y}$, update $\mathbf{D}$.

As $\mathbf{B}$, $\mathbf{X}$ and $\mathbf{Y}$ fixed, discarding terms irrelevant to $\mathbf{d}_i$ in Eq(6), we can rewrite the objective function as follows:

$$\begin{aligned}\arg\min_{\mathbf{d}_i\in\{\pm 1\}^r} &\sum_{u\in U_i^+}\sum_{j\in I_u^-} z_u^+ z_u^- \left(2r - \mathbf{b}_u{}^T(\mathbf{d}_i - \mathbf{d}_j)\right)^2 \\ &+ \sum_{u\in U_i^-}\sum_{j\in I_u^+} z_u^+ z_u^- \left(2r - \mathbf{b}_u{}^T(\mathbf{d}_j - \mathbf{d}_i)\right)^2 \\ &- 2\beta n tr(\mathbf{D}^T\mathbf{Y}).\end{aligned} \tag{10}$$

Making use of r_{ui} to represent whether user u has action on item i, we can deduce the following objective.

$$\begin{aligned}\arg\min_{\mathbf{d}_i\in\{\pm 1\}^r} &\sum_{u,j} z_u^+ z_u^- r_{ui}(1-r_{uj})\left(\left(\mathbf{b}_u^T(\mathbf{d}_i - \mathbf{d}_j)\right)^2 - 4r\mathbf{b}_u^T\mathbf{d}_i\right) \\ &+ \sum_{u,j} z_u^+ z_u^- r_{uj}(1-r_{ui})\left(\left(\mathbf{b}_u^T(\mathbf{d}_j - \mathbf{d}_i)\right)^2 + 4r\mathbf{b}_u^T\mathbf{d}_i\right) \\ &- 2\beta n \mathbf{y}_i{}^T\mathbf{d}_i.\end{aligned} \tag{11}$$

Similar to B-subproblem, we can derive the update rule of d_{ik} as follows:

$$d_{ik} = \text{sgn}\left(K(-\hat{d}_{ik}, d_{ik})\right) \tag{12}$$

where

$$\begin{aligned}\hat{d}_{ik} = &\sum_{u,j} z_u^+ z_u^- r_{ui}(1-r_{uj})(-d_{jk} - 2rb_{uk}) + \\ &+ \sum_{u,j} z_u^+ z_u^- r_{ui}(1-r_{uj})\left(\mathbf{b}_{u\bar{k}}^T(\mathbf{d}_{i\bar{k}} - \mathbf{d}_{j\bar{k}})b_{uk}\right) \\ &+ \sum_{u,j} z_u^+ z_u^- r_{uj}(1-r_{ui})\left(-\mathbf{b}_{u\bar{k}}^T(\mathbf{d}_{j\bar{k}} - \mathbf{d}_{i\bar{k}})b_{uk}\right) \\ &\sum_{u,j} z_u^+ z_u^- r_{uj}(1-r_{ui})(-d_{jk} + 2rb_{uk}) - \beta n y_{ik}.\end{aligned} \tag{13}$$

For its efficient computation , please refer to Appendix.

X-subproblem: Fix $\mathbf{B}$, $\mathbf{D}$ and $\mathbf{Y}$, update $\mathbf{X}$.

The X-subproblem is

$$\underset{\mathbf{X}\in\mathbb{R}^{r\times n}}{\text{argmax}}\ tr\left(\mathbf{B}^T\mathbf{X}\right),\ s.t.\ \mathbf{X1} = 0,\ \mathbf{X}\mathbf{X}^T{=}n\mathbf{I}.$$

It can be solved by the aid of SVD according to (Liu et al. 2014a). In particular, $\mathbf{X}$ can be updated by

$$\mathbf{X} = \sqrt{n}[\mathbf{P}_b\ \widehat{\mathbf{P}}_b][\mathbf{Q}_b\ \widehat{\mathbf{Q}}_b]^T, \tag{11}$$

where $\mathbf{P}_b$ and $\mathbf{Q}_b$ are the left and right singular vectors of the row-centered matrix $\bar{\mathbf{B}}$: $\bar{b}_{iu} = b_{iu} - \frac{1}{n}\sum_{u=1}^{n} b_{iu}$, $\widehat{\mathbf{P}}_b$ is stacked by the left singular vectors of zero singular values and $\widehat{\mathbf{Q}}_b$ can be calculated by Gram-Schmidt orthogonalization based on $[\mathbf{Q}_b\ \mathbf{1}]$.

