IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received March 10, 2018, accepted April 11, 2018, date of publication April 23, 2018, date of current version May 24, 2018.

*Digital Object Identifier 10.1109/ACCESS.2018.2828401*

# Collaborative Filtering Recommendation Based on All-Weighted Matrix Factorization and Fast Optimization

**HONGMEI LI[1], (Fellow, IEEE), XINGCHUN DIAO[1], (Member, IEEE), JIANJUN CAO[2], (Member, IEEE), AND QIBIN ZHENG[1], (Member, IEEE)**
[1]Command and Control Engineering College, Army Engineering University of PLA, Nanjing 210007, China
[2]Nanjing Telecommunication Technology Institute, Nanjing 210007, China

Corresponding author: Jianjun Cao (jianjuncao@yeah.net)

This work was supported by the Natural Science Foundation of China under Grant 61371196.

**ABSTRACT** Collaborative filtering recommendation with implicit feedbacks (e.g., clicks, views, and plays) is regarded as one of the most challenging issues in both academia and industry. From implicit feedbacks, we can only get a small fraction of observed data (positive examples), and the massive unobserved data are the mixture of negative examples and unlabeled positive examples. However, most of the existing efforts either treat unobserved data equally by assigning a uniform weight or uniformly weight observed data while ignoring the hidden information (i.e., visit frequency) in implicit feedbacks. This assumption may not hold in real-life scenarios since they cannot distinguish the contributions of the whole data and it easily leads to prediction bias. Besides, those approaches still suffer from low-efficiency issue. To this end, we propose an all-weighted matrix factorization and fast optimization strategy for effective and efficient recommendation. We first design a frequency-aware weighting scheme for observed data and a user-oriented weighting scheme for unobserved data nonuniformly. Then, the weighting schemes of both observed and unobserved data are combined in a unified way to form an all-weighted matrix factorization model. Afterwards, we present a surrogate objective function and develop a fast optimization strategy to enhance the efficiency. Extensive experimental results on real-world datasets demonstrate that our method outperforms the competitive baselines on several evaluation metrics.

**INDEX TERMS** Personalized recommendation, implicit feedback, collaborative filtering, all-weighted matrix factorization, fast optimization, visit frequency.

## I. INTRODUCTION

Recent years have witnessed the prevalence of personalized recommendation in various real applications, such as Netflix, Last.fm, Foursquare, Amazon, etc. Personalized recommendation makes it much easier for people to acquire their needs in a short time. State-of-the-art works indicate that collaborative filtering (CF) recommendation with implicit feedbacks has been receiving more and more attentions due to its effectiveness and flexibility. It predicts users' preferences based on user-item interactions among a large number of users and items. For the interaction data, compared with explicit feedbacks (e.g., numerical ratings), implicit feedbacks are relatively more abundant and can be easily collected, since most users express their preferences implicitly. Typical implicit feedbacks include views, purchases, clicks, check-ins, etc. However, this kind of data are extremely sparse and imbalanced. For a specific user, only a small fraction of items are observed (positive examples), and the majority are unknown. This has been a thorny issue in CF recommendation with implicit feedbacks [1]–[6].

Matrix factorization (MF) models have received great success in CF recommendation with both explicit feedbacks and implicit feedbacks. The majority of existing strategies for CF with implicit feedbacks can be divided into two categories: pointwise regression and pairwise ranking strategies. Two typical approaches are weighted regularized matrix factorization (WRMF) [1], [3], [6]–[10] and Bayesian personalized ranking with matrix factorization (BPR-MF) [8], [10]–[17]. Bayer *et al.* [5] have argued that both WRMF and BPR-MF have been compared on a variety of datasets and a large body of results indicate both approaches have their merits.

WRMF is a kind of pointwise regression approach which learns user/item latent factors by minimizing the weighted reconstruction error. One major limitation of most existing

researches on WRMF is the uniform weighting scheme on the unobserved data, which limits the model's effectiveness and extensibility. Only several works have considered non-uniform weighting scheme [3], [2]. However, all observed items are assumed to be equally relevant for a specific user. This assumption may not hold in real-life scenarios where the numerical values of implicit feedbacks describe the visit frequency of users, e.g., how much time the user has watched a certain movie, how many times a user has visited a certain position, etc. In general, the visit frequency reflects users' graded preferences on different items. A larger value indicates the higher confidence that user prefers a certain item. Traditional methods simplify users' actions in frequencies on a unique item, and only consider the binary values which indicate whether a user has visited the items. Hu *et al.* [1] and Lian *et al.* [18] have noticed this problem and proposed to assign confidence weights according to the visit frequencies. Nevertheless, they treat the unobserved items equally by assigning the uniform weights. This kind of treatment ignores the unique characteristic of different users and items, and limits model's fidelity for real applications. This is because that the unobserved examples are the mixture of negative and unknown examples, it is desired to assign a higher weight to the negative examples. If we simply impose uniform weights on all of the unobserved data, it may mislead the learning process.

BPR-MF is a ranking approach and optimizes pairwise objective that attempts to place observed items above the unobserved ones [19]. However, we argue that all observed (unobserved) items are assumed to be equally relevant (irrelevant). This assumption not only ignores the definitely useful numerical values which indicate preference confidence, but also models unobserved data in an unrealistic way. This is clearly suboptimal. Some state-of-the-art works also suggest to leverage auxiliary data to assist in improving WRMF and BPR-MF via varying weighting scheme [20], [21] and augmenting preference values [10]. Nevertheless, they did not take full consideration of the weights of whole data (both of the observed and unobserved data). These observations suggest that it is highly desirable to consider an all-weighted matrix factorization model.

In addition, we also argue that the efficiency of WRMF suffers from the full consideration of both observed and unobserved data. Aimed at matrix factorization with whole data, alternating least square (ALS) [1] is a relatively more effective method compared with stochastic gradient descent (SGD) [19]. ALS works without learning rate via a closed-form updating formulation. It could converge to a satisfied local optima with finite steps much less than that of SGD [1], [2]. By contrast, SGD is subjected to the choice of learning rate which has been proved by [3]. Even so, it has been demonstrated that ALS algorithm involves matrix inversion which is an expensive operation when updating one user/item latent vector [1], [2]. Its inefficiency is the main obstacle for practical use. Aimed at this problem, Rendle *et al.* [22] presented an element-wise ALS (eALS) learner, which avoids the expensive matrix inversion and reduces the time complexity effectively. Afterwards, Devooght *et al.* [23] proposed the randomized block coordinate descent (RCD) learner to reduce the complexity in a dynamic scenario. However, it only applies to WRMF with uniform weights for unobserved data, which is sub-optimal for real applications. Recently, a fast eALS learning algorithm was proposed by He *et al.* [3], which further speeds up learning by avoiding the massive repeated computations. Due to this reason, in this paper, we mainly focus on eALS-based optimization tailored for WRMF. Nevertheless, the above method aims at item-oriented weighting scheme for unobserved data, it could not be directly applied to our proposed all-weighted regularized matrix factorization model directly. In this case, we devise an alternative optimization method tailored for our proposed model.

Based on our preliminary analysis, we put forward an effective and efficient all-weighted matrix factorization model for item recommendation with implicit feedbacks. Specifically, we first design a weighted regularized matrix factorization which takes full consideration of the weighting scheme of both observed and unobserved data. In particular, we explore a user-oriented non-uniform weighting scheme for unobserved data, and frequency-aware non-uniform weighting scheme for observed data, with more flexibility and fidelity for real applications. Then, we present a surrogate objective function for efficient training and develop a fast optimization method based on element-level alternative least square. Our algorithm could find the optimal solution for model parameters within a few iterations.

To summarize, our contributions are:

1. We design an effective all-weighted matrix factorization model, which takes account into both observed data and unobserved data and assigns non-uniform weights.
2. We develop a fast optimization algorithm for learning model parameters efficiently.
3. We conduct experiments on two real-world datasets, showing that our method consistently outperforms state-of-the-art matrix factorization methods on top-$n$ recommendation tasks.

The remainder of this paper is arranged as follows. In section II we review the related work on matrix factorization and optimization methods for collaborative filtering recommendation with implicit feedbacks. In section III we give detailed explanation of our all-weighted matrix factorization model and fast optimization scheme. The experimental results and analysis are presented in Section IV, followed by the conclusions and future work in section V.

