Dot-Product Based Preference Preserved Hashing for Fast Collaborative Filtering

Yan Zhang, Guowu Yang, Lin Hu*, Hong Wen*, Jinsong Wu†
Big Data Research Centre, School of Computer Science and Engineering,
University of Electronic Science and Technology of China, Chengdu, 611731, China
*National Key Lab of Commun., University of Electronic Science and Technology of China, Chengdu, 611731, China
†Department of Electrical Engineering, Universidad de Chile, Santiago, Chile
Emails: yixianqianzy@gmail.com, guowu@uestc.edu.cn, lin.hu.uestc@gmail.com, sunlike@uestc.edu.cn, wujs@ieee.org

***Abstract*—Recommendation is widely used to deal with information overloading by suggesting items based on historical information of users. One of the most popular recommendation techniques is matrix factorization (MF), in which the preferences of users are estimated by dot products of their real latent factors between users and items. Although MF can achieve high recommendation accuracy, it suffers from efficiency issues when making preferences ranking in real space. Hash retrieval technique can be applied to recommender systems to speed up preferences ranking. Due to the existence of discrete constraints in learning hash codes, it is possible to exploit a two-stage learning procedure according to most existing methods. This two-stage procedure consists of relaxed optimization by discarding discrete constraints and subsequent binary quantization. However, existing methods have not been able to well handle the change of dot product arising from quantization. To this end, we propose a dot-product based preference preserved hashing method, which quantizes both norm and cosine similarity in dot product respectively. We also design an algorithm to optimize the bit length for norm quantization. Based on the evaluation to several datasets, the proposed framework shows consistent superiority to the competing baselines even though only using shorter binary code.**

***Index Terms*—Recommender system, preference ranking, hash codes, norm quantization.**

I. Introduction

Recommender systems have been recently used in a large number of e-commerce websites to support their customers in finding desirable products for purchase. Personalized recommender systems try to match social goods with tastes of users, which has been widely applied in many domains, such as e-commerce [1], social networks [2], online education systems [3] and medical systems. For Amazon online shopping, fast response is required when users search products. So a recommender system should perform fast responses to recommend products from a billion-scale products collection. However, the recommendation accuracy and efficiency are hard to balance.

It has been shown that dimension reduction techniques, such as matrix factorization, is able to balance perfectly between accuracy and efficiency in recommendation [4]. Such matrix factorization methods factorize a $n \times m$ user-item rating matrix to map both users and items into a joint b-dimensional latent space, where m and n are the number of items and users in a dataset, respectively. Then the preferences of users over items could be efficiently predicted by inner product between them. In Netflix competition, several variants of MF were applied for movie recommendations [4], demonstrating that MF-based models are superior to neighbor-based recommendation techniques. However, due to the large size of item sets, items ranking for all users is time-consuming. The total time complexity is $\mathcal{O}(nm)$. On the entire set of Netflix dataset, such preferences ranking takes several days [5]. Generally speaking, the real world datasets are very large, even larger than the Netflix dataset (100-million ratings). Therefore, it will be much more expensive for preference ranking in the real world applications.

In order to speed up recommendations, some technologies have been proposed for efficient recommendation [6] via encoding real-valued vectors into compact binary codes. 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 [7], [8]. The fast hamming ranking process has been evaluated in PPH [5].

Due to the discrete constraints challenge, the learning of the compact binary codes is generally NP-hard [9], but can resort to a two-stage procedure [5], [10], [11], which consists of relaxed optimization via discarding the discrete constraints, and subsequent binary quantization. The aim of relaxed optimization is to learn real latent factors by stochastic gradient descent or alternative least square, and binary quantization process is to quantize real latent factors in a binary way. One representative work was proposed by K. Zhou et al. [11], where latent factors are rotated and quantized into hash code based on iterative threshold quantization. This method has been shown its potential in speeding up recommendation greatly, but suffers from low recommendation accuracy, compared with the traditional MF based models.

The deterioration of recommendation performance not only depends on information loss arising from hashing itself, but also lies in not considering the change of dot product caused by binary quantization, as analyzed in PPH that proposed a preference preserving hashing model. In the relaxed optimization process, PPH added the constant norm constaint for all real latent factors of users and items. In the quantization process, PPH offers a method transferring real latent factors to binary codes based on similarity and norm, respectively, i.e., each real latent factor is quantized by a hash code stacked by two parts directly: norm quantization and similarity quantization, on the basis of linear approximation. However, the assumption of linear approximation will suffer large information loss when the number of norm quantization bits is large. Besides, most bits of each code come from similarity quantization. Norms were quantized by only two bits in order to compensate the accuracy loss caused by similarity quantization. Only 2-bit of norm based hash codes will also lead to information loss.