Y-subproblem: Fix $\mathbf{B}$, $\mathbf{D}$ and $\mathbf{X}$, update $\mathbf{Y}$. The Y-subproblem is

$$\underset{\mathbf{Y}\in\mathbb{R}^{r\times m}}{\arg\max}\ tr\left(\mathbf{D}^T\mathbf{Y}\right),\ s.t.\ \mathbf{Y1} = 0,\ \mathbf{Y}\mathbf{Y}^T{=}m\mathbf{I}.$$

Similar to the X-subproblem, $\mathbf{Y}$ can be updated by

$$\mathbf{Y} = \sqrt{m}[\mathbf{P}_d\ \widehat{\mathbf{P}}_d][\mathbf{Q}_d\ \widehat{\mathbf{Q}}_d]^T. \tag{12}$$

Initialization

Due to learning DPR is a mixed-integer non-convex optimization, initialization is significant for better convergence and local optimal solution. In order to achieve an efficient initialization, we initialize $\mathbf{B}$,$\mathbf{D}$, $\mathbf{X}$ and $\mathbf{Y}$ by relaxing the discrete constraints of $\mathbf{B}$,$\mathbf{D}$ in Eq (6). In particular, in Eq (6), substitute $\mathbf{B}$ with $\sqrt{2r}\mathbf{P}$ and substitute $\mathbf{D}$ with $\sqrt{2r}\mathbf{Q}$. Eq (6) will be finally equivalent to

$$\begin{aligned}\arg\min_{\mathbf{P},\mathbf{Q},\mathbf{X},\mathbf{Y}} &\sum_{(u,i,j)\in D_S} \frac{1}{|U|\,|I_u^+|\,|I_u^-|}\left(1 - \mathbf{p}_u^T(\mathbf{q}_i - \mathbf{q}_j)\right)^2 \\ &+ \frac{\alpha}{2r}\|\mathbf{P}\|_F^2 + \frac{\beta}{2r}\|\mathbf{Q}\|_F^2 - \frac{\alpha\sqrt{2r}}{2r^2}tr(\mathbf{P}^T\mathbf{X}) - \frac{\beta\sqrt{2r}}{2r^2}tr(\mathbf{Q}^T\mathbf{Y}) \\ &s.t.,\ \mathbf{X1} = 0, \mathbf{Y1} = 0,\ \mathbf{X}\mathbf{X}^T{=}n\mathbf{I}, \mathbf{Y}\mathbf{Y}^T{=}m\mathbf{I},\end{aligned}$$

where $\mathbf{P} \in \mathbb{R}^{r\times n}$, $\mathbf{Q} \in \mathbb{R}^{r\times m}$. The user-item affinity can be approximated by the dot product of real latent features of users and items, $\widehat{x}_{ui} = p_u^T q_i$.

Making use of r_{ui} to represent whether user u has action on item i, we can rewrite the initialization problem as follows:

$$\begin{aligned}\arg\min_{\mathbf{P},\mathbf{Q},\mathbf{X},\mathbf{Y}} &\sum_{u,i,j} z_u^+ z_u^- r_{ui}(1-r_{uj})\left(1 - \mathbf{p}_u^T(\mathbf{q}_i - \mathbf{q}_j)\right)^2 + \\ &\alpha_1 n\|\mathbf{P}\|_F^2 + \beta_1 n\|\mathbf{Q}\|_F^2 - 2\alpha_2 n tr(\mathbf{P}^T\mathbf{X}) - 2\beta_2 n tr(\mathbf{Q}^T\mathbf{Y}) \\ &s.t.\ \mathbf{X}\mathbf{1}_n = 0, \mathbf{Y}\mathbf{1}_m = 0,\ \mathbf{X}\mathbf{X}^T = n\mathbf{I}_r, \mathbf{Y}\mathbf{Y}^T = m\mathbf{I}_r,\end{aligned} \tag{14}$$

where α_1, α_2, β_1 and β_2 are the corresponding parameters. In order to solve the objective function of Eq(14), we