## II. RELATED WORK

In this section, we describe existing matrix factorization approaches for recommendation with implicit feedbacks and optimization methods.

### A. MATRIX FACTORIZATION FOR RECOMMENDATION WITH IMPLICIT FEEDBACK

Emerging popularity of e-commerce has highlighted the importance of CF recommendation, and various models have been studied. In recent years, matrix factorization models have received great success in CF recommendation with implicit feedbacks due to their accuracy and efficiency [3], [23]–[27]. Both pointwise regression approaches and pairwise ranking approaches based on matrix factorization are popular for CF recommendation. Here we mainly review the two most popular approaches: WRMF [1], [3], [6]–[10] and BPR-MF [8], [10]–[17].

WRMF [1], [2] is a kind of rating-based approach, which assumes observed feedbacks as positive ratings and unobserved feedbacks as negative ratings with corresponding confidence weights. Typically, Hu *et al.* [1] and Lian *et al.* [18] proposed the WRMF methods for CF with implicit feedbacks, which assigned the higher confidence levels to observed data according to the visit frequency, while treating unobserved data equally by assigning a lower weight. However, this uniform assumption on unobserved data ignores the unique characteristics of different users and items, and limits model's fidelity in real applications. Actually, in addition to [1] and [18], the majority of existing works [6], [23], [28], [29] assigned uniform weights on unobserved data. This is due to that the whole user-item interaction matrix is generally very large and sparse, it would be too consuming to store and compute all zero entries with individualized weights. The uniform weighting scheme would make the algorithm more efficient [3]. However, it would reduce the accuracy of recommendation to some extent. Only several works considered non-uniform weighting on unobserved data [2], [3]. To be specific, Pan *et al.* [2] designed a WRMF model by assigning equal weights on observed data, followed by the lower weights of unobserved data via item- and user-oriented weighting scheme. The item-oriented weighting scheme posits that if an item has less observed uses, that is, the item is not popular, the unobserved data for this item are negative with higher probability. The user-oriented weighting scheme assumes that if a user has more positive examples, it is more likely that he/she does not like the other items, that is, the unobserved data for this user is negative with higher probability. Different from [2], He *et al.* [3] proposed an alternative item-oriented WRMF approach based on item popularity, which posits that if an item has more observed users, a miss on such a popular item is more probable to be negative with higher probability. It is more flexible and effective than [2]. However, their common limitation is that they treat the observed data equally, this assumption would not hold in real-life scenarios where numerical implicit signals (i.e., the visit frequency of users on specific items) imply graded preference. The larger values imply the higher confidence that users prefer the items. Simply considering the binary values would affect the recommendation accuracy of model.

BPR-MF [19] is a type of ranking-based approach, which regards implicit feedbacks as relative preferences rather than absolute ratings. It assumes that a user usually prefers an observed item to an unobserved item. BPR-MF integrates MF into ranking-oriented BPR optimization framework to perform personalized ranking and recommendation. Various researches based on BPR-MF framework have been made [8], [10]–[17]. Nevertheless, this pairwise ranking assumption may not always hold in real-life scenarios where noisy implicit signals tend to produce incorrect/outlier preferences. Moreover, they equally treat all of the observed/unobserved data, which is the common limitation of BPR-MF and WRMF.

In addition, a variety of works are reported to leverage additional data to improve WRMF and BPR-MF [10], [20], [21]. For example, Yuan *et al.* [21] considered incorporating the rich user and item content information to determine the personalized weight of every unknown user-item pair and improved recommendation performance. In contrast to the previous works, Li *et al.* [10] focused on varying the preference values instead of the weighting schemes and presented an augmented WRMF. It suggested augmenting the binary preference values to ternary values, where the middle values indicate users' potential preference to those potential items (i.e., the unvisited items that her friends have visited before). Guo *et al.* [20] developed a weighted ranking method for POI recommendation, which gives each POI pair a different weight according to the visit frequency and the geographical distance between them. Nevertheless, we argue that they either only consider the unique weighting scheme of observed data or only focus on that of unobserved data partially. These observations motivate us to take full consideration of the weighting scheme of the whole data, to reduce prediction bias and improve recommendation accuracy.

### B. OPTIMIZATION SCHEME

Typically, alternating least squares (ALS) and stochastic gradient descent (SGD) are the two commonly used methods to minimize objective function of WRMF. For BPR-MF, SGD is the most popular optimization method.

In general, the objective of WRMF is a non-convex minimization problem, and it may have many local optima. For this problem, ALS is a more effective method. This is due to that it is not sensitive to the starting point, and it ensures the exact decrease of objective function and converges to a local optimum at least under the mathematical theory. SGD could not assure the convergence in mathematical theory. In addition, ALS has a significant advantage over SGD since ALS has less parameters to turn with a closed-form updating formulation. And it can converge to a satisfied local optima with finite steps much less than that of SGD. SGD is easily subjected to the learning rate. He *et al.* [3] have tested and verified that a higher learning rate would lead to a faster convergence, but the final accuracy may be suffered.

Through Volkovs and Yu [6] claim that the convergence of BPR-MF can be achieved significantly faster through sampling, the gradient updates are usually distributed very unevenly. The gradient updates are inclined to the popular items and easily lead to prediction bias. A good sampling approach is desired to improve BPR-MF. Moreover, SGD is unsuitable for whole-data based WRMF due to the large amount of training instances [3]. Due to this reason, in this paper, we mainly focus on ALS-based optimization tailored for whole-data based WRMF.

Nevertheless, ALS is expensive in computational overhead due to the inversion operation of $K \times K$ matrix when updating a user/item latent vector. It usually assumes $O(K^3)$ in time complexity, where $K$ is the size of latent factors of users/items [2]. Fortunately, Rendle *et al.* [22] designed an element-level ALS (eALS) to lower time complexity, which optimizes parameters at the element level optimizing each coordinate of the latent vector. This is because eALS avoids the expensive computation of matrix inversion. Recently, He *et al.* [3] developed a faster eALS learning algorithm by avoiding the massive repeated computations introduced by the weighted unobserved data.

## III. PROPOSED METHOD

In this section, we first give the definition of problem, and then introduce an all-weighted matrix factorization model followed by a fast optimization method.

### A. DEFINITION

Let $\mathcal{U} = (1, \ldots, u \ldots, M)$ be the set of $M$ users, $\mathcal{I} = (1, \ldots, i \ldots, N)$ be the set of $N$ items, and $\mathcal{K} = (1, \ldots, k \ldots, K)$ be the set of $K$ dimensions. We use matrix $R$ to represent the whole user-item interactions, where $R_{ui} = 1$ denotes that user $u$ has interaction with item $i$ (i.e., the observed data), otherwise $R_{ui} = 0$ (i.e., the unobserved data). Following matrix factorization, we denote users and items as $K$-dimensional latent factor matrices $U \in \mathbb{R}^{M \times K}$ and $V \in \mathbb{R}^{N \times K}$ respectively, which share the same latent space. The symbols and descriptions are shown in Table 1. Our goal is to recommend a personalized ranking list of items for each user $u$ given the observed interactions, including the frequency of interactions.

In the following, we first propose an all-weighted matrix factorization model to improve recommendation performance, by data. Afterwards, we develop a fast optimization algorithm using eALS to enhance learning efficiency.