As preferences represented by dot-products of the real latent factors of users and items in MF can achieve high recommendation accuracy. Intuitively, hash codes that preserve the dot-products ranking will obtain high recommendation accuracy. Therefore, in this paper, we devote to design hash codes that can preserve dot-products of users and items. Previous two-stage hashing methods are not dot-products preserving, which lead to large information loss. Besides, the accuracy of hashing recommendation is generally influenced by the bits of hash codes. However, there exist very limited studies on bit optimization for norm quantization. So in this paper we propose a method to search the local optimal bit for binary quantization.

To this end, we propose a dot-product based preserved preference method that keeps user preferences ranking over items in this paper. The rest of this paper is organized as follows: we first introduce the real latent factors for users and items based on L_2 regularized MF model [4] but without the constant norm constraint considered in PPH, which will be addressed in section II. Then we will discuss the quantization method in section III. Specifically, we will introduce the preference prediction model in Hamming space, subsequently with norm quantization and similarity quantization. In order to minimize the information loss caused by binary quantization, we further propose a model to optimize the bits of norm quantization in section III. Finally, we choose NDCG@K as the evaluation metric of recommendation accuracy, which is widely utilized in evaluating ranking task performance [12]. Experimental results on 4 open datasets: IMDB-100K, IMDB-1M, MovieLens-1M, MovieLens-10M, indicate that the recommendation accuracy of our proposed scheme is consistent superiority to the competing baselines even though only using shorter binary code. Experiments in this paper are conducted on a PC with Intel 64 bit CPU and 8G memory. So we don't evaluate our algorithm on larger datasets since the limit of memory. In section V, we summarize the main contributions and the experimental results of this paper.

II. Preliminary

We will introduce some notations related to this paper in this section. All vectors in this paper represent column vectors. Uppercase and lowercase bold letters denote matrices and vectors, respectively. Non-bold letters represent scalars. We first introduce some notations closely related to this paper. We then introduce the most popular MF model, MF-L_2.

A. Notations

In recommendation, suppose user and item sets are denoted by $U = \{1, \cdots, n\}$ and $I = \{1 \cdots, m\}$ respectively. An $n \times m$ matrix $\mathbf{R}$ represents the known ratings rated by n users, specifically, each element of $\mathbf{R}$, r_{ui} represents the rating of user u to item i. Generally, if user u has rated item i, r_{ui} is a positive value in a range. Otherwise, r_{ui} is equal to 0. In this paper, we assume that r_{ui} is within [0,5].

Model based collaborative filtering aims to provide or suggest interesting items to users according to users' past ratings. In this paper, we assume that rating r_{ui} represents the preference of user u to item i (i.e., if r_{ui} is greater than r_{uj}, then item i is more popular than j for user u). The goal of recommendation is to search the top-K popular items for each user. Therefore, the first step is to predict the ratings of items that are not rated by users in the past.

B. MF-L_2

Some successful realizations of latent factor models are based on matrix factorization (MF). In MF, users' preferences over items are modeled as dot products of real latent factors. Let the real latent factor of user u be represented by $\mathbf{p}_u \in \mathbb{R}^b$, the latent factor of item i be expressed as $\mathbf{q}_i \in \mathbb{R}^b$. Then the preference of u over i is measured by

$$\hat{r_{ui}} = \mathbf{p}_u^T \mathbf{q}_i. \tag{1}$$

In order to obtain latent features of items and users by known ratings. MF models learn real latent factors by minimizing the square information loss between known ratings and the estimated ratings. In order to avoid overfitting, two L_2-norm regularization terms are added in the objective (2) by Y. Koren, etc. [4], which was denoted by MF-L_2 model.

$$\begin{aligned}\mathbf{P}^*, \mathbf{Q}^* = &\arg\min_{\mathbf{p}_u, \mathbf{q}_i} \sum_{u=1}^{n}\sum_{i=1}^{m} I_{ui}(r_{ui} - \mathbf{p}_u^T\mathbf{q}_i)^2 + \\ &\lambda(\sum_{u=1}^{n}\|\mathbf{p}_u\|^2 + \sum_{i=1}^{m}\|\mathbf{q}_i\|^2),\end{aligned} \tag{2}$$

where $\mathbf{P}^* = (\mathbf{p_1}, \mathbf{p_2}, \cdots, \mathbf{p_n})^T$ and $\mathbf{Q}^* = (\mathbf{q_1}, \mathbf{q_2}, \cdots, \mathbf{q_m})$ are the learned real latent factors of users U and items I respectively. $\mathbf{R}$ is the known ratings. If user u has rated to item i in $\mathbf{R}$, $I_{ui} = 1$. Otherwise, $I_{ui} = 0$. λ is a regularizer parameter.