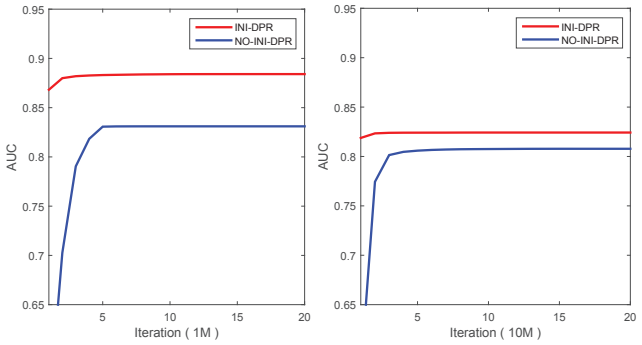

Figure 1: Convergence curve of AUC with/without initialization on MovieLens-1M/10M

also adopt alternating optimization. $\mathbf{P}$ and $\mathbf{Q}$ are randomly initialized. They are updated based on the updating rules through setting the corresponding derivative of the objective function to zero. $\mathbf{X}$ and $\mathbf{Y}$can also be solved by X/Y-subproblem introduced in the previous subsection. Assume the solutions are $\mathbf{P}^0$, $\mathbf{Q}^0$, $\mathbf{X}^0$ and $\mathbf{Y}^0$, we can initialize DPR framework as

$$\mathbf{B} \leftarrow \text{sgn}\left(\mathbf{P}^0\right),\ \mathbf{D} \leftarrow \text{sgn}\left(\mathbf{Q}^0\right),\ \mathbf{X} \leftarrow \mathbf{X}^0,\ \mathbf{Y} \leftarrow \mathbf{Y}^0. \tag{15}$$

It's easy to see that the above initialization is feasible to Eq(6). The effectiveness of the initialization is illustrated in Figure 1. We can see that the proposed initialization scheme can help to achieve faster convergence and better performance on personalized ranking. The objective functions' convergence of the initialization and DPR framework are demonstrated on MovieLens-1M shown in Figure 2. To keep this paper concise, theoretical analysis of convergence is omitted.

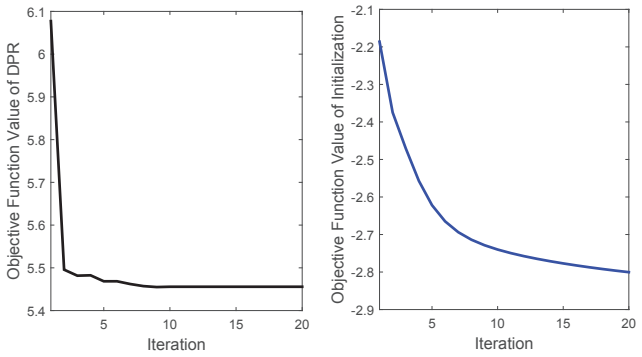

Figure 2: Convergence of DPR/Initialization on MovieLens-1M

Experiments

In this section, we mainly introduce our experiment settings and analyze the results. Experiments on three real world datasets show that DPR recommendation framework outperforms existing hashing based recommendation methods in personalized ranking.

Experiment Settings

We evaluate our method on three real world open datasets: MovieLens-1M, MovieLens-10M, and a subset of Netflix-100M. The algorithm can also be applied in larger datasets because the training procedure scales linearly with the data size, which will be shown in Figure 4. MovieLens-1M contains 1,000,209 ratings from 6040 users to 3706 movies. MovieLens-10M includes 10,000,054 ratings from 69878 users to 10677 movies. For Netflix dataset, we use a subset selected by bootstrap sampling from Netflix dataset which contains 10,387,786 ratings from 20,000 users to 7,000 movies. All of these ratings are within [0,5]. As we focus on an implicit feedback task in this paper, we remove rating scores from datasets and keep rating actions in S.

For all datasets, we select subsamples such that each user u has at least 10 positive items ($|I_u^+| \geq 10$) and each item i has at least 10 positive users ($\left|U_i^+\right| \geq 10$). We use leave one out method to evaluate our scheme and tune hyper-parameters in Eq(6) and Eq(14) . The hyper-parameters α and β are tuned within $[10^{-4}, 10^{-2}]$ and $[10^{-3}, 10^{-1}]$ respectively.

Comparison Methods

BPR-MF: This is a Bayesian Personalized Ranking framework based on Matrix Factorization, which directly optimized the ranking based evaluation with Bayesian. To align with the ultimate goal of recommender systems, matrix factorization has been equipped with ranking based objective functions. BPR-MF learns the real latent factors of users and items by BPR-OPT (Rendle et al. 2009). Users' preferences for items are ranked by inner products between real latent factors.