### B. ALL-WEIGHTED MATRIX FACTORIZATION MODEL

Formally, the optimization objective function of weighted matrix factorization is formulated as a squared loss

$$L(U, V) = \sum_{u=1}^{M} \sum_{i=1}^{N} C_{ui}(R_{ui} - \hat{R}_{ui})^2 + \lambda\left(\sum_{u=1}^{M} ||U_u||^2 + \sum_{i=1}^{N} ||V_i||^2\right) \quad (1)$$

**TABLE 1. Symbols and description.**

| Symbols | Description |
|---|---|
| $\mathcal{U}$ | The set of users |
| $\mathcal{I}$ | The set of items |
| $M$ | Number of users |
| $N$ | Number of items |
| $K$ | Number of latent factors |
| $U(U_{uk})$ | User latent factor matrix (k-th latent factor of user $u$ ) |
| $V(V_{ik})$ | Item latent factor matrix (k-th latent factor of item $i$ ) |
| $R(R_{ui})$ | User-item interaction matrix (binary preference of $u$ to $i$ ) |
| $\hat{R}(\hat{R}_{ui})$ | Predicted rating matrix (predicted preference of $u$ to $i$ |
| $C(C_{ui})$ | Confidence matrix (confidence weight of user $u$ to item $i$ ) |
| $f_{ui}$ | The frequency of interactions of user $u$ with item $i$ |
| $R_o$ | The set of observed user-item pairs |
| $R_u$ | The set of items observed by user $u$ |
| $R_i$ | The set of users observed by item $i$ |
| $\|R_o\|$ | The size of observed data |
| $\|R_u\|$ | The size of items observed by user $u$ |
| $\|R_i\|$ | The size of users observed by item $i$ |
| $\lambda$ | Regularization parameter of $U$ and $V$ |

where $C_{ui}$ denotes the weight of each user-item pair, $\hat{R}_{ui}$ is the predicted score following matrix factorization, $\hat{R}_{ui} = \sum_{k=1}^{K} U_{uk} V_{ik}$. $\lambda$ is the regularization parameter to prevent overfitting.

The key of WRMF model (Equation (1)) is the weighting scheme of both observed data and unobserved data. We discussed earlier that treating all observed (unobserved) items as equal is not optimal. In this section, we build on this idea and present an all-weighted matrix factorization model that takes fully account into the weighting scheme of the whole data, followed by the appropriate surrogate objective function for efficient training.

#### 1) THE WEIGHTING SCHEME OF OBSERVED DATA

Traditionally, some works ignore the difference of visit frequency, and treat the observed data equally by giving uniform weights [2], [3]. This kind of treatment misses significant preference information and affects the accuracy of recommendation. Some recent works noticed this problem, and assigned weights to observed data according to the visit frequency [1], [20]. Take [1] for instance, it assigns a uniform weight $C_{ui} = 1$ to each unobserved user-item pair, and introduces a constant $\alpha$ to ensure the confidence weight of per observed user-item pair grows linearly with the absolute frequency $f_{ui}$, $C_{ui} = 1 + \alpha f_{ui}$. Nevertheless, they ignore the frequency scales of different users. For example, some users visit some positions frequently, and their average visit frequency is relatively higher than that of other users. If we determine the weights of observed data only according to the absolute frequency, that is treating different users equally, it may lead to prediction bias. They also ignore to discriminate unobserved data, we will tackle this issue in the next subsection.

For this reason, we propose a weighting scheme for observed data according to the relative frequency of each users. The weight is restricted to the total frequencies and the number of actions for each user. It is defined as

$$C_{ui} = \alpha\,|R_u|\,\frac{f_{ui}^{\tau}}{\sum\limits_{i\in R_u} f_{ui}^{\tau}} \tag{2}$$

where $\alpha$ determines the global weight of observed data. $f_{ui}$ denotes how many times user $u$ visits item $i$. In order to enhance the flexibility and effectiveness, we also introduce $\tau$ to control the level of frequent items over other infrequent ones. Specifically, when $\tau = 0$, $C_{ui}$ turns into a constant $\alpha$, which means the uniform confidence on different observed user-item pairs. When $\tau \neq 0$, the confidence of frequent items is strengthened to distinguish the other infrequent items.

#### 2) THE WEIGHTING SCHEME OF UNOBSERVED DATA

For implicit feedbacks, only a small fraction of them are observed examples, and the majority are unobserved. The unobserved data are the mixture of negative examples and unlabeled positive examples that may be preferred by users. We expect the negative examples with higher weights. Hence, Pan *et al.* [2] and He *et al.* [3] proposed to treat them unequally . For example, He *et al.* [3] proposed item-oriented weighting scheme for unobserved data, which assumes that a miss on a popular item is more likely to be truly irrelevant (i.e., a higher probability to be negative) to the user. Different from the above, we introduce an alternative non-uniform user-oriented weighting scheme. The user-oriented weighting scheme assumes that if a user has more positive examples, it is more likely that she does not like the other items, that is, the unobserved data for this user is negative with higher probability.

Accordingly, we define the unobserved weights as

$$C_{ui} = \beta\,\frac{|R_u|^{\pi}}{\sum\limits_{u=1}^{M} |R_u|^{\pi}} \tag{3}$$

where $\beta$ is the coefficient to control the overall weight of unobserved data. $|R_u|$ is the number of observed items for user $u$. Similarly, we introduce $\pi$ to control the level of active users over other not active ones. When $\pi = 0$, $C_{ui}$ turns into a constant $C_{ui} = \frac{\beta}{M}$, which means the uniform weighting for each unobserved data. When $\pi \neq 0$, the non-uniform weight $C_{ui}$ is controlled by $\pi$. Compared with [2], our weighting scheme is more flexible.

#### 3) THE SURROGATE OBJECTIVE FUNCTION

It is worth to say that, since the weight $C_{ui}$ of unobserved user-item $(u, i)$ pair has no concern with item $i$, it is only dependent on user $u$. Thus, we replace $C_{ui}$ with $d_u$. Accordingly, we rewrite Equation (3) as

$$d_u = \beta\,\frac{|R_u|^{\pi}}{\sum\limits_{u=1}^{M} |R_u|^{\pi}} \tag{4}$$

Thus, the original objective function (Equation (1)) is transformed into the following surrogate formulation

$$L(U,V) = \sum_{u=1}^{M}\sum_{i\in R_u} C_{ui}(1-\hat{R}_{ui})^2 + \sum_{u=1}^{M}\sum_{i\in\mathcal{I}\setminus R_u} d_u(0-\hat{R}_{ui})^2 + \lambda\Big(\sum_{u=1}^{M}||U_u||^2 + \sum_{i=1}^{N}||V_i||^2\Big) \tag{5}$$

In the following, we study how to optimize the surrogate objective functions for efficient training.

### C. FAST OPTIMIZATION USING EALS

We first optimize Equation (5) using eALS and get a closed-form updating formulations of $U_{uk}$ and $V_{ik}$. Then we develop a fast learning algorithm by avoiding the massive unnecessary computations.

#### 1) FAST OPTIMIZATION OF USER LATENT FACTOR $U_{uk}$

① Closed-form solution of $U_{uk}$

Specifically, we first take partial derivatives of $L(\mathbf{U}, \mathbf{V})$ over $U_{uk}$

$$\frac{\partial L(U,V)}{\partial U_{uk}} = \sum_{i\in R_u} C_{ui}(1-\hat{R}_{ui})(-V_{ik}) + \sum_{i\in\mathcal{I}\setminus R_u} d_u(0-\hat{R}_{ui})(-V_{ik}) + \lambda U_{uk} \tag{6}$$

Since we can represent $\hat{R}_{ui}$ as $\hat{R}_{ui} = \hat{R}_{ui}^{k} + U_{uk}V_{ik}$, Equation (6) is transformed into

$$\frac{\partial L(U,V)}{\partial U_{uk}} = \sum_{i\in R_u} C_{ui}(1-\hat{R}_{ui}^{k}-U_{uk}V_{ik})(-V_{ik}) + \sum_{i\in\mathcal{I}\setminus R_u} d_u(0-\hat{R}_{ui}^{k}-U_{uk}V_{iki})(-V_{ik}) + \lambda U_{uk} \tag{7}$$

By setting the derivative to zero, $\frac{\partial L(U,V)}{\partial U_{uk}} = 0$, we obtain the closed-form solution of $U_{uk}$

$$U_{uk} = \frac{\sum\limits_{i\in R_u} C_{ui}(1-\hat{R}_{ui}^{k})V_{ik} + \sum\limits_{i\in\mathcal{I}\setminus R_u} d_u(0-\hat{R}_{ui}^{k})V_{ik}}{\sum\limits_{i\in R_u} C_{ui}V_{ik}^2 + \sum\limits_{i\in\mathcal{I}\setminus R_u} d_u V_{ik}^2 + \lambda} \tag{8}$$

where $\hat{R}_{ui}^{k} = \sum\limits_{k'\in\mathcal{K}\setminus k} U_{uk'}V_{ik'}$.

② Fast updating of $U_{uk}$.