In general, $\mathbf{R}$ is a high sparsity matrix. This model utilizes the observed ratings $\mathbf{R}$ to predict the unknown ratings by formula (1). The above MF-L_2 model decomposes $\mathbf{R}$ into an $n \times b$ matrix $\mathbf{P}^*$ and a $b \times m$ matrix $\mathbf{Q}^*$ approximately. The aim of this model is to obtain optimal real latent factors by minimizing the objective (2). Due to the objective function is convex with respect to $\mathbf{P}^*$ and $\mathbf{Q}^*$ separately, $\mathbf{P}^*$ and $\mathbf{Q}^*$ can be easily obtained by stochastic gradient descent method.

III. Dot-Product Based Preference Preserved Hashing

As users' preference over items can be well approximated by dot-products between user-item real latent factors. Therefore, in this section we mainly introduce our quantization method that can preserve dot-products of users and items. Previous hashing recommendation was mainly based on similarity other than dot-product, which lead to much information loss due to ignoring the impact of norms in inner product. This section will provide our proposal for preference prediction in Hamming space, subsequently with quantization method consisting of similarity quantization and norm quantization. Our quantization technique can be regarded as the quantization of dot-product. At last, in order to minimize the information loss caused by quantization procedure, we propose a model to learn the local optimal bit for norm quantization.

A. Preference Model

From section II, we have obtained the real latent factors of users and items by MF-L_2. In this section, we propose a new hashing preference prediction model that preserve dot-products between users' and items' real latent factors. We quantize each real latent factor into two binary codes: norm based code and similarity based code. Because the dot-product between two vectors is composed of two norms and the cosine similarity. Accordingly, in this paper dot-product between two vectors in real latent space is approximated by the product of similarities of the two groups of codes (similarity codes and norm codes), respectively. Specifically, we assume that the norm and similarity based codes of user u and item i respectively are denoted by $\mathbf{p}_u^n,\ \mathbf{p}_u^s$ and $\mathbf{q}_i^n,\ \mathbf{q}_i^s$. The dot-product can be approximated by

$$\mathbf{p}_u^T\mathbf{q}_i \approx \mathrm{sim}\left(\mathbf{p}_u^n,\ \mathbf{q}_i^n\right)\cdot \mathrm{sim}\left(\mathbf{p}_u^s,\ \mathbf{q}_i^s\right).$$

Combined with the preference prediction model (1) discussed in section II, we can obtain the following preference prediction model in Hamming space,

$$\widehat{r_{ui}}^h = \mathrm{sim}\left(\mathbf{p}_u^n,\ \mathbf{q}_i^n\right)\cdot \mathrm{sim}\left(\mathbf{p}_u^s,\ \mathbf{q}_i^s\right). \qquad (3)$$

Preferences are estimated by the above equation in this paper and hash code of each real latent factor composed of two parts: similarity code and norm code.

Taking norm quantization as an example, we are interested in mapping users and items into k-bits binary codes for fast recommendation, where user's preference can be efficiently calculated via Hamming distance in the k-dimensional Hamming space. Let $\mathbf{p}_u^n\left\{\pm1\right\}^k$ and $\mathbf{q}_i^n \in \left\{\pm1\right\}^k$ represent the norm code of user u and the norm code of item i respectively. We stack them columnly into matrix $\mathbf{P}^n \in \left\{\pm1\right\}^{n\times k}$ and $\mathbf{Q}^n \in \left\{\pm1\right\}^{k\times m}$, respectively. The user-item similarity [11] can be defined as:

$$\mathrm{sim}(u,i) = \frac{1}{k}\sum_{r=1}^{k}\mathbb{I}\left(\mathbf{p}_u^n(r) = \mathbf{q}_i^n(r)\right) = \frac{1}{2} + \frac{1}{2k}\mathbf{p}_u^{nT}\mathbf{q}_i^n$$

If $\mathrm{sim}\left(u,i\right) > \mathrm{sim}\left(u,j\right)$, the user u prefers to item i instead of j. Assume user-item cosine similarity can be mapped into b-dimensional Hamming space. Substituting the above equation into the preference prediction model (3), we get

$$\begin{aligned}\widehat{r_{ui}}^h &= sim\left(\mathbf{p}_u^n,\ \mathbf{q}_i^n\right)\cdot sim\left(\mathbf{p}_u^s,\ \mathbf{q}_i^s\right)\\ &= \frac{1}{4}(1+\frac{1}{k}\mathbf{p}_u^{nT}\ \mathbf{q}_i^n)\cdot(1+\frac{1}{b}\mathbf{p}_u^{sT}\ \mathbf{q}_i^s).\end{aligned} \qquad (4)$$

Norm and similarity hash codes will be obtained by quantization methods introduced in the following subsections.