DCF: This is a Discrete Collaborative Filtering framework that directly learns binary codes with the discrete constraints. But the objective function of this framework is rating prediction instead of personalized items ranking, thus without aligning with the ultimate goal of recommendation system.

PPH: This is a two-stage Preference Preserving Hashing framework. Different from traditional MF (Matrix Factorization model), they emphasize the important of real latent feature norm. They imposed Constant Feature Norm (CFN) constraint when learning user/item latent representation, and then quantized their magnitudes and similarity respectively. PPH quantized each real latent vector into $(r-2)$-bit phase codes and $2-$bit magnitude codes. Hence, in order to keep the code length consistent to our framework, we only learned $(r-2)$-dim real latent features at the relaxed optimization stage.

Results analysis

We evaluate the recommendation performance of all methods above by AUC, which is widely used for assessing ranking based tasks.

Figure 3 shows the personalized ranking performances (AUC) of DPR and the three state-of-the-art collaborative filtering methods on three real world datasets. We make the following observations from the experimental results:

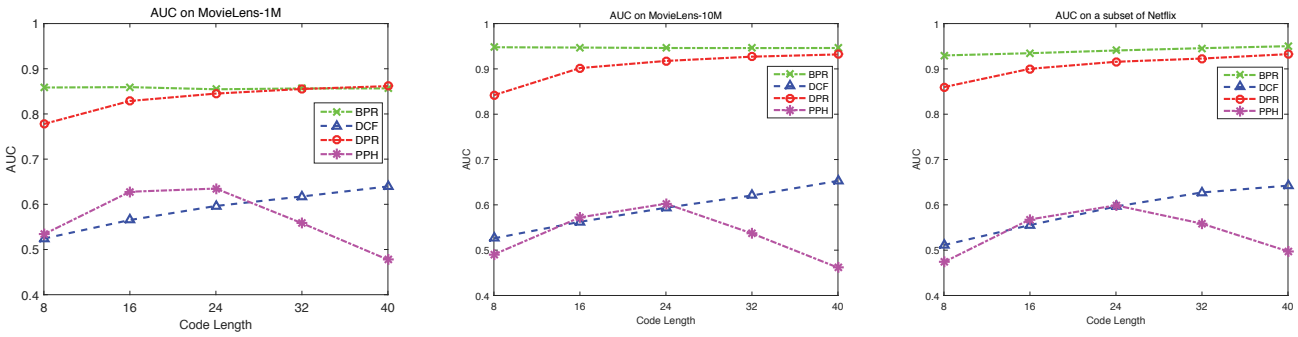

Figure 3: AUC on MovieLens-1M, 10M and a subset of Netflix

(1) Compared with BPR-MF, the performance of our method is very close to the classic BPR method. However, BPR-MF is a personalized ranking framework based on real latent features. For the consideration of storage and time cost, hash techniques shows superiority to methods based on real valued data as introduced in the 3rd paragraph of the Introduction Section. Specifically, we obtain compact and informatics binary codes under the constraints of balance and un-correlation. So that DPR can even achieve similar performance by using only 32-bit binary codes as compared to BPR-MF using 32-dim real features. This suggests that DPR can reduce a huge amount of space cost. Besides, Fast top-k recommendation using hash codes of users and items is considered as a hash-based retrieval problem, whose efficiency has been theoretically and empirically studied in information retrieval fields, as introduced in the Introduction Section.

(2) Compared with existing hashing based methods, the proposed DPR shows consistent superiority to the competing baselines, such as DCF and PPH. Because DCF aims to optimize rating square loss instead of items ranking, which leads to ranking performance is not very good. PPH adopt a two-stage hashing scheme which brings large information loss. While DPR proposed in this paper aims to optimize ranking task directly and adopt an alternating optimization to learn compact and informatics binary codes. So DPR can even achieve better performance by using only 8 bits as compared to DCF and PPH using 40 bits.

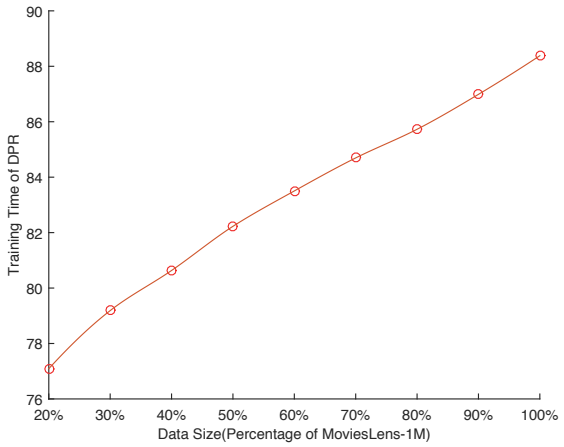

Figure 4: Training time varies with data size on MovieLens-1M

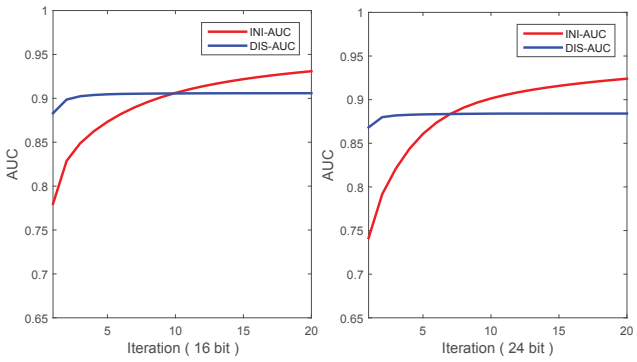

Figure 5: Convergence curve of AUC for Initialization/DPR on MovieLens-1M

We ran the proposed algorithm given different size of the training data and investigated the variation of running time, whose result is shown in Figure 4. From this figure, we see that the training procedure scales linearly the data size, the same as BPR. So hashing framework proposed in this paper can be extended to larger datasets. From Figure 5, we can conclude that the alternating optimization is an effective method to solve the discrete optimization problem proposed in this paper because the AUC is also converged in the process of training.

Conclusion

In this paper, we developed a learning-based hashing framework DPR, which directly addresses the discrete optimization problem of personalized ranking from implicit feedback. Additionally, DPR imposes the balance and uncorrelation constraints to derive the compact but informative codes. For the tractable discrete optimization of DPR, we develop an efficient alternating optimization method consisting of iteratively solving mixed-integer programming subproblems. Based on the evaluation on several datasets, the pro-

posed framework shows consistent superiority to the competing baselines.

Acknowledgments

We thank Dr. Zi Huang and Dr. Hongzhi Yin for their suggestions and also the anonymous reviewers for their feedback. This work is supported by grants from the National Natural Science Foundation of China (61272175, 61572109, 61502077, 61631005), the 863 High Technology Plan (2015AA01A707) and the Fundamental Research Funds for the Central Universities (ZYGX2014Z012).

Appendix

Fast calculation of $\hat{b}_{uk}$ in B-subproblem

$\hat{b}_{uk}$ can be rewritten as

$$\hat{b}_{uk} = \left(-\mathbf{b}_u^T\overline{\mathbf{d}_u^-} + b_{uk}\overline{d_{uk}^-}\right)\overline{d_{uk}^+} - \left(\mathbf{b}_u^T\overline{\mathbf{d}_u^+} - b_{uk}\overline{d_{uk}^+}\right)\overline{d_{uk}^-} + \left(\mathbf{b}_u^T\overline{(d_k\mathbf{d})_u^+} - 2r\overline{d_{uk}^+} - b_{uk}\right) + \left(\mathbf{b}_u^T\overline{(d_k\mathbf{d})_u^-} + 2r\overline{d_{uk}^-} - b_{uk}\right) - \alpha n x_{uk}. \quad (16)$$

Fast calculation of $\hat{d}_{ik}$ in D-subproblem

$\hat{d}_{ik}$ can be rewritten as

$$\widehat{d}_{ik} = \sum_u z_u^- (1 - r_{ui}) \left(\mathbf{b}_u^T b_{uk}\right) \left(\mathbf{d}_i - \overline{\mathbf{d}_u^+}\right) + \sum_u z_u^+ r_{ui} \left(\mathbf{b}_u^T b_{uk}\right) \left(\mathbf{d}_i - \overline{\mathbf{d}_u^-}\right) - \left(\overline{z_i^-} + \overline{z_i^+}\right) d_{ik} + 2r\overline{b_{ik}^-} - 2r\overline{b_{ik}^+} - \beta n y_{ik}. \quad (17)$$

Table 1: Sub-formulation

$\overline{\mathbf{d}_u^+} = \sum_i r_{ui}\mathbf{d}_i z_u^+$	$\overline{\mathbf{d}_u^-} = \sum_j (1 - r_{uj})\,\mathbf{d}_j z_u^-$
$\overline{(d_k\mathbf{d})_u^+} = \sum_i r_{ui} d_{ik}\mathbf{d}_i z_u^+$	$\overline{(d_k\mathbf{d})_u^-} = \sum_j (1 - r_{uj})\,d_{jk}\mathbf{d}_j z_u^-$
$\overline{b_{ik}^+} = \sum_u z_u^+ r_{ui} b_{uk}$	$\overline{b_{ik}^-} = \sum_u z_u^- (1 - r_{ui})\,b_{uk}$
$\overline{z_i^+} = \sum_u z_u^+ r_{ui}$	$\overline{z_i^-} = \sum_u z_u^- (1 - r_{ui})$

References

Das, A. S.; Datar, M.; Garg, A.; and Rajaram, S. 2007. Google news personalization: scalable online collaborative filtering. In *Proc. of WWW*, 271–280. ACM.

Gao, W.; Jin, R.; Zhu, S.; and Zhou, Z.-H. 2013. One-pass auc optimization. In *ICML (3)*, 906–914.

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

Hu, Y.; Koren, Y.; and Volinsky, C. 2008. Collaborative filtering for implicit feedback datasets. In *ICDE*, 263–272. IEEE.

Karatzoglou, A.; Smola, A. J.; and Weimer, M. 2010. Collaborative filtering on a budget. In *AISTATS*, 389–396.

Liu, W.; Mu, C.; Kumar, S.; and Chang, S.-F. 2014a. Discrete graph hashing. In *Advances in Neural Information Processing Systems*, 3419–3427.

Liu, X.; He, J.; Deng, C.; and Lang, B. 2014b. Collaborative hashing. In *Proc. of CVPR*, 2139–2146.

Muja, M., and Lowe, D. G. 2009. Fast approximate nearest neighbors with automatic algorithm configuration. In *Proc. International Conference on Computer Vision Theory and Applications, 2009*, 331–340.

Rendle, S.; Freudenthaler, C.; Gantner, Z.; and Schmidt-Thieme, L. 2009. Bpr: Bayesian personalized ranking from implicit feedback. In *Proceedings of the twenty-fifth onference on uncertainty in artificial intelligence*, 452–461. AUAI Press.

Shen, F.; Shen, C.; Liu, W.; and Tao Shen, H. 2015. Supervised discrete hashing. In *CVPR*, 37–45.

Shi, Y.; Larson, M.; and Hanjalic, A. 2010. List-wise learning to rank with matrix factorization for collaborative filtering. In *Proceedings of the fourth ACM conference on Recommender systems*, 269–272. ACM.

Wang, J.; Liu, W.; Kumar, S.; and Chang, S.-F. 2016. Learning to hash for indexing big dataa survey. *Proc. of the IEEE* 104(1):34–57.

Wang, J.; Kumar, S.; and Chang, S.-F. 2012. Semi-supervised hashing for large-scale search. *IEEE TPAMI* 34(12):2393–2406.

Weimer, M.; Karatzoglou, A.; Le, Q. V.; and Smola, A. 2007. Maximum margin matrix factorization for collaborative ranking. *Advances in neural information processing systems* 1–8.

Weiss, Y.; Torralba, A.; and Fergus, R. 2009. Spectral hashing. In Koller, D.; Schuurmans, D.; Bengio, Y.; and Bottou, L., eds., *Advances in Neural Information Processing Systems 21*. Curran Associates, Inc. 1753–1760.

Yin, H.; Cui, B.; Chen, L.; Hu, Z.; and Zhang, C. 2015. Modeling location-based user rating profiles for personalized recommendation. *ACM TKDD* 9(3):19.

Zhang, Z.; Wang, Q.; Ruan, L.; and Si, L. 2014. Preference preserving hashing for efficient recommendation. In *Proc. of SIGIR*, 183–192. ACM.

Zhang, H.; Shen, F.; Liu, W.; He, X.; Luan, H.; and Chua, T.-S. 2016. Discrete collaborative filtering. In *Proc. of SIGIR*, volume 325–334.

Zhou, K., and Zha, H. 2012. Learning binary codes for collaborative filtering. In *Proc. of ACM SIGKDD*, 498–506. ACM.