It is worth to say that, since users typically rate only a small fraction of all available items, the size of unobserved data is

much larger than the observed ones. The majority of computational cost lies in the massive unobserved data part, they are $\sum_{i\in\mathcal{I}\setminus R_u} d_u(0-\hat{R}^k_{ui})V_{ik}$ and $\sum_{i\in\mathcal{I}\setminus R_u} d_u V^2_{ik}$ of Equation (8).

Fortunately, it could be achieved by separating the less observed data from the whole data to enhance the learning speed. The main idea behind that is the whole data part is independent with user $u$, and it could be pre-computed to avoid the massive unnecessary computation. Accordingly, we could reformulate the unobserved data part as following:

For the term $\sum_{i\in\mathcal{I}\setminus R_u} d_u(0-\hat{R}^k_{ui})V_{ik}$, it can be rewritten as

$$\begin{aligned}
&\sum_{i\in\mathcal{I}\setminus R_u} d_u(0-\hat{R}^k_{ui})V_{ik} \\
&= d_u\sum_{i=1}^{N}(0-\hat{R}^k_{ui})V_{ik} - d_u\sum_{i\in R_u}(0-\hat{R}^k_{ui})V_{ik} \\
&= -d_u\sum_{i=1}^{N}(\sum_{\mathcal{K}\setminus k} U_{uk'}V_{ik'})V_{ik} + \sum_{i\in R_u} d_u\hat{R}^k_{ui}V_{ik} \\
&= -d_u\sum_{k'\in\mathcal{K}\setminus k} U_{uk'}\sum_{i=1}^{N} V_{ik'}V_{ik} + \sum_{i\in R_u} d_u\hat{R}^k_{ui}V_{ik} \qquad (9)
\end{aligned}$$

Since the whole data part, that is $P = \sum_{i=1}^{N} V_i^T V_i = V^T V$ ($P \in \mathbb{R}^{K\times K}$), is independent with user $u$, we could pre-compute it and reformulate Equation (9) as

$$\begin{aligned}
&\sum_{i\in\mathcal{I}\setminus R_u} d_u(0-\hat{R}^k_{ui})V_{ik} \\
&\qquad = -d_u\sum_{k'\in\mathcal{K}\setminus k} U_{uk'}P_{k'k} + \sum_{i\in R_u} d_u\hat{R}^k_{ui}V_{ik} \qquad (10)
\end{aligned}$$

Note that the above expression involves only the small set of items visited by user $u$, denoted by $R_u$, and this could be implemented efficiently.

Similarly, the term $\sum_{i\in I\setminus R_u} d_u V^2_{ik}$ can be rewritten as

$$\begin{aligned}
\sum_{i\in I\setminus R_u} d_u V^2_{ik} &= d_u\sum_{i=1}^{N} V^2_{ik} - d_u\sum_{i\in R_u} V^2_{ik} \\
&= d_u P_{kk} - d_u\sum_{i\in R_u} V^2_{ik} \qquad (11)
\end{aligned}$$

By incorporating Equation (10) and Equation (11) into Equation (8), $U_{uk}$ is reformulated as

$$U_{uk} = \frac{\sum_{i\in R_u}(C_{ui}(1-\hat{R}^k_{ui}) + d_u\hat{R}^k_{ui})V_{ik} - d_u\sum_{k'\in\mathcal{K}\setminus k} U_{uk'}P_{k'k}}{\sum_{i\in R_u}(C_{ui}-d_u)V^2_{ik} + d_u P_{kk} + \lambda} \qquad (12)$$

### 2) FAST OPTIMIZATION OF ITEM LATENT FACTOR $V_{ik}$

① Closed-form solution of $V_{ik}$

The counterpart for $V_{ik}$ is performed likewise. Taking partial derivative of $L(\mathbf{U},\mathbf{V})$ over $V_{ik}$

$$\begin{aligned}
\frac{\partial L}{\partial V_{ik}} &= \sum_{u\in R_i\mathcal{U}} C_{ui}(1-\hat{R}_{ui})(-U_{uk}) \\
&\quad + \sum_{u\in\mathcal{U}\setminus R_i} d_u(0-\hat{R}_{ui})(-U_{uk}) + \lambda V_{ik} \qquad (13)
\end{aligned}$$

With the transformation of $U_u^T V_i = \hat{R}^k_{ui} + U_{uk}V_{ik}$, we rewrite the above equation as

$$\begin{aligned}
\frac{\partial L}{\partial V_{ik}} &= \sum_{u\in R_i} C_{ui}(1-\hat{R}^k_{ui} - U_{uk}V_{ik})(-U_{uk}) \\
&\quad + \sum_{u\in\mathcal{U}\setminus R_i} d_u(0-\hat{R}^k_{ui} - U_{uk}V_{ik})(-U_{uk}) + \lambda V_{ik} \qquad (14)
\end{aligned}$$

By setting the derivative to zero, $\frac{\partial L(U,V)}{\partial V_{ik}} = 0$, we obtain the closed-form solution of $V_{ik}$

$$V_{ik} = \frac{\sum_{u\in R_i} C_{ui}(1-\hat{R}^k_{ui})U_{uk} + \sum_{u\in\mathcal{U}\setminus R_i} d_u(0-\hat{R}^k_{ui})U_{uk}}{\sum_{u\in R_i} C_{ui}U^2_{uk} + \sum_{u\in\mathcal{U}\setminus R_i} d_u U^2_{uk} + \lambda} \qquad (15)$$

② Fast updating of $V_{ik}$

Analogously, we reformulate the two terms of $V_{ik}$ (Equation (15)) for more fast optimization, they are $\sum_{u\in\mathcal{U}\setminus R_i} d_u(0-\hat{R}^k_{ui})U_{uk}$ and $\sum_{u\in U\setminus R_i} d_u U^2_{uk}$.

We rewrite $\sum_{u\in\mathcal{U}\setminus R_i} d_u(0-\hat{R}^k_{ui})U_{uk}$ as

$$\begin{aligned}
&\sum_{u\in\mathcal{U}\setminus R_i} d_u(0-\hat{R}^k_{ui})U_{uk} \\
&= \sum_{u=1}^{M} d_u(0-\hat{R}^k_{ui})U_{uk} - \sum_{u\in R_i} d_u(0-\hat{R}^k_{ui})U_{uk} \\
&= -\sum_{u=1}^{M} d_u\sum_{k'\in\mathcal{K}\setminus k} U_{uk'}V_{ik'}U_{uk} + \sum_{u\in R_i} d_u\hat{R}^k_{ui}U_{uk} \\
&= -\sum_{k'\in\mathcal{K}\setminus k} V_{ik'}\sum_{u=1}^{M} d_u U_{uk'}U_{uk} + \sum_{u\in R_i} d_u\hat{R}^k_{ui}U_{uk} \qquad (16)
\end{aligned}$$

Due to that $Q = \sum_{u=1}^{M} d_u U_u^T U_u (Q \in \mathbb{R}^{k\times k})$ has no connection with item $i$, we could preprocess it and Equation (16) is simplified as

$$\begin{aligned}
&\sum_{u\in\mathcal{U}\setminus R_i} d_u(0-\hat{R}^k_{ui})U_{uk} \\
&\qquad = -\sum_{k'\in\mathcal{K}\setminus k} V_{ik'}Q_{k'k} + \sum_{u\in R_i} d_u\hat{R}^k_{ui}U_{uk} \qquad (17)
\end{aligned}$$

where $Q$ can also be rewritten as $Q = U^T\tilde{d}U$ in real implementation, $\tilde{d}$ is a diagonal matrix with the elements of $d$ on the diagonal.

Likewise, the term $\sum_{u\in U\setminus R_i} d_u U_{uk}^2$ can be rewritten as

$$\sum_{u\in U\setminus R_i} d_u U_{uk}^2 = \sum_{u=1}^{M} d_u U_{uk}^2 - \sum_{u\in R_i} d_u U_{uk}^2 = Q_{kk} - \sum_{u\in R_i} d_u U_{uk}^2 \tag{18}$$

By incorporating Equation (17) and Equation (18) into Equation (15), we obtain the new formulation of $V_{ik}$

$$V_{ik} = \frac{\sum_{u\in R_i} (C_{ui}(1-\hat{R}_{ui}^k) + d_u \hat{R}_{ui}^k) U_{uk} - \sum_{k'\in\mathcal{K}\setminus k} V_{ik'} Q_{\mathrm{k'k}}}{\sum_{u\in R_i} (C_{ui} - d_u) U_{uk}^2 + Q_{kk} + \lambda} \tag{19}$$

#### 3) LEARNING ALGORITHM

The whole training procedure is shown in Algorithm 1.

**Algorithm 1** WRMF-UO

**1. Input**: user-item interaction matrix $R$, weighting matrix $w$, dimension $K$ of latent factor matrix, regularization parameters $\lambda$, and the maximal number of iterations $\gamma$.
**2. Output**: Parameters $\theta = \{U, V\}$
**3. Initialize** $\theta$: $U \sim N(0, 0.1)$, $V \sim N(0, 0.1)$
**4. Compute** $\hat{R}_{ui} = U_u V_i^T (\forall (u, i) \in R_o)$
**5. while** not convergence or iteration $<$ max_iteration
**6. precompute** $P = V^T V$
7. **for** $u = 1 \ldots M$
8. **for** $k = 1 \ldots K$
9. Compute $\hat{R}_{ui}^k = \hat{R}_{ui} - U_{uk} V_{ik} (\forall i \in R_u)$
10. Update $U_{uk}$ via Equation (12)
11. Update $\hat{R}_{ui} = \hat{R}_{ui}^k + U_{uk} V_{ik} (\forall i \in R_u)$
12. **endfor**
13. **endfor**
**14. precompute** $Q = U^T \tilde{d} U$
15. **for** $i = 1 \ldots N$
16. **for** $k = 1 \ldots K$
17. Compute $\hat{R}_{ui}^k = \hat{R}_{ui} - U_{uk} V_{ik} (\forall u \in R_i)$
18. Update $V_{ik}$ via Equation (19)
19. Update $\hat{R}_{ui} = \hat{R}_{ui}^k + U_{uk} V_{ik} (\forall u \in R_i)$
20. **endfor**
21. **endfor**
**22. return** $\theta = \{U, V\}$

With the learned parameters $\theta = \{U, V\}$, we can easily obtain the predicted scores of the unobserved items for each user over $\hat{R}_{ui} = \sum_{k=1}^{K} U_{uk} V_{ik}$. Then we rank items in a decreasing order according their scores, and generate top-$n$ recommendation list of items for each user.

### D. COMPLEXITY ANALYSIS

Time complexity: Updating each user $u$ with one latent factor takes $O(|R_u| + K)$ time. For $m$ users with $K$ latent factors, the time complexity is $O(|R_o|K + MK^2)$, where $|R_o|$ is the size of observed data. Accordingly, for $n$ items with $K$ latent factors, it takes $O(|R_o|K + NK^2)$ time. Hence, the whole-time complexity is $O(|R_o|K + (M + N)K^2)$.

Space complexity: In real implementation, we only need to store the observed data with the size of $|R_o|$ instead of the user-item interaction matrix, and the latent factor matrices $U \in \mathbb{R}^{M\times K}$ and $V \in \mathbb{R}^{N\times K}$. In the whole, the main space complexity is $O(|R_o|)$.

## IV. EXPERIMENTAL RESULTS AND ANALYSIS

### A. DATASETS

To evaluate our recommendation method with implicit feedbacks, we perform experiments on two real-world datasets: Lastfm[1] and Foursquare.[2]

#### 1) LASTFM

This dataset contains user-artist listening information from a set of users from Last.fm online music system. For the purpose of our experiments, we keep users and items with at least 10 observations and get 62376 observations from 1797 users and 1507 artists [2].

#### 2) FOURSQUARE

The Foursquare dataset was collected from Foursquare by Gao *et al.* [32] Foursquare is one of the most popular check-in datasets. For the purpose of our experiments, we keep users and items with at least 10 observations and get 85173 observations from 1925 users and 2759 locations.

### B. EVALUATION METRICS

We study the recommendation performance on various commonly used evaluation metrics, including *top-n* evaluation metrics: Pre@$n$ (Precision) [17], Rec@$n$ (Recalln) [17], and ranking-oriented evaluation metrics: NDCG@$n$ (Normalized Discounted Cumulative Gain) [32], HLU@$n$ (Half-life Utility) [2] and MRR (Mean Reciprocal Rank) [32].

Firstly, we define some notations. For item ranked list $l$, $l(k)$ represents the item located at position $k$. For each item $i$, we can also have its position $1 \le p_u(i) \le N$. $I_u^{test}$ denotes the positive (observed) item set for a specific user $u$ in the test dataset.

#### 1) Pre@n

Pre@$n$ is the average of precision over all users in the test set. Pre@$n$ $= \frac{1}{M}\sum_{u=1}^{M} \mathrm{Pre}_u(n)$, where $\mathrm{Pre}_u(n)$ is the precision (fractions of retrieved items that are preferred by the user) in a cut-off ($n$) rank list. It is defined as $\mathrm{Pre}_u(n) = \frac{1}{n}\sum_{k=1}^{n} \delta(l(k) \in I_u^{test})$, where $\delta(l(k) \in I_u^{test})$ will return 1 if the item at position $k$ is observed in the test dataset or 0 otherwise.

[1] https://grouplens.org/datasets/hetrec-2011/
[2] https://foursquare.com/

#### 2) Rec@n

Rec@$n$ is the average of recall over all users in the test set. $\text{Rec@}n = \frac{1}{M}\sum_{u=1}^{M}\text{Rec}_u(n)$. $\text{Rec}_u(n)$ for user $u$ is defined as $\text{Pre}_u(n) = \frac{1}{|I_u^{test}|}\sum_{k=1}^{n}\delta(l(k) \in I_u^{test})$, where $|I_u^{test}|$ is the number of items preferred by user $u$ in the test dataset.

#### 3) NDCG@n

It penalizes relevant items appearing lower position in a result list. It computes the average of $NDCG_u$ with a cut-off $n$ over all users in the test set. $NDCG@n = \frac{1}{M}\sum_{u=1}^{M}NDCG_u@n$, where $NDCG_u@n = \frac{1}{Z_u}DCG_u@n$ with $DCG_u@n = \sum_{k=1}^{n}\frac{2^{\delta(l(k)\in I_u^{test})}-1}{\log(k+1)}$ and $Z_u$ is the best $DCG_u@n$ score.

#### 4) HLU@n

It estimates how likely a user will view/choose an item from a ranked list, which assumes that the user will view each consecutive item in the list with an exponential decay of possibility [2]. A half-life utility over all users in a test set is defined as $\frac{\sum_u HLU_u@n}{\sum_u HLU_u^{\max}@n}$. $HLU_u@n$ is the expected utility of the cut-off ($n$) ranked list for user $u$ defined as $HLU_u@n = \sum_{k=1}^{n}\frac{\delta(l(k)\in I_u^{test})}{2^{(k-1)/(\beta-1)}}$, where $\beta$ is the half-life parameter which is set to 2. $HLU_u^{\max}$ is the maximally achievable utility if all true positive items are at the top of the ranked list.

#### 5) MRR

MRR is the average of the Reciprocal Rank (RR) across all the recommendation lists for individual users $MRR = \frac{1}{n}\sum_{u=1}^{n}RR_u$. RR measures how early in the ranked list (i.e. how highly ranked) the first relevant recommended item is ranked. $RR_u = \frac{1}{\min_{i\in I_u^{test}} p_u(i)}$, where $\min_{i\in I_u^{test}} p_u(i)$ is the rank position of the first relevant item in the estimated ranking list for user $u$.

### C. BASELINES AND PARAMETER SETTINGS

We mainly compare our method with several baselines:

#### 1) BASELINES

##### *a: MF*

The matrix factorization method is proposed by Salakhutdinov and Mnih [25], which is traditional collaborative filtering recommendation problems with explicit ratings.

##### *b: WRMF-I*

The weighted regularized matrix factorization method with item-oriented weighting scheme on unobserved data is proposed by He *et al.* [3]. It assigns the uniform confidence weights on the observed examples, while gives the lower weights on unobserved examples based on the item popularity. Since it mainly focuses on the item-oriented weighting scheme on unobserved data of WRMF, we call it WRMF-I.

##### *c: WRMF-O*

The weighted regularized matrix factorization method with frequency-based weighting scheme on observed data is proposed by Hu *et al.* [1] and Lian *et al.* [18]. It assigns uniform confidence weights on the unobserved examples, while gives the higher weights on the observed examples according to the visit frequency. Since this method only focuses on the weighting scheme on observed data, in this paper, we call it WRMF-O.

##### *d: WRMF ALS*

The ALS-based optimization scheme for weighted regularized matrix factorization method with is proposed by Pan *et al.* [2], which optimizes the weighted least square loss by iteratively updating the latent vector for a user (item), while leaving the others fixed.

##### *e: WRMF eALS [22]*

The generic element-wise ALS Learner, which optimizes parameters at the element level by optimizing each coordinate of the latent factor vector, while leaving the others fixed.

#### 2) OUR METHODS

##### *a: WRMF-U*

The model is originated from our method, which assigns non-uniform weights to unobserved data using user-oriented weighting scheme, following Equation (2).

##### *b: WRMF-UO*

The model is still originated from our method, which full consideration of weighting schemes of both observed data and unobserved data based on WRMF, and the learning procedure is shown in Algorithm 1. Still, in view of the optimization, we also call our method WRMF fast-eALS.

For fair comparison, we use the same initializations for the model variables $U \sim N(0, 0.01)$, $V \sim N(0, 0.01)$. For the maximum iteration number $\gamma$, we tried $\gamma = 100$ for all methods on different datasets. For the dimension of latent factor $U$, $V$, we use $d = 20$ [31]. For the regularization parameters, we search the best values from $\lambda \in \{0.0001, 0.001, 0.01, 0.1\}$.

### D. EFFECT OF PARAMETERS

In this subsection, we analyze how the weights of the whole data affect the final recommendation accuracy when other parameters are fixed. For the weighting scheme, we mainly have four parameters to learn for our proposed algorithm WRMF-UO. The first two parameters are associated with the weights of observed data: $\alpha$ is the coefficient parameter to is control the global weight of observed data, $\tau$ controls the level of frequent items over other not frequent ones. The other two parameters account for the weights of

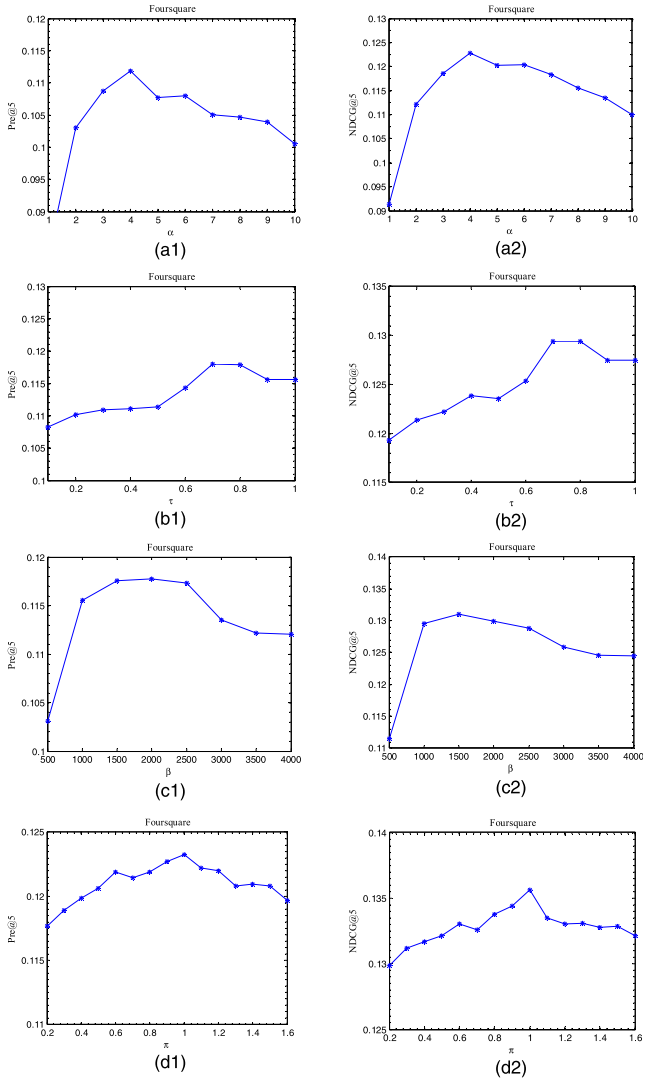


**FIGURE 1.** Effect of parameters $\alpha$, $\tau$, $\beta$, $\pi$ on Pre@5 and NDCG@5for Foursquare dataset. Fig. 1(a1) (b1) (c1) (d1) are the effects on Pre@5, Fig. 1(a2) (b2) (c2) (d2) are the effects on NDCG@5.

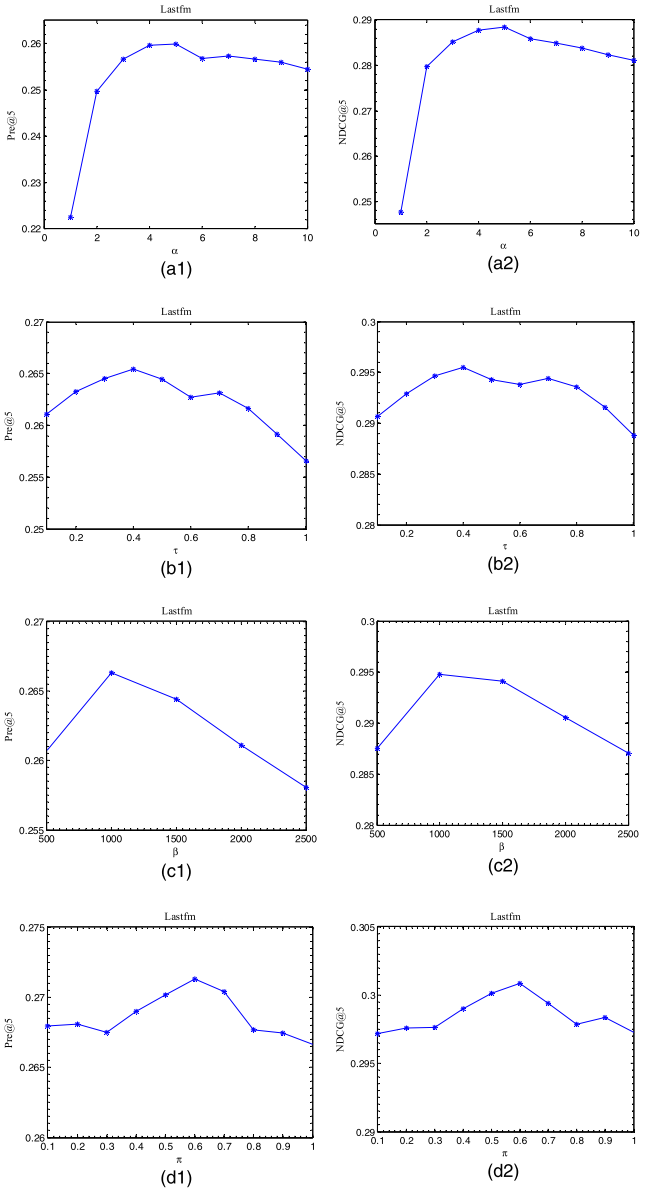


**FIGURE 2.** Effect of parameters $\alpha$, $\tau$, $\beta$, $\pi$ on Pre@5 and NDCG@5for Lastfm dataset. Fig. 2(a1) (b1) (c1) (d1) are the effects on Pre@5, Fig. 2(a2) (b2) (c2) (d2) are the effects on NDCG@5.

unobserved data: $\beta$ reflects the overall weight of unobserved data, $\pi$ controls the level of active users over other not active ones. The influences of different parameters on Foursquare dataset and Lastfm dataset are shown in Fig. 1 and Fig. 2 respectively.

We first focus on the observed weights with parameter $\alpha$ and $\tau$ on Foursquare data (Fig. 1(a1) (b1) (a2) (b2)). Before that, we fix the weights of unobserved data as 1 following [1] and [18], which means treating unobserved data equally. Then, we study how the user-oriented weighting strategy affects recommendation performance. From Fig. 1(a1) (a2) we can see that, by setting $\tau = 0$ which means the uniform weighting, the recommendation performance of WRMF-UO varies with $\alpha$. And the best performance locates at $\alpha = 4$ evaluated by both Pre@5 and NDCG@5. The more or less value of $\alpha$ would impact the performance. This result implies the importance of $\alpha$ which determines the global weight of observed data. Then we fix $\alpha = 4$, varying $\tau$ to study how does it affects the recommendation result. As can be seen from Fig. 1(b1) (b2), with the increase of $\tau \in [0, 1]$, the performance of WRMF-UO enhances gradually and drops off after $\tau$ is nearby $[0.7, 0.8]$. The optimal value of $\tau$ locates around $[0.7, 0.8]$, and performs better than that with uniform weight. This reveals the drawback of conventional weighting strategy, which overweights the frequent items. It also makes clear the effectiveness of our user-oriented weighting scheme of observed data.

Next, we fix the weighs of observed data with parameters $\alpha = 4$ and $\tau = 0.7$, and explore whether the full consideration of whole-data weights performs better than that only accounts for observed weights (Fig.1(c1) (c2) (d1) (d2)).

Specially, we first set $\pi = 0$ which signifies the uniform weighting scheme for unobserved data, followed by varying $\beta$. As is shown in Fig.1(c1) (c2), recommendation performance on Pre@5 and NDCG@5 changes with $\beta$, and gets the best results at around 1500. This result reflects the significance of overall weight of unobserved data. However, we obtain only a little gain compared with Fig.1(b1) (b2). This is because they both assign uniform weights on unobserved data, Fig.1(c1) (c2) just find the sub-optimal whole weights of unobserved data fairly with that of Fig.1(b1) (b2). Afterwards, we fix $\beta$ as 1500, and try to find how parameter $\pi$ affects recommendation performance of WRMF-UO. From Fig.1(d1) (d2) we can observe that, the performance of WRMF-UO enhances gradually with the increase of $\pi(\pi \in [0, 1])$, and performs best at around 1. This result reflects the importance of proper weighting items with active users as negative ones.

On Lastfm dataset, as is shown in Fig.2, the influence of whole-data weighting schemes presents the similar results with that on Foursquare dataset, and we will not give details here.

### E. RECOMMENDATION PERFORMANCE

In this section, we compare our method with baselines on several evaluation metrics, including top-$n$ evaluation metrics: Pre@$n$, Rec@$n$, and ranking-oriented evaluation metrics: NDCG@$n$, HLU@$n$, and MRR. Experimental comparison results are shown in Fig.3 and Fig.4 on Foursquare dataset and Lastfm dataset respectively.

From Fig.3 and Fig.4, we can have the following observations,

① As is shown in Fig.3 (a)-(b) and Fig.4 (a)-(b), our method WRMF-UO performs best among baselines in terms of top-$n$ evaluation metrics (Pre@$n$ and Rec@$n$) on both datasets. From Fig.3 (c)-(e) and Fig.4 (c)-(e), we can also see that our method WRMF-UO achieves best performance in terms of ranking-oriented evaluation metrics (NDCG@$n$, HLU@$n$ and MRR) on both dataset. The results clearly demonstrate the advantage of our method in personalized recommendation task.

② The comparison results also show that WRMF-UO achieves better performance than WRMF-O and WRMF-U, which demonstrates that WRMF-UO benefits from taking full consideration of the all-weighted scheme of the whole data. It could nicely combine the advantages of conventional methods to improve recommendation performance.

③ More specifically, in order to further investigate the effectiveness of our user-oriented weighting scheme on unobserved data, we also compare our proposed method WRMF-U with item-oriented WRMF-I. Experimental results show that WRMF-U performs slightly better than WRMF-I.

④ From Fig.3 we can also observe that WRMF-O outperforms WRMF-U and WRMF-I, which demonstrates that significance of accounting for the weighting

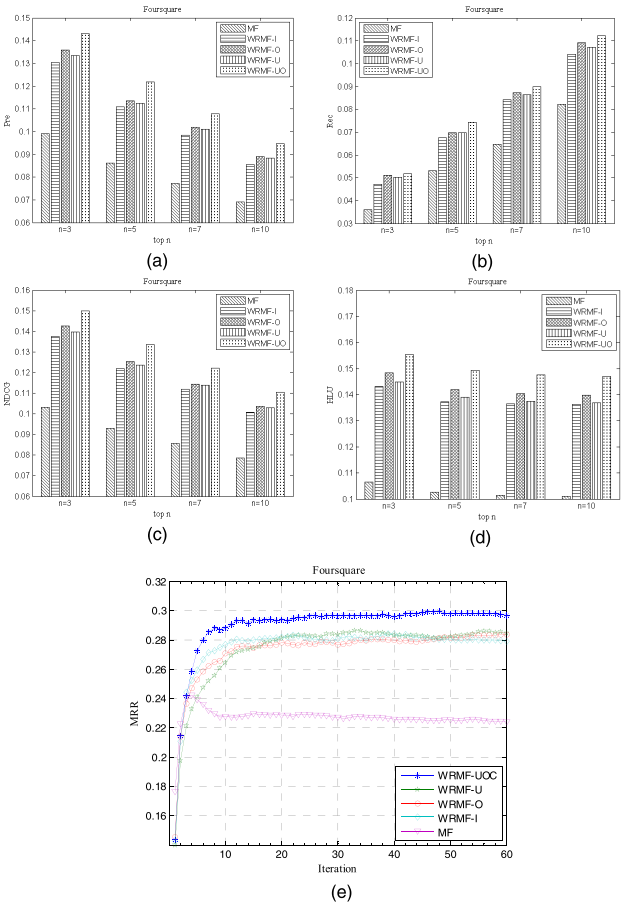


**FIGURE 3.** Recommendation performance comparison on Foursquare dataset. (a) Pre@$n$ on Foursquare dataset. (b) Rec@$n$ on Foursquare dataset. (c) NDCG@$n$ on Foursquare dataset. (d) HLU@$n$ on Foursquare dataset. (e) MRR vs. Iterations on Foursquare dataset.

scheme of observed data according to visit frequency on Foursquare dataset. Miss of this part data would badly affect recommendation accuracy. However, as is observed from Fig.4, WRMF-O performs worse than WRMF-U, but outperforms WRMF-I on Lastfm dataset. The difference indicates the importance of weights of observed data and unobserved data varies with different datasets.

⑤ In view of all the baselines, all methods beat MF, which shows the effectiveness of weighting scheme of weighted matrix factorization. Traditional MF only takes account into the observed preference and ignores the hidden information in massive unobserved data.

⑥ We also observe that the evaluation values generated by all the methods on Lastfm dataset are much higher than those on Foursquare dataset. It is mainly due to that Foursquare dataset is sparser than Lastfm dataset.

### F. ANALYSIS OF TIME COMPLEXITY

We compared the running times of the ALS, eALS and fast eALS algorithms in terms of $K$ per iteration (see Fig. 5). Observe that the running time per iteration of ALS is much larger than that of eALS, and both perform worse than our fast

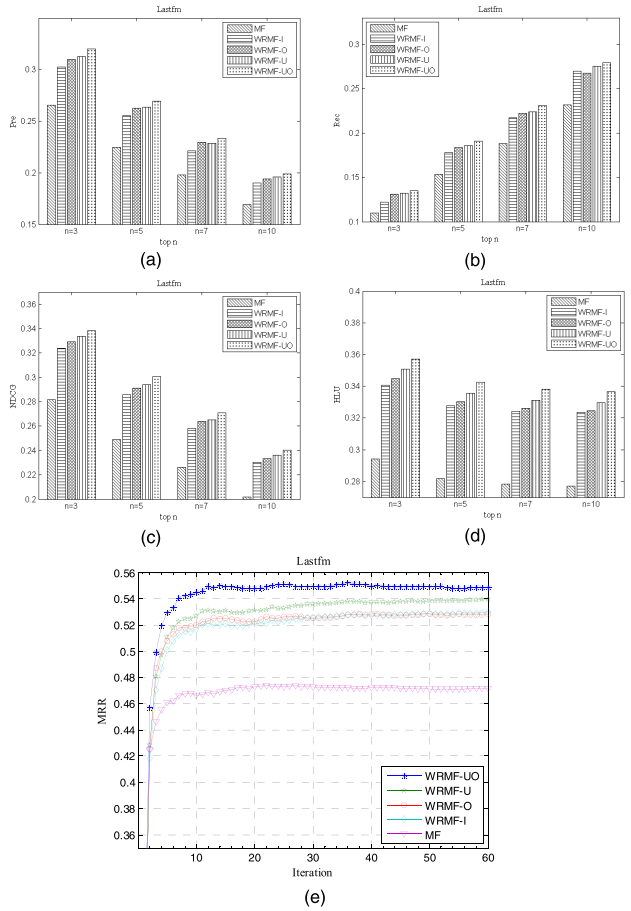


**FIGURE 4.** Recommendation performance comparison on Lastfm dataset. (a) Pre@*n* on Lastfm dataset. (b) Rec@*n* on Lastfm dataset. (c) NDCG@*n* on Lastfm dataset. (d) HLU@*n* on Lastfm dataset. (e) MRR vs. Iterations on Lastfm dataset.

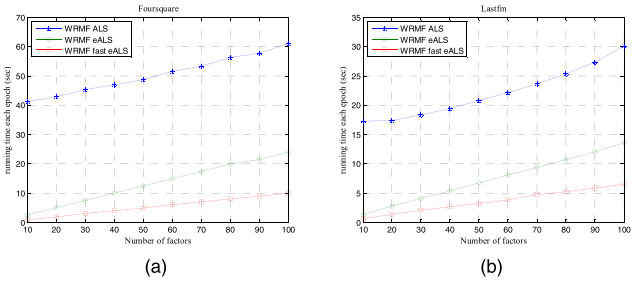


**FIGURE 5.** Learning runtime in seconds for one iteration over the whole training dataset. (a) running time vs. number of factors on Foursquare dataset. (b) running time vs. number of factors on Lastfm dataset.

eALS. In order to explain more carefully, we give the details of ALS, eALS and fast eALS respectively.

For ALS, the closed-form solution of $U$ for each user is $U_{u.} = R_{u.}\tilde{C}_{u.}V\left(V^T\tilde{C}_{u.}V + \lambda I\right)^{-1}, (\forall 1 \leq u \leq M)$, and the closed-form solution of $V$ for each item is $V_{i.} = R_{.i}^T\tilde{C}_{.i}U\left(U^T\tilde{C}_{.i}U + \lambda I\right)^{-1}, (\forall 1 \leq i \leq N)$, where $\tilde{C}_{u.}$ is a $K \times K$ diagonal matrix with the elements of $C_{u.}$ on its diagonal; I is a $K \times K$ identify matrix. From the above equations of ALS, we can observe that updating a user/item latent vector involves the inversion operation of $K \times K$ matrix, which takes $O(K^3)$ in time complexity and dominants the main part of complexity. The time complexity for updating all users and items of ALS is $O((m+n)K^3 + nmK^2)$.

For eALS, the updating formulation of each element is $U_{uk} = \sum_{i=1}^{n} C_{ui}(R_{ui} - \hat{R}_{ui}^k)V_{ik}/(\sum_{i=1}^{n} C_{ui}V_{ik}^2 + \lambda), (\forall 1 \leq u \leq M, 1 \leq k \leq K)$ for $U$, and for each element of $V$ the updating equation is $V_{ik} = \sum_{u=1}^{m} C_{ui}(R_{ui} - \hat{R}_{ui}^k)U_{uk}/(\sum_{u=1}^{m} C_{ui}U_{uk}^2 + \lambda), (\forall 1 \leq i \leq N, 1 \leq k \leq K)$. As is can be seen, eALS updates $U(V)$ element by element, nevertheless, it avoids expensive computation of matrix inversion and saves massive time. Accordingly, the time complexity for updating all users and items of eALS is $O(nmK^2)$, which is less than that of ALS. This can be observed in Fig.5.

In view of fast eALS, since it avoids massive repeated computation by pre-computation, it further lowers computation cost to $O(|R|K + (M+N)K^2)$. This is due to that the number of unobserved elements is much larger than that of observed elements, we could separate the observed part from the whole data which is independent of the context(user/item) and can be pre-computed.

## V. CONCLUSIONS

In this paper, we propose an effective and efficient recommendation framework with all-weighted scheme and fast optimization scheme. In contrast to previous work that either assigned a uniform weight on the unobserved data or applied a fixed weight on the observed data, the all-weighted scheme takes full consideration of the whole data, and assigns available confidence weights on both observed data and unobserved data. The fast optimization scheme enhances the efficiency of parameter learning significantly without sacrifice of the prediction performance. Experimental results on two real-world recommender applications demonstrate that the proposed method outperforms the competitive baselines on several prediction-oriented and ranking-oriented evaluation metrics.

For future works, we are mainly interested in extending our method in two aspects, (1) study how to leverage the auxiliary data (e.g., social network, item contents) to further improve the weighting scheme and enhance the accuracy of recommendation (2) research how to kindly integrate regression-based method with ranking-based method (i.e., Bayesian personalized ranking) in a more feasible way, making full use of the advantage of heterogeneous losses to gain better recommendation performance.

## ACKNOWLEDGMENTS

The authors would like to thank the anonymous reviewers for their constructive comments.

## REFERENCES


[1] Y. Hu, Y. Koren, and C. Volinsky, "Collaborative filtering for implicit feedback datasets," in *Proc. 8th Int. Conf. Data Mining*, Dec. 2008, pp. 263–272.

[2] R. Pan *et al.*, "One-class collaborative filtering," in *Proc. Int. Conf. Mach. Learn. Principles Pract. Knowl. Discovery Databases*, Dec. 2008, pp. 502–511.

[3] X. He, H. Zhang, M.-Y. Kan, and T.-S. Chua, "Fast matrix factorization for online recommendation with implicit feedback," in *Proc. 39th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, 2016, pp. 549–558.

[4] W. Pan, H. Zhong, C. Xu, and Z. Ming, "Adaptive Bayesian personalized ranking for heterogeneous implicit feedbacks," *Knowl.-Based Syst.*, vol. 73, pp. 173–180, Jan. 2015.

[5] I. Bayer, X. He, B. Kanagal, and S. Rendle, "A generic coordinate descent framework for learning from implicit feedback," in *Proc. 26th Int. Conf. World Wide Web*, Apr. 2017, pp. 1341–1350.

[6] M. Volkovs and G. W. Yu, "Effective latent models for binary feedback in recommender systems," in *Proc. 38th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, 2015, pp. 313–322.

[7] X. Ning and G. Karypis, "Slim: Sparse linear methods for top-n recommender systems," in *Proc. 11th Int. Conf. Data Mining*, Dec. 2011, pp. 497–506.

[8] T. Zhao, J. McAuley, and I. King, "Leveraging social connections to improve personalized ranking for collaborative filtering," in *Proc. 23rd ACM Int. Conf. Inf. Knowl. Manage.*, 2014, pp. 261–270.

[9] S. Sedhain, A. K. Menon, S. Sanner, and D. Braziunas, "On the effectiveness of linear models for one-class collaborative filtering," in *Proc. 30th Conf. Artif. Intell.*, 2016, pp. 229–235.

[10] H. Li, Y. Ge, R. Hong, and H. Zhu, "Point-of-interest recommendations: Learning potential check-ins from friends," in *Proc. 22nd Int. ACM SIGKDD Conf. Knowl. Discovery Data Mining*, 2016, pp. 975–984.

[11] L. Wu, E. Chen, Q. Liu, L. Xu, T. Bao, and L. Zhang, "Leveraging tagging for neighborhood-aware probabilistic matrix factorization," in *Proc. 21st ACM Int. Conf. Inf. Knowl. Manage.*, 2012, pp. 1854–1858.

[12] L. Du, X. Li, and Y.-D. Shen, "User graph regularized pairwise matrix factorization for item recommendation," in *Proc. 7th Int. Conf. Adv. Data Mining Appl.*, 2011, pp. 372–385.

[13] K. Verstrepen and B. Goethals, "Unifying nearest neighbors collaborative filtering," in *Proc. 8th ACM Int. Conf. Rec. Syst.*, 2014, pp. 177–184.

[14] S. Rendle and C. Freudenthaler, "Improving pairwise learning for item recommendation from implicit feedback," in *Proc. 7th ACM Int. Conf. Web Search Data Mining*, 2014, pp. 273–282.

[15] L. Yu, G. Zhou, C. Zhang, J. Huang, C. Liu, and Z. K. Zhang, "RankMBPR: Rank-aware mutual Bayesian personalized ranking for item recommendation," in *Proc. