HoORaYs: High-order Optimization of Rating Distance for Recommender Systems

Jingwei Xu  
State Key Laboratory for Novel Software Technology  
China 210023  
jingwei.xu@smail.nju.edu.cn

Yuan Yao  
State Key Laboratory for Novel Software Technology  
China 210023  
y.yao@nju.edu.cn

Hanghang Tong  
Arizona State University  
Phoenix, Arizona  
hanghang.tong@asu.edu

Xianping Tao  
State Key Laboratory for Novel Software Technology  
China 210023  
txp@nju.edu.cn

Jian Lu  
State Key Laboratory for Novel Software Technology  
China 210023  
lij@nju.edu.cn

ABSTRACT

Latent factor models have become a prevalent method in recommender systems, to predict users’ preference on items based on the historical user feedback. Most of the existing methods, explicitly or implicitly, are built upon the first-order rating distance principle, which aims to minimize the difference between the estimated and real ratings. In this paper, we generalize such first-order rating distance principle and propose a new latent factor model (HoORaYs) for recommender systems. The core idea of the proposed method is to explore high-order rating distance, which aims to minimize not only (i) the difference between the estimated and real ratings of the same (user, item) pair (i.e., the first-order rating distance), but also (ii) the difference between the estimated and real rating difference of the same user across different items (i.e., the second-order rating distance). We formulate it as a regularized optimization problem, and propose an effective and scalable algorithm to solve it. Our analysis from the geometry and Bayesian perspectives indicate that by exploring the high-order rating distance, it helps to reduce the variance of the estimator, which in turns leads to better generalization performance (e.g., smaller prediction error). We evaluate the proposed method on four real-world data sets, two with explicit user feedback and the other two with implicit user feedback. Experimental results show that the proposed method consistently outperforms the state-of-the-art methods in terms of the prediction accuracy.

CCS CONCEPTS

•Information systems →Information systems applications;  
Collaborative filtering; Data mining;

KEYWORDS

High-order distance, recommender systems, latent factor model, collaborative filtering

1 INTRODUCTION

In recent years, researchers have devoted great efforts to the development of recommender systems in many real-world applications [4, 6, 11]. The key task of recommender systems is to predict the users’ preference on items. Collaborative filtering (CF) methods and content-based methods have been widely used to achieve this task. For example, matrix factorization [11] takes ratings as input and outputs the latent vectors for users and items; it becomes a popular base for recommender systems, largely due to its great success at the Netflix Prize. To further improve the recommendation accuracy, Wang et al. [20] propose the collaborative topic regression (CTR) rating model to incorporate item content; Ma et al. [12] model social trust (Sorec) by incorporating social relationships and ratings. The combination of CTR and Sorec is also explored [2, 17].

A line of existing work has focused on employing different types of data (e.g., ratings, item content, social relationships, etc.) so as to make more informed and accurate recommendations. In this work, we focus on an orthogonal line work, i.e., the optimization formulation aspect.

From the optimization viewpoint, most of the existing methods, explicitly or implicitly, are built upon the first-order rating distance principle. That is, these methods seek for an ‘optimal’ latent representations for users and items, which minimize the differences between the estimated and real ratings of the same (user, item) pair. Conceptually, minimizing the first-order distance (between the real rating to estimated rating) can be viewed as a self-calibration process. However, the solution space of the optimization problem could be large, especially when the available user feedback information is sparse, which might result in a biased estimator for the latent vectors of users and items.

In this paper, we generalize the first-order distance principle and propose to leverage the high-order distance to improve the recommendation performance. The core idea of the proposed method is to explore high-order rating distance, which aims to minimize not only (i) the difference between the estimated and real ratings of the
same (user, item) pair (i.e., the first-order rating distance), but also (ii) the difference between the estimated and real rating difference of the same user across different items (i.e., the second-order rating distance). We hypothesize that by exploring high-order distance, it will help shrink the solution space of the corresponding optimization problem. By doing so, the variance of the estimator (i.e., the latent representations of users and items) could be mitigated, which will in turn lead to better generalization performance (e.g., a smaller prediction error).

The main contributions of this paper include:

- **New Model and Algorithm** that embrace the high-order rating distance in the latent factor methods for recommender systems. The proposed model HoORaYs can handle both explicit and implicit user feedback, as well as the case when content information is available. The proposed algorithm is able to find local optima with a linear time complexity.
- **Analysis** from both the geometric perspective and the Bayesian perspective for the proposed model, which provides key insight on how the high-order distance reduces the variance, and how to generalize the high-order distance based optimization to other recommendation models.
- **Experimental evaluations** on four real-world data sets showing the effectiveness of the proposed method. For example, the proposed method outperforms the best competitors by up to 24.3% improvement in terms of the prediction accuracy.

The rest of the paper is organized as follows. In Section 2, we present the problem statement. In Section 3, we describe the proposed model with the geometric interpretation, Bayesian interpretation, and algorithm analysis. In Section 4, we present the experimental results. In Section 5, we review the related work. Finally, we conclude the paper in Section 6.

## 2 PROBLEM STATEMENT

In this section, we provide the problem statement and some background knowledge of our proposed model.

### 2.1 The Recommendation Problem

In recommender systems, the two kinds of fundamental elements are users and items. We assume there are $M$ users and $N$ items in the recommender system. We denote the latent vectors for users as $U = \{u_i\}_{i=1}^M$, and the latent vectors for items as $V = \{v_j\}_{j=1}^N$. The length of these latent vectors is $K$. The observed ratings are usually denoted as $R = \{r_{ij} | r_{ij} \in [1, r_{\text{max}}]\}$, where $r_{ij}$ represents the rating that user $i$ gives to item $j$, and the value $r_{\text{max}}$ is the scale of rating in the target recommender system (e.g., 5 stars on MovieLens).

Based on the above notations, we define the target problem as follows

**Problem 1. The Recommendation Problem**

Given: (1) the set of existing ratings $R$ from users to items, (2) a user $i$, and (3) an item $j$;

Find: the estimated rating $\hat{r}_{ij}$ from user $i$ to item $j$.

As we can see from the definition, the input of our problem includes the existing ratings. Our focus is on the optimization aspect instead of employing more types of data, although the proposed method can be similarly applied when more types data are available.

The goal is to predict the unobserved ratings from users to items, and we can directly obtain the ratings as long as we have learned the latent vectors $U$ and $V$.

### 2.2 Latent Factor Models: Matrix Factorization

In recent years, matrix factorization based collaborative filtering [11] becomes one of the most popular methods to solve the recommendation problem. In the view of matrix factorization, the users and items could be represented by factors in the same latent factor space. For example, user $i$ is represented by a latent factor vector $u_i$, and item $j$ is represented by a latent factor vector $v_j$. So we predict the rating that user $i$ gives to item $j$ with the inner product of the two corresponding latent factor vectors

$$
\hat{r}_{ij} = u_i^T v_j
$$

We use the observed ratings to learn the latent factor vectors. Commonly, we minimize the following optimization function

$$
\min_{U^*,V^*} \sum_{r_{ij} \in R} (r_{ij} - u_i^T v_j)^2
$$

where square loss is used as the loss function, and $R$ is the set of the observed ratings.

### 2.3 Model Variance Reduction

From statistical learning perspective, a good estimator (e.g., latent vectors in our recommendation problem) should have small prediction errors ($PE$) on both training and the new data. With the bias-variance decomposition [5], the expected prediction error is the sum of three terms: the irreducible errors, Bias, and Variance

$$
PE = \sigma^2 + \text{Bias}^2 + \text{Variance}
$$

It is well known that the local curvature can be picked up to fit the training data when the model becomes more complex. However, such a complex model suffers from the high variance, and hence to a high PE when estimating on the new data (overfitting). To deal with the overfitting in Eq. (1), researchers added ridge constraints (i.e., L2-regularization) on the parameters $u_i$ and $v_j$ by controlling their sum of squares, so that the original unconstrained optimization problem becomes

$$
\minimize \sum_{r_{ij} \in R} (r_{ij} - u_i^T v_j)^2 \text{ s.t. } \sum_{i=1}^M (u_i)^2 \leq t_u, \sum_{j=1}^N (v_j)^2 \leq t_v
$$

where $t_u > 0$ and $t_v > 0$. The above optimization problem can be re-written as

$$
\mathcal{L} = \sum_{r_{ij} \in R} (r_{ij} - u_i^T v_j)^2 + \lambda_u \sum_{i=1}^M ||u_i||^2 + \lambda_v \sum_{j=1}^N ||v_j||^2
$$

where $\lambda_u$ and $\lambda_v$ are the parameters to control the L2 regularization terms. With the proper $\lambda_u$ and $\lambda_v$, Eq. (5) balances bias and variance to reach the lower PE.
3 HIGH-ORDER OPTIMIZATION OF RATING DISTANCE

In this section, we describe the proposed HoORaYs model (Subsection 3.1). Then, we analyze why the proposed model can reduce the variance of the estimator from two different perspectives, including geometric interpretation (Subsection 3.2) and Bayesian interpretation (Subsection 3.3). Finally, we present a brief algorithm analysis in terms of its optimality and complexity (Subsection 3.4).

3.1 The HoORaYs Model

The core idea of HoORaYs is to use high-order rating distance to reduce the variance of the estimator (i.e., the latent factors for users and items) by shrinking the solution space of the optimization problem in Eq. (4). The intuition is as follows. By optimizing the first-order distance, we basically want to find a good estimator which matches a user’s preference on each of the observed items. By introducing the additional high-order distance, we require the learnt latent factors to also capture the subtle preference difference of a user across different item pairs.

In particular, the error between the real rating \( r_{ij} \) and the estimated rating \( \hat{r}_{ij} \) in Eq. (4) can be treated as the distance between two ratings. In addition the rating distance of \(< r_{ij}, \hat{r}_{ij} > \) pair, other rating errors from different kinds of pairs (e.g., \(< r_{ij}, \hat{r}_{ij} >, < r_{ij}, \hat{r}_{ij} \), and so on) can also be measured as rating distances. These rating distances have their own meaning in the context of recommender systems. For example, the distance between \( r_{ij} \) and \( r_{ij} \) (denoted by \( D \)) means the real difference between item \( j \) under user \( i \); the distance between \( r_{ij} \) and \( r_{ij} \) (denoted by \( D \)) means the estimated difference between item \( j \) and item \( j^\prime \) under user \( i \) after estimating \( \hat{r}_{ij} \). Furthermore, when \( r_{ij} \) is fixed for both \( D \) and \( \hat{D} \), we can measure the distance between \( D \) and \( \hat{D} \). The error between \( D \) and \( \hat{D} \) reflects the accuracy of learned latent vectors of user \( i \) and item \( j \). This error is the distance between two rating distances, and we call this distance of distance as second-order rating distance.

As we can see, both first-order and second-order distances reflect the learned latent vectors of user/items. If we add second-order rating distance to the optimization problem in Eq. (4) as an additional constraint, we could further reduce the variance of the latent vectors of users/items in recommender systems. The optimization problem of our proposed HoORaYs is written as below

\[
\begin{align*}
\text{minimize} & \quad \sum_{r_{ij} \in R} (r_{ij} - u^T_i v_j)^2 \\
\text{s.t.} & \quad \frac{m}{n}(u_i)^2 \leq t_u, \quad \frac{n}{m}(v_j)^2 \leq t_v \\
& \quad \sum_{r_{ij} \in R} l_{ij} (\sigma(u^T_i v_j, r_{ij}) - \sigma(r_{ij}, r_{ij}'))^2 = 0 
\end{align*}
\]

where \( \sigma(x, y) = 1/(1 + e^{-(x-y)}) \) and \( l_{ij} = 1 \) if ratings \( r_{ij} \) and \( r_{ij}' \) exist with \( i' = i \land j' \neq j \land i' 
eq i \land j' = j \). The optimization problem with second-order rating distance constraint can be re-written as follow

\[
\begin{align*}
\text{argmin} & \quad U, V \sum_{r_{ij} \in R} (r_{ij} - u^T_i v_j)^2 + \lambda_u \sum_{i=1}^m ||u_i||^2 + \lambda_v \sum_{j=1}^n ||v_j||^2 \\
& \quad + \lambda_d \sum_{r_{ij} \in R} l_{ij} (\sigma(u^T_i v_j, r_{ij}) - \sigma(r_{ij}, r_{ij}'))^2 
\end{align*}
\]

where \( \lambda_d \) is the parameter to control the effect of second-order rating distance.

**Speed Gear.** Since the complexity of second-order distance in Eq. (7) is roughly \( O(|R|(|R_u| + |R_v|)) \), we propose to speedup the model learning process based on two key observations: (i) the rating scale is usually small for recommender systems (e.g., 1-5 stars), and (ii) the contributions of different ratings from other users/items are equal to each other if they share the same rating value. Hence, the loss function in Eq. (7) could be re-written as

\[
\begin{align*}
\text{argmin} & \quad U, V \sum_{r_{ij} \in R} (r_{ij} - u^T_i v_j)^2 + \lambda_u \sum_{i=1}^m ||u_i||^2 + \lambda_v \sum_{j=1}^n ||v_j||^2 \\
& \quad + \lambda_d \sum_{r_{ij} \in R} \max_{r_{ij} \in R} ||\sigma(u^T_i v_j, r_{ij}) - \sigma(r_{ij}, r_{ij})||^2 
\end{align*}
\]

where \( \lambda_d \) is the maximal rating value (e.g., \( r_{max} = 5 \) in MovieLens), and \( ||\sigma(u^T_i v_j, r_{ij}) - \sigma(r_{ij}, r_{ij})|| \) is the total number of ratings that the user i rated to other items with value \( r \) or the item \( j \) received from other users with value \( r \).

**Update Rules.** We use stochastic gradient descent to optimize Eq. (8). In detail, we alternatively optimize \( U \) and \( V \) in each iteration. In each iteration, we take the partial derivatives of Eq. (8) with respect to \( u_i \) and \( v_j \), which lead to the following update rules

\[
\begin{align*}
u_i & \leftarrow -\lambda_u u_i + (r_{ij} - u^T_i v_j) \\
& + \lambda_d \sum_{r_{ij} \in R} \max_{r_{ij} \in R} ||\sigma(u^T_i v_j, r_{ij}) - \sigma(r_{ij}, r_{ij})||^2 \\
& - \sigma(u^T_i v_j, r) (1 - \sigma(u^T_v v_j, r)) v_j \\
v_j & \leftarrow -\lambda_v v_j + (r_{ij} - u^T_i v_j) u_i \\
& + \lambda_d \sum_{r_{ij} \in R} \max_{r_{ij} \in R} ||\sigma(u^T_i v_j, r_{ij}) - \sigma(r_{ij}, r_{ij})||^2 \\
& - \sigma(u^T_v v_j, r) (1 - \sigma(u^T_i v_j, r)) u_i 
\end{align*}
\]

We summarize our fast learning algorithm in Alg. 1, where \( \alpha \) is the learning rate, \( \nabla \) is the partial derivative operator, and we use stochastic gradient descent method to learn the parameters.

3.2 Geometric Interpretation of HoORaYs

Here we present the geometric interpretation of the proposed HoORaYs. By adding the constraint of second-order rating distance to the optimization problem in Eq. (4), the solution space to \( U \) and \( V \) can be further shrunk, which leads to the decrease on variance.

The illustrative example in Fig. 1 shows how the constraint of second-order rating distance could shrink the original solution space. Suppose that the latent factor space is 2-dimensional, and each user/item has its place, which is presented as the latent vector in the space. In Fig. 1(a), taking user \( i \) and item \( j \) for example
Figure 1: The illustrative example of applying the constraint of second-order rating distance to matrix factorization. We can see that second-order distance further shrinks the solution space, which means to reduce variance of prediction error for the optimization problem. Notice that, the second-order distance will also shrink the solution space on the other side, for clarity, we do not indicate that in the figure.

Algorithm 1: Learning HoORAYs

Input: the set of observed ratings \( R \), and the maximal rating scale \( r_{\text{max}} \)

Output: latent vectors of users \( U \), latent vectors of items \( V \)

1. Initialize \( U, \ V \)

2. \( \text{repeat} \)

3. \( \text{for } r_{ij} \in R \) \( \text{do} \)

4. \( \text{for } r = 1, \ldots, r_{\text{max}} \) \( \text{do} \)

5. \( \text{for } f = 1, \ldots, k \) \( \text{do} \)

6. Update \( u_{i,f} \leftarrow u_{i,f} - \alpha \nabla u_{i,f} \) as defined in Eq. (9)

7. Update \( v_{j,f} \leftarrow v_{j,f} - \alpha \nabla v_{j,f} \) as defined in Eq. (10)

8. \( \text{until convergence} \)

9. \( \text{return } U^*, \ V^* \)

and by assuming that \( v_j \) and \( r_{ij} \) are given, we want to find \( u_i \) by first-order rating distance (e.g., matrix factorization). Ideally, \( u_i^T v_j = r_{ij} \) exists for the perfect \( u_i \), and this line could be plotted in the space. As shown in Fig. 1(a), any point in this line is a solution to \( u_i \), and all points satisfy the best condition to \( r_{ij} \). Then, we allow the error to \( u_i \), e.g., \( |u_i^T v_j - r_{ij}| < \epsilon \). With the two paralleled lines \( u_i^T v_j = r_{ij} + \epsilon \) and \( u_i^T v_j = r_{ij} - \epsilon \) (two red lines in Fig. 1(a)), any point inside the two bounded lines is a solution that satisfies to the error \( \epsilon \). Now, we consider another item \( j' \) with the given \( v_{j'} \) into the space, and the solution space of \( u_i \) is then bounded by the four red lines in the Fig. 1(a). In Fig. 1(b), the black point is the origin of the latent factor space, and the length of perpendicular distance from origin to line \( u_i^T v_j = r_{ij} \) is \( r_{ij} \) when \( u_i \) is normalized. \( r_{ij'} \), denoted as a black line segment, can also measured in the space. After coinciding line segment \( r_{ij'} \) to line segment \( r_{ij} \), several second-order distances appear. In Fig. 1(b), the second-order distance between \( r_{ij} \) and \( r_{ij'} \) is denoted as \( \hat{D} \). Based on the two worst estimated \( \hat{r}_{ij} \) bounded by \( \epsilon \), we could point out two second-order distances between \( \hat{r}_{ij} \) and \( \hat{r}_{ij'} \) as \( \hat{D} \) in Fig. 1(b). By the constraint of \( |\hat{D} - D|^2 = 0 \), the error bounds would shrink from both sides. Ideally, as the optimization problem in Eq. (6) reaches the optima, the shaded area will be compressed by the constraint of second-order rating distance. Shrinking the solution space reduces the variance (e.g., almost half of original solution space is shrunk in Fig. 1(c)), and might reduce the prediction error \( PE \) of the rating model in Eq. (1).

3.3 Bayesian Interpretation of HoORAYs

In addition to the geometric interpretation, we present our proposed HoORAYs from Bayesian perspective. Minh et al. [14] presented a probabilistic model for matrix factorization. In probabilistic matrix factorization, they assumed that ratings are generated by a specific generative process. In order to leverage information from content (e.g., reviews, tags), researchers used topic modeling approaches to extract latent topics from items. Collaborative topic regression (CTR) model was proposed by Wang et al. [20] to deal with recommendation problem by considering the merit of both probabilistic topic modeling and collaborative filtering.

The proposed HoORAYs can apply to both MF and CTR. Take CTR model as the rating model, The graphical model of HoORAYs is shown in Fig. 2. We treat the constraint of second-order distance as an observed random variable after observing the rating. Next, instead of simply using the distance between the real \( r_{ij} \) and estimated \( \hat{r}_{ij} \) as the optimization target in CTR, we optimize over the second-order distance \( d_{ijfj'} \) which is the distance between the real rating distance (between \( r_{ij} \) and \( r_{ij'} \)) and the estimated rating distance (between \( \hat{r}_{ij} \) and \( \hat{r}_{ij'} \)). The generative process of HoORAYs is as follows:

- For each user \( i \), draw the latent vector \( u_i \sim N(0, \lambda_u^{-1} I_k) \)
- For each item \( j \)
  - Draw topic proportions \( \theta_j \sim \text{Dirichlet}(\alpha) \)
  - Draw the latent offset \( e_j \sim N(0, \lambda_e^{-1} I_k) \) and set the item latent vector as \( v_j = e_j + \theta_j \)
  - For each word \( w_{ijn} \),

---

**Figure 1:** The illustrative example of applying the constraint of second-order rating distance to matrix factorization. We can see that second-order distance further shrinks the solution space, which means to reduce variance of prediction error for the optimization problem. Notice that, the second-order distance will also shrink the solution space on the other side, for clarity, we do not indicate that in the figure.

**Algorithm 1:** Learning HoORAYs

1. **Input:** the set of observed ratings \( R \), and the maximal rating scale \( r_{\text{max}} \)
2. **Output:** latent vectors of users \( U \), latent vectors of items \( V \)
3. **initialize** \( U, V \)
4. **repeat**
5. **for** \( r_{ij} \in R \) **do**
6. **for** \( r = 1, \ldots, r_{\text{max}} \) **do**
7. **for** \( f = 1, \ldots, k \) **do**
8. update \( u_{i,f} \leftarrow u_{i,f} - \alpha \nabla u_{i,f} \) as defined in Eq. (9)
9. update \( v_{j,f} \leftarrow v_{j,f} - \alpha \nabla v_{j,f} \) as defined in Eq. (10)
10. **until** convergence
11. **return** \( U^*, V^* \)
Wang et al. [20] to solve the problem (Eq. (8)) is not convex wrt the coefficients (Eq. (11)). Therefore, we use stochastic gradient ascent to optimize Eq. (13). In detail, we iteratively optimize other parameters similarly to Eqs. (9) and (10). Given the current \( \theta \), we use projection gradient approach to optimize the other parameters \( U, V, \theta_{1N} \), and \( \phi_{1N} \). After estimating \( U, V, \phi, \theta \), we could optimize \( \theta_{1N} \) and apply Jensen’s inequality as follows:

\[
L(\theta_j) \geq -\frac{\lambda_U}{2}(v_j - \theta_j)^T(v_j - \theta_j) \\
+ \sum_{n_w=1}^{r_{max}} \sum_{r=1}^{m} \phi_{jn_{uw}k} \log(\theta_{jk} \theta_{nk} w_{jn_{uw}} - \log \phi_{jn_{uw}k}) \\
= L(\theta_j, \phi_{jn_{uw}k})
\]

Let \( \Phi_j = (\phi_{jn_{uw}k})_{n_w=1}^{r_{max}, k=1} \). \( L(\theta_j) \) has a tight lower bound \( L(\theta_j, \Phi_j) \). Analytically, we cannot optimize \( \theta_j \). Hence, we use projection gradient approach to optimize the other parameters \( U, V, \theta_{1N}, \Phi_{1N} \). After estimating \( U, V, \phi, \theta, \theta_{1N} \), and \( \phi_{1N} \), each rating \( r_{ij} \) can be estimated as

\[
r_{ij} \approx (u_i^T v_j)^* 
\]

### 3.4 Algorithm Analysis

In this part, we analyze the effectiveness and efficiency of our algorithms.

The effectiveness of the proposed HoORaYs is summarized in Lemma 3.1. Overall, it finds local optima in the solution space of the latent vectors from users and items. The proposed optimization problem (Eq. (8)) is not convex wrt the coefficients \( (u_i, v_j) \), and such a local minimum is acceptable in practice.

**Lemma 3.1 (Effectiveness of HoORaYs).** Fixing the ratings in \( R \), HoORaYs finds the local minimum for the optimization problem in Eq. (8).
Table 1: The statistics of the four data sets.

<table>
<thead>
<tr>
<th>Data</th>
<th>Lastfm</th>
<th>Delicious</th>
<th>MovieLens</th>
<th>Google Play</th>
</tr>
</thead>
<tbody>
<tr>
<td>#users</td>
<td>1,892</td>
<td>1,867</td>
<td>2,113</td>
<td>170,781</td>
</tr>
<tr>
<td>#items</td>
<td>17,632</td>
<td>69,226</td>
<td>10,197</td>
<td>104,061</td>
</tr>
<tr>
<td>#tags</td>
<td>11,946</td>
<td>53,388</td>
<td>13,222</td>
<td>N/A</td>
</tr>
<tr>
<td>#taggings</td>
<td>186,479</td>
<td>437,593</td>
<td>47,957</td>
<td>N/A</td>
</tr>
<tr>
<td>#words</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>10,569</td>
</tr>
<tr>
<td>#reviews</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>3,367,435</td>
</tr>
<tr>
<td>#ratings/hits</td>
<td>92,834</td>
<td>104,799</td>
<td>855,598</td>
<td>3,367,435</td>
</tr>
<tr>
<td>rating Scale</td>
<td>[0.5-5]</td>
<td>[1-5]</td>
<td>[0.5-5]</td>
<td>[1-5]</td>
</tr>
</tbody>
</table>

Proof Sketch. If we fix either the U matrix or the V matrix, the optimization problem becomes convex and the corresponding step in Alg. 1 can find the global optima. Next, based on the alternating procedure of learning parameters U and V in Alg. 1, we have that Alg. 1 finds a local minimum for the optimization problem in Eq. (8).

The time complexity of the proposed HoORAYs is summarized in Lemma 3.2. This Lemma shows that HoORAYs requires linear time for learning latent vectors of users and items (e.g., step 3-7 in Alg. 1); and it scales linearly wrt the number of observed ratings in the training phase (e.g., step 2-7 in Alg. 1).

LEMMA 3.2 (Time Complexity of HoORAYs). Fixing the set of ratings R, and $r_{\max}$, HoORAYs requires $O(|R|)$ time for each iteration in Alg. 1.

Proof. For each iteration in Alg. 1, we need $O(|R|)$ time for the loop that starts from step 3. The time cost for the loop that starts from step 4 is $r_{\max}$, and step 5-7 costs $O(k)$ time for updating parameters. Therefore, the total time cost of the iteration of step 2-7 is $O(|R| \cdot r_{\max} \cdot k \cdot m)$, where $m$ is the maximum iteration number for Alg. 1. Notice that, $r_{\max} \cdot k$ and $m$ are small constants, so the total time cost of Alg. 1 can be written as $O(|R|)$. □

4 EXPERIMENTS

In this section, we present the experimental evaluations. All the experiments help us to answer the following questions:

- How accurate is the proposed method compared to the state-of-the-art methods?
- How efficient is the proposed method compared to the state-of-the-art methods? How scalable is the proposed method?
- How do the parameters affect the performance of our model?

4.1 Experimental Setup

4.1.1 Data Sets. In this paper, we use four real-world data sets, i.e., Google Play, MovieLens, Lastfm, and Delicious. The first data set was collected by Chen et al. [3], and the other three data sets were provided by Cantador et al. [1]. Table 1 shows the statistics of the four data sets. For the Lastfm and Delicious data, the user feedback is implicitly given by listening to a song (on Lastfm) and bookmarking an item (on Delicious), respectively. Following typical implicit feedback setting, we set the user rating as ‘1’ if the implicit feedback is observed, and ‘0’ otherwise. For the MovieLens and Google Play data, there are explicit ratings from users to items. The rating scale is [0.5 ~ 5] with step 0.5 for MovieLens data, and [1 ~ 5] with step 1 for Google Play data. As for the content information, we use the aggregated review content in the first data set. We follow standard processing steps including stop-words removal, short-words removal, low-frequency words removal, high-frequency words removal, and stemming. For the other three data sets, we directly use the tag information on items as content input.

4.1.2 Evaluation Results. In this paper, we use the following four evaluation metrics. Specially, we use Root Mean Square Error (RMSE) and Mean Absolute Error (MAE) for the case of explicit feedback, and use Recall@N and Area Under the Curve (AUC) for the case of implicit feedback. In other words, RMSE and MAE are used for the Google Play and MovieLens data, and they are defined as

$$\text{RMSE} = \sqrt{\frac{\sum_{r_{ij} \in T} (\hat{r}_{ij} - r_{ij})^2}{|T|}}$$

$$\text{MAE} = \frac{\sum_{r_{ij} \in T} |\hat{r}_{ij} - r_{ij}|}{|T|}$$

where $T$ is the set of ratings to be evaluated as the test set.

Recall@N and AUC are used for Lastfm and Delicious. For a given user, Recall@N is defined as the ratio between the number of items that the user likes in Top N ranking list, and the total number of items that the user likes; AUC indicates the probability that a randomly chosen observed example is ranked higher than a randomly chosen unobserved example. For these two metrics, we average them over all the users as the final result.

4.1.3 Compared methods. In the experiment section, we use HoORAYs to denote the proposed model that considers content information, and use HoORAYs0 to denote the proposed model without content information. We compare our methods (HoORAYs0 and HoORAYs) with some state-of-the-art recommendation algorithms including probabilistic matrix factorization (PMF) [14], collaborative topic regression (CTR) [20], and Bayesian personalized ranking (BPR) [18]. Note that BPR is specially designed for the implicit feedback, and we only compare with BPR in the implicit feedback case. As for parameters, the dimension of the latent vectors is set to 200 for the proposed methods and all the competitors. The reported results come from the best parameters tuned for each model. For HoORAYs0 and HoORAYs, we set $\alpha = 0.01$, $\lambda_u = 0.1$, and $\lambda_v = 0.1$ ($\lambda_v = 0.5$ for Google Play), where $\alpha$ is the learning rate. Since $\lambda_d$ is more sensitive to data, we set $\lambda_d = 0.01$ on Google Play and MovieLens data sets, $\lambda_d = 0.1$ on Lastfm data, and $\lambda_d = 5$ on Delicious data.

For all the four data sets, we randomly select 75% of the user feedback as training data, the use the remaining data as test set.

4.1.4 Reproducibility of experiments. All the datasets are publicly available. All the parameter settings are stated in the previous subsection. We will release the code of the proposed algorithm through the first author’s website upon the publication of the paper.

4.2 Evaluation Results

Here, we present the experimental results.

*http://moon.nju.edu.cn/people/jingweixu/
Table 2: The comparisons of RMSE results on Google Play and MovieLens data. The proposed methods (HoORAYs0 and HoORAYs) outperform the compared methods.

<table>
<thead>
<tr>
<th>Data set</th>
<th>PMF</th>
<th>CTR</th>
<th>HoORAYs0</th>
<th>HoORAYs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Google Play</td>
<td>1.2958</td>
<td>1.2842</td>
<td>1.2747</td>
<td>1.2733</td>
</tr>
<tr>
<td>MovieLens</td>
<td>0.7764</td>
<td>0.7724</td>
<td>0.7645</td>
<td>0.7620</td>
</tr>
</tbody>
</table>

Table 3: The comparisons of MAE results on for Google Play and MovieLens data. The proposed methods (HoORAYs0 and HoORAYs) outperform the compared methods.

<table>
<thead>
<tr>
<th>Data set</th>
<th>PMF</th>
<th>CTR</th>
<th>HoORAYs0</th>
<th>HoORAYs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Google Play</td>
<td>1.0132</td>
<td>0.9997</td>
<td>0.9855</td>
<td>0.9827</td>
</tr>
<tr>
<td>MovieLens</td>
<td>0.5977</td>
<td>0.5952</td>
<td>0.5814</td>
<td>0.5808</td>
</tr>
</tbody>
</table>

Figure 3: The comparisons of Recall@N results on Lastfm and Delicious data. The proposed methods outperform the compared methods on both data sets.

(a) Lastfm data

(b) Delicious data

Explicit user feedback. We first show the performance of the proposed methods for explicit feedback. Table 2 and Table 3 show the results on Google Play and MovieLens with RMSE and MAE, respectively.

We can first observe from the tables that, the proposed HoORAYs0 and HoORAYs significantly outperform PMF and CTR in terms of both RMSE and MAE. For example, in Table 2, HoORAYs achieves 0.85% and 1.35% improvement over the best competitor (CTR) wrt RMSE on Google Play and MovieLens, respectively. As for the MAE metric in Table 3, HoORAYs outperforms the best competitor (CTR) by 1.70% v.s. 2.42% on Google Play and MovieLens, respectively.

Second, we can see that HoORAYs0 also outperforms CTR on both data sets, although CTR considers content information while HoORAYs0 does not. This indicates the importance of using high-order distance during the optimization process.

Third, the performance on the MovieLens data is better than that on the Google Play data. This is due to the fact that the Google Play data is much sparser than the MovieLens data (0.02% sparsity on Google Play and 3.97% sparsity on MovieLens).

Overall, the above results indicate that the proposed methods are more accurate than the compared methods for the case of explicit feedback, and that the high-order rating distance plays an important role for improving the prediction accuracy of recommendation.

Implicit user feedback. Next, we present the results of the proposed methods for the implicit feedback case. We compare the proposed methods with PMF, CTR, and BPR, and show the results in Fig. 3 and Table 4.

In Fig. 3, we show the Recall results with top N from 10 to 50 with fixed step 5. As we can see, the two proposed methods HoORAYs0 and HoORAYs significantly outperform the compared methods in all cases on both Lastfm and Delicious data. In Fig. 3(a), both HoORAYs0 and HoORAYs are consistently better than the best competitors on Lastfm with 7% improvement on average. In Fig. 3(b), the proposed methods outperform the compared methods especially when number of N is small. For example, when N is 15, HoORAYs0 and HoORAYs achieve 21.7% and 24.3% improvement over the best competitor. Overall, the proposed methods outperform the best competitors with averagely 12.6% improvement in this series of evaluation.

Similar results are observed in Table 4 which shows the AUC scores. Specially, we can observe that HoORAYs0 outperforms the BPR method. This again indicates the usefulness of the proposed high-order distance minimization as BPR uses an AUC-like optimization target. We also notice that the results on Lastfm is better than that on Delicious. Again, this is due to the data sparsity (0.08% sparsity on Delicious v.s. 0.28% sparsity on Lastfm).

Together with the results in Table 2, Table 3, Table 4, and Fig. 3, we can conclude that the proposed methods outperform the compared methods in both explicit feedback case and implicit feedback case. Moreover, the proposed methods can outperform the compared methods even when the content information is unavailable.

Efficiency and Scalability. Next, we present the results of the proposed methods in terms of efficiency and scalability. All the
The effects of $\lambda_u$ and $\lambda_v$ on $K=10, 20, 50, 100, 150, 200$. In general, as shown in Fig. 9(a) and Fig. 9(b), our proposed HoORAYs scales linearly when number of ratings increases.

**Figure 7:** The effects of $\lambda_u$, $\lambda_v$, and $\lambda_d$ of the proposed HoORAYs on MovieLens data.

**Figure 8:** The effects of $\lambda_u$, $\lambda_v$, and $\lambda_d$ of the proposed HoORAYs on Google Play data.

**Figure 9:** The effects of the latent vector dimension $K$ of the proposed HoORAYs with Recall on Lastfm and Delicious data sets.

**Study of Parameters.** Finally, we conduct a parameter study of the proposed methods. We first study the parameters of $\lambda_u$, $\lambda_v$, and $\lambda_d$. We use Recall@50 as an example, and plot the results on Lastfm and Delicious data in Fig. 5 and Fig. 6, respectively. As we can see from the figures, HoORAYs can achieve best performance when $\lambda_u = \lambda_v = 0.1$ for both data sets. As for $\lambda_d$, HoORAYs is sensitive to this parameter, and it achieves the best performance with different $\lambda_d$ for different data sets. In practice, we suggest to set $\lambda_u = \lambda_v = 0.1$ by default, and tune the $\lambda_d$ parameter when using the proposed models. For the effects of parameters on data set with explicit feedback, we study the parameters of $\lambda_u$, $\lambda_v$, and $\lambda_d$ on MovieLens and Google Play data, and the results are plotted in Fig. 7 and Fig. 8. For MovieLens data, HoORAYs have the best performance when $\lambda_u = \lambda_v = 0.1$ in Fig 7(a). As for $\lambda_d$ in Fig 7(b), HoORAYs can achieve good performance when $\lambda_d < 0.05$. As a result, we select $\lambda_d = 0.01$ in our evaluation. As we can see from Fig. 8(a), HoORAYs can achieve best performance when $\lambda_u = 0.1$ and $\lambda_v = 0.5$. For $\lambda_d$, we can find the similar results in Fig. 8(b) to the results on MovieLens data. In practice, we select $\lambda_d = 0.01$ for the evaluation.

Another parameter of the proposed method is the latent vector’s dimension $K$. Fig. 9 presents the effects of $K$ with Recall@N on Lastfm and Delicious data. We vary the size of $K$ with $K = 10, 20, 50, 100, 150, 200$. In general, as shown in Fig. 9(a) and Fig. 9(b),
a larger $K$ usually brings better performance before it overfits the data set. Fig. 9(a) shows the performance with different $K$ selection on Lastfm data. We can also observe from the figures that, the Recall performance improves significantly when we increase $K$ from 10 to 100, and the improvement becomes minor when $K$ is larger than 100. In practice, we suggest to set $K$ between 100 and 200 for the proposed HoORAYs.

5 RELATED WORK

In this section, we briefly review some related work.

Collaborative filtering approaches with user feedback as input have been widely used in recommender systems [7, 10, 11, 14]. For example, matrix factorization methods [11, 14] take ratings as input, and learn the latent vectors of users and items for recommendation. To improve recommendation accuracy, side information is also widely explored. For example, Wang et al. [20] and McAuley and Leskovec [13] incorporate content information; Ma et al. [12] and Tang et al. [19] incorporate social relationships; Chen et al. [2] and Purushotham et al. [17] consider both content and social information.

While many recommendation algorithms are designed for explicit user feedback, several researchers put their focus on case of implicit user feedback. For example, Rendle et al. [18] propose Bayesian personalized ranking to optimize the rankings instead of ratings. Formulating the problem as one-class collaborative filtering, traditional approaches are also adapted for implicit feedback [8, 15, 16], and side information is also considered in this one-class setting [23, 24].

Different from and orthogonal to most of the existing recommendation methods, we propose a new regularized optimization problem by involving high-order rating distance as the constraint for shrinking the solution space. By reducing the variance, the better recommendation accuracy could be reached. Similar strategies are also explored in some related problems. For example, Rendle et al. [18] and Kabbur et al. [9] propose an AUC-like optimization function. Our high-order optimization problem is different from the AUC-like optimization as we use the other existing ratings to shrink the solution space while AUC focuses on the order of observed-unobserved examples; additionally, we have experimentally shown that the proposed method outperforms BPR with same input. This work generalizes the rating comparison strategy [21, 22], which can be viewed as a second-order rating distance, primarily designed for the cold-start case. Moreover, it also justifies the rationality behind the higher-order rating distance from two complementary perspectives (the geometric vs. Bayesian interpretations).

6 CONCLUSION

In this paper, we have proposed a high-order optimization of rating distance for recommender systems HoORAYs to further reduce the solution space of latent vectors for users and items. The proposed HoORAYs model used second-order rating distance as the constraint to the optimization problem. HoORAYs can be applied for both explicit and implicit user feedback. We presented a geometric interpretation to show how HoORAYs helps reduce the variance of the estimated latent factors. Based on the Bayesian interpretation, we further explained the HoORAYs from the generative model perspective. By connecting to CTR rating model, our HoORAYs can naturally handle the case when content information is available. The experimental evaluations on four real-world data sets show that the proposed method consistently outperforms the state-of-the-art methods in terms of prediction accuracy.

ACKNOWLEDGMENTS

This work is supported by the National 863 Program of China (No. 2015AA01A203), National Key Research and Development Program of China (No. 2016YFB1000802), National Natural Science Foundation of China (No. 61373011, 61690204), the Collaborative Innovation Center of Novel Software Technology and Industrialization. This work is supported by NSF (IIS-1651203), DTRA (HDTRA1-16-0017), ARO (W911NF-16-1-0168), NIH (R01LM011986), and a Baidu gift.

REFERENCES


[18] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In