B. Similarity Quantization

In this subsection we mainly focus on similarity quantization. Suppose the similarity hash codes of $\mathbf{P}^*$ and $\mathbf{Q}^*$ are represented by $\mathbf{P^s}$ and $\mathbf{Q^s}$ respectively, where $\mathbf{P^s} \in \left\{-1,1\right\}^{n\times b}$ and $\mathbf{Q^s} \in \left\{-1,1\right\}^{b\times m}$, $\mathbf{p_u}^s \in \mathbf{P^s}$ and $\mathbf{q_i}^s \in \mathbf{Q^s}$.

As discussed above, preference based on dot-product consists of two parts: cosine similarity and norms. Intuitively, if we can design ranking preserving hash mapping based on cosine similarity and norms separately, then the items' ranking in Hamming space will be consistent with that in real space. We define the ranking preserved hash mapping as follows:

We assume that a hash mapping $H : \mathbb{R} \to \left\{-1,1\right\}^Y$ which maps elements of $\mathbb{R}$ to elements of $\left\{-1,1\right\}^Y$. For $a,\ b \in \mathbb{R}$, the corresponding images are $\mathbf{a^h},\ \mathbf{b^h} \in \left\{-1,1\right\}^Y$ respectively. We define the probe vector in $\left\{-1,1\right\}^Y$ denoted by $\mathbf{pr} = \left[1,1,\cdots,1\right]_Y{}^T$. If $a \geq b$, the corresponding similarities between $\mathbf{a^h},\ \mathbf{b^h},\ \mathbf{pr}$ satisfy $sim(\mathbf{a^h},\mathbf{pr}) \geq sim(\mathbf{b^h},\mathbf{pr})$, then H is called as ranking preserving mapping.

Based on the above consideration, for each user u we define H_u^s as the similarity based hash mapping. The probe vector is $\mathbf{p_u^s} = \left[1,1,\cdots,1\right]_b{}^T$, which is regarded as the similarity based hash codes of $\mathbf{p_u}$. H_u^s can be expressed as

$$\begin{aligned}&H_u^s : \mathbb{R} \to \left\{-1,1\right\}^b,\\ &\cos\theta_{\mathbf{p}_u,\mathbf{q}_i} \to \mathbf{q}_i^s = \left[\mathbf{q}_i^s(1),\mathbf{q}_i^s(2),\cdots,\mathbf{q}_i^s(b)\right]^T,\end{aligned} \qquad (5)$$

where

$$\mathbf{q}_i^s(m) = \begin{cases} 1\ , & if\ sgn(\mathbf{p}_u(m)) = sgn(\mathbf{q}_i(m)),\\ -1\ , & else,\end{cases}$$

and $m \in \{1,2,\cdots,b\}$.

Through the above similarity based hash mapping, we can get similarity based hash codes of items and users with low time complexity. Since we only need to compare the signs of real latent factors, such hash mapping process is high efficient. Furthermore, the probe vector will search the most similar vectors to $\mathbf{p}_u$, since the probe vector $\mathbf{p}_u^s$ can be regarded as an evaluation metric of similarities between the hash codes of users and items. If $sim(\mathbf{p}_u^s,\mathbf{q}_i^s) \geq sim(\mathbf{p}_u^s,\mathbf{q}_j^s)$, then $\mathbf{q}_i^s$ is more similar to $\mathbf{p}_u^s$ compared with $\mathbf{q}_j^s$. It can also be regarded as an evaluation metric of cosine similarity in real latent space. If $sim(\mathbf{p}_u^s,\mathbf{q}_i^s) \geq sim(\mathbf{p}_u^s,\mathbf{q}_j^s)$, then $\mathbf{q}_i$ has more same sign elements with $\mathbf{p}_u$ compared to $\mathbf{q}_j$. Therefore, the angle between $\mathbf{p}_u$ and $\mathbf{q}_i$ is smaller than the angle between $\mathbf{p}_u$ and $\mathbf{q}_j$. We can further conclude $\cos\theta_{\mathbf{p}_u,\mathbf{q}_i} \geq \cos\theta_{\mathbf{p}_u,\mathbf{q}_j}$, and vice versa.

It can be validated that H_u^s is ranking preserving. To keep the paper reasonably concise, the validation process is not mentioned in this paper.

C. Norm Quantization

In this subsection we will propose a new norm quantization method to generate hash codes from norms of real latent factors, that can improve recommendation accuracy to a great extent.

As discussed in the above subsection, in order to preserve items' ranking, we need to design ranking preserving hash mapping based on norms. We define H_u^n as the norm based hash mapping for user u. The probe vector is $\mathbf{p}_u^n = {[1, 1, \cdots, 1]_k}^T$, which can be also looked as the norm hash codes for $\mathbf{p}_u$. H_u^n can be expressed as

$$\begin{aligned} &H_u^n : \mathbb{R} \rightarrow \{-1, 1\}^k, \\ &\quad \|\mathbf{q}_i\| \rightarrow \mathbf{q}_i^n = [\mathbf{q}_i^n(1), \mathbf{q}_i^n(2), \cdots, \mathbf{q}_i^n(k)]^T, \end{aligned} \tag{6}$$

where

$$\mathbf{q}_i^n(t) = \begin{cases} 1 \quad , & if \, \|\mathbf{q}_i\| \geq \dfrac{10t-5}{2k}, \\ -1, & else, \end{cases}$$

$t \in \{1, 2, \cdots, k\}$, k is the bit of norm quantization. The value of k will be determined in the following subsection. Our method is designed under an assumption that all norms are within [0,5]. It is an appropriate assumption because the norms of $\mathbf{p}_u$ and $\mathbf{q}_i$ are restricted by L_2-norm regularization terms in (2). Therefore, most of the learned latent factors, $\mathbf{p}_u$ and $\mathbf{q}_i$, have small norms. Besides, for norms with more than 5, the method is also effective. Norms with greater than 5 will be transferred to ${[1, 1, \cdots, 1]_k}^T$ by our quantization method.

Through the above norm based hash mapping, we can obtain $k-$bit norm based hash codes that can achieve higher accuracy than 2-bit norm based codes in PPH, since k-bit quantization is more refined. Besides, the probe vector $\mathbf{p}_u^n$ can be used to search the most similar items to user u.

We can validate that the above norm based hash mapping is ranking preserving. For each user u and any two items i and j, the corresponding real latent factors are $\mathbf{p}_u$, $\mathbf{q}_i$ and $\mathbf{q}_j$ respectively. If $\|\mathbf{q}_i\| \geq \|\mathbf{q}_j\|$. From (5) and (6), we can attain that $\|\mathbf{q}_i^n\|$ has more bits with 1 than $\left\|\mathbf{q}_j^n\right\|$. Therefore,

$$sim(\mathbf{p}_u^n, \mathbf{q}_i^n) \geq sim(\mathbf{p}_u^n, \mathbf{q}_j^n).$$

Hence, the norm based mapping is ranking preserving.

For example, for a particular user u and two items i and j, let the corresponding real latent factors be $\mathbf{p}_u$, $\mathbf{q}_i$ and $\mathbf{q}_j$ respectively, $\|\mathbf{p}_u\| = 2.6$, $\|\mathbf{q}_i\| = 2.2$ and $\|\mathbf{q}_j\| = 3.7$. Suppose $k = 5$ (which will be determined in the next subsection), then we can calculate the norm based hash codes listed below by (6),

$$\begin{aligned} \mathbf{p}_u^n &= [1, \ 1, \ 1, \ 1, \ 1], \\ \mathbf{q}_i^n &= [1, \ 1, \ 1, \ -1, \ -1], \\ \mathbf{q}_j^n &= [1, \ 1, \ 1, \ 1, \ -1]. \end{aligned}$$

The norms $\|\mathbf{q}_i\| \leq \|\mathbf{q}_j\|$, the similarity between the corresponding hash codes satisfy

$$sim(\mathbf{p}_u^n, \mathbf{q}_i^n) \leq sim(\mathbf{p}_u^n, \mathbf{q}_j^n).$$

Therefore, the above norm based hashing mapping is ranking preserving.

D. Learning the Local Optimal Bit

The similarity codes usually lead to significant accuracy loss. In order to offset the loss, we proposed the norm quantization method in the above subsection. In this subsection, we will discuss how many bits of norm quantization are appropriate for given dimension b. As a solution, we propose a model to learn the local optimal bit by minimizing information loss caused by quantization process. To our knowledge, it is the first research effort to learn the local optimal bit for norm quantization. Substituting the learned local optimal bit k into norm quantization procedure in the above subsection, the information loss can be reduced significantly.

As discussed above, similarity based hashing mapping and norm based hashing mapping are ranking preserving, separately. Hash codes in this paper consists of norm codes and similarity codes. We can estimate users' preference in Hamming space according to (4), the preference is within $[0, 1]$. While the predicted ratings in real space calculated by (1) are in the range of $[0, 5]$. For data consistency, we need to normalize the predicted ratings in real latent space into $[0, 1]$. In order to minimize the information loss caused by hashing quantization, we learn the local optimal k for norm quantization by minimizing the following information loss.

$$Loss(k) = (\frac{1}{5}\mathbf{p}_u^T\mathbf{q}_i - \frac{1}{4}(1 + \frac{1}{k}\mathbf{p}_u^{nT} \ \mathbf{q}_i^n) \cdot (1 + \frac{1}{b}\mathbf{p}_u^{sT} \ \mathbf{q}_i^s))^2.$$

We can get the local optimal k by iterative approach, but it will lead to poor generalization due to overfitting. In order to avoid overfitting, we add a regularization term to penalize large k. Besides, the constraint of k can also reduce the complexity of norm quantization. Therefore, we get the following objective problem.

$$\begin{aligned} \arg\min_k \sum_{u=1}^{n}\sum_{i=1}^{m}\Big(\frac{1}{5}\mathbf{p}_u^T\mathbf{q}_i - \frac{1}{4}(1+\frac{1}{k}\mathbf{p}_u^{nT} \ \mathbf{q}_i^n)\cdot(1+\frac{1}{b}\mathbf{p}_u^{sT} \ \mathbf{q}_i^s)\Big)^2 \\ + \beta k. \end{aligned} \tag{7}$$

where β is a constant selected by cross-validation which weighs generalization and overfitting. The above problem can be solved by Algorithm 1, where k_{max} is the number of iterations, we set $k_{max} = 40$, since larger k will lead to higher quantization complexity.

Algorithm 1 Learning the Local Optimal Bit: k

```
1: Input:$\mathbf{P}^*$, $\mathbf{Q}^*$, $b$, $\beta$. ($\mathbf{P}^*$, $\mathbf{Q}^*$, $b$ can be obtained by (2)).
2:     for $k = 1$ to $k_{max}$
3:         Compute the objective of (7).
4:     end
5:         $k$ is the value that reach the minimum of (7).
6: Output $k$.
```

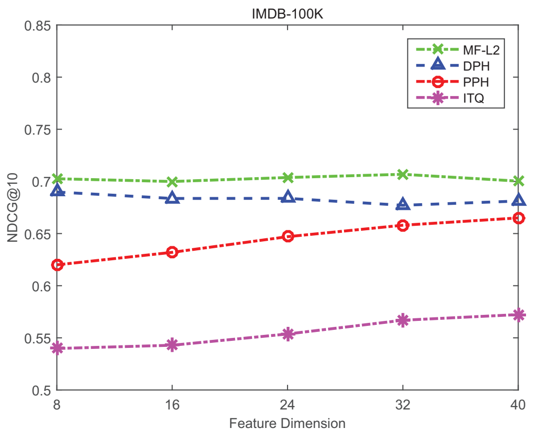

Fig. 1. Recommendation performance(NDCG@10) on IMDB-100k

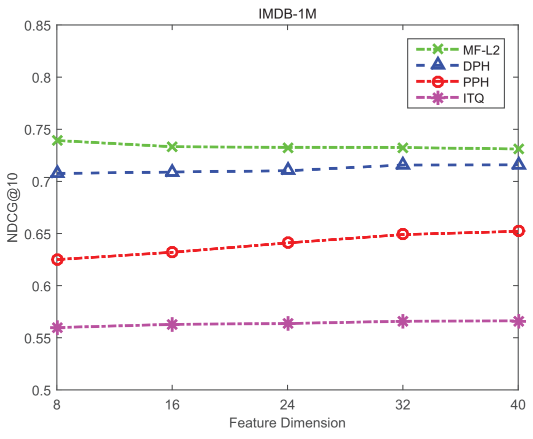

Fig. 2. Recommendation performance(NDCG@10) on IMDB-1M

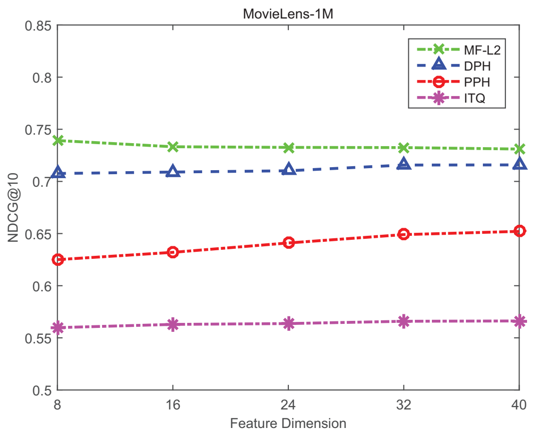

Fig. 3. Recommendation performance(NDCG@10) on MovieLens-1M

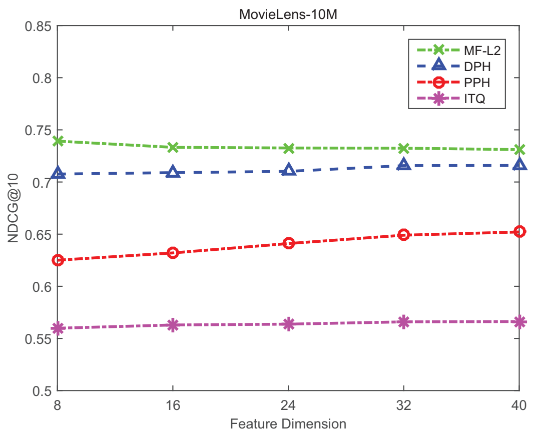

Fig. 4. Recommendation performance(NDCG@10) on MovieLens-10M

IV. Experiments

In this section, we mainly introduce our experiment settings and analyze some results. Experiments on several real world datasets show that our quantization scheme can achieve much higher recommendation accuracy than previous quantization methods.

A. Datasets and Evaluation Metric

We use two MovieLens datasets and two IMDB datasets [13]. 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. IMDB-100K ratings come from 1000 users on 1700 movies. IMDB-1M ratings come from 6000 users on 4000 movies.

We choose NDCG@K to evaluate the recommendation accuracy, which has been widely used to evaluate the performance of predicted ranking. As a matter of fact, recommendation can also be regarded as items' ranking for each user. In our experiments, we predict the top-10 popular items for each user. Therefore, we evaluate the recommendation accuracy by NDCG@10.

B. Experiment Settings

In our experiments, we divide each of the above 4 rating datasets R into 3 disjoint subsets randomly: training dataset, validation dataset and testing dataset. Training datasets are composed of 20 ratings of each user. Validation datasets include 10 ratings of each user. The rest at least 10 ratings of each user make up the testing datasets, since we need to predict the top-10 popular items for each user. Thus, users having less than 40 ratings are not considered in our experiments.

Training datasets and validation datasets are utilized to learn the real latent factors by MF-L_2 (2). Validation datasets are used to select the regularization parameter λ and β. The datasets split is randomly repeated 5 times to report the average results. We tune parameter λ in $[1, 100]$. Experiments on these 4 validation datasets show that the recommendation accuracy is quite stable within a large range of $\lambda \in [8, 30]$. So in this paper, we choose λ=10 for all datasets.

In the process of learning the local optimal bit k for norm quantization, similarly, we divide the learned real latent factors $\mathbf{P}^*$ and $\mathbf{Q}^*$ into 3 disjoint datasets respectively: training dataset, validation dataset and testing dataset. Experiments on validation datasets show that the hyper-parameter β varies with b. After selecting β for various b and λ=10, we test the performance of our algorithm on the 4 testing datasts that is shown in Figure 1-4.

C. Experimental Results

Figure 1-4 show the performances (NDCG@10) of DPH (method in this paper) on 4 datasets respectively are superior to the state-of-the-art two-stage hashing methods. ITQ were implemented by K. Zhou [11]and PPH was proposed by Zhang et al. [5]. Moreover, we conclude that the recommendation accuracy of DPH is very close to MF-L_2. We make the following observations from the results:

Compared with other hash quantization methods, our method has much higher recommendation accuracy. Other hash codes are mainly generated from similarity quantization. However, similarities between real latent factors of users and items can not be well approximated as preferences. The proposed model outperforms previous methods mainly because it quantizes both norms and cosine similarity in real latent space. The norm quantization process can compensate the accuracy loss caused by similarity quantization. Furtherly, the process of learning the optimal bit can minimize the information loss for given b.

Compared with MF-L_2, the accuracy of our method is very close to the classic MF-L_2 model. Due to hash codes can be used in fast online recommendation by hashing index technology, hashing recommendation is more popular than MF-L_2.

Based on Algorithm 1, we can get the optimal bits k for different dimensions b. Figure 5 illustrates the optimal k varies with b on IMDB-1M. We can conclude: When b is small, similarity quantization brings much information loss. The optimal k increases faster than b for compensating information loss. As b rises up, information loss caused by similarity quantization is reduced, meanwhile, the complexity of quantization is increased. Therefore, the optimal k increases slower than b. As $b \geq 25$, fewer information loss and higher quantization complexity impel the optimal k to be stabilized gradually. Similarity quantization plays a dominant role in recommendation.

V. Conclusion

In this paper, we have proposed a new hashing framework that can preserve the dot-products. Accordingly preferences ranking is also be preserved. Quantization method in this paper consists of similarity quantization and norm quantization. The norm quantization can compensate for similarity quantization significantly. Furthermore, we develop a model to learn the optimal bit for norm quantization that can minimize the information loss caused by the entire hash codes. Experimental results have shown that our hashing recommendation method is superior to other hashing methods on four real world datasets.

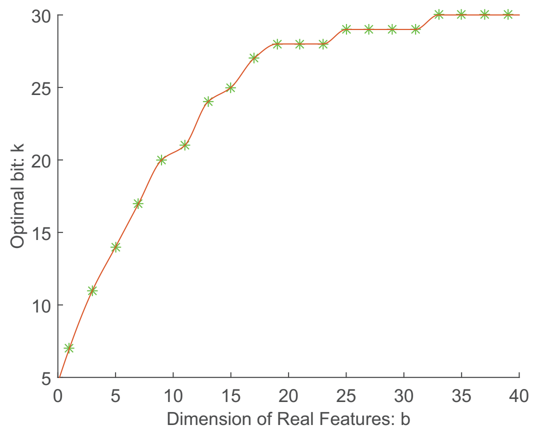

Fig. 5. Optimal k varies with b on IMDB-1M

In addition, we get a rule based on amounts of experiments: When the dimension b of real latent factors is small, the norm quantization is more important than that when b is large, since it can offset the information loss caused by similarity quantization to a great extent. When the dimension of real features rising up, similarity quantization plays a dominant role in recommendation.

Acknowledgements

This work is supported by the National Natural Science Foundation of China (No. 61572109) and the 863 High Technology Plan (Grant No. 2015AA01A707).

References

[1] Q. Zhao, Y. Zhang, D. Friedman, and F. Tan, "E-commerce recommendation with personalized promotion," in *Proc. ACM RecSys*, Sep. 2015, pp. 219–226.
[2] F. E. Walter, S. Battiston, and F. Schweitzer, "A model of a trust-based recommendation system on a social network," *Autonomous Agents and Multi-Agent Systems*, vol. 16, no. 1, pp. 57–74, Feb. 2008.
[3] H. Tan, J. Guo, and Y. Li, "E-learning recommendation system," in *Proc. IEEE CSSE*, vol. 5, Dec. 2008, pp. 430–433.
[4] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, no. 8, pp. 30–37, Aug. 2009.
[5] Z. Zhang, Q. Wang, L. Ruan, and L. Si, "Preference preserving hashing for efficient recommendation," in *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*. ACM, 2014, pp. 183–192.
[6] J. Wang, H. T. Shen, J. Song, and J. Ji, "Hashing for similarity search: A survey," *arXiv preprint arXiv:1408.2927*, 2014.
[7] J. Wang, S. Kumar, and S.-F. Chang, "Semi-supervised hashing for large-scale search," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 34, no. 12, pp. 2393–2406, Dec. 2012.
[8] M. Muja and D. G. Lowe, "Fast approximate nearest neighbors with automatic algorithm configuration," in *Proc. VISAPP*, Feb. 2009, pp. 331–340.
[9] J. Håstad, "Some optimal inapproximability results," *Journal of the ACM (JACM)*, vol. 48, no. 4, pp. 798–859, Jul. 2001.
[10] X. Liu, J. He, C. Deng, and B. Lang, "Collaborative hashing," in *Proc. IEEE CVPR*, Jun. 2014, pp. 2139–2146.
[11] K. Zhou and H. Zha, "Learning binary codes for collaborative filtering," in *Proc. ACM SIGKDD*, Aug. 2012, pp. 498–506.
[12] M. Volkovs and R. S. Zemel, "Collaborative ranking with 17 parameters," in *Proc. NIPS*, Dec. 2012, pp. 2294–2302.
[13] H. Shan and A. Banerjee, "Generalized probabilistic matrix factorizations for collaborative filtering," in *Proc. IEEE ICDM*, Dec. 2010, pp. 1025–1030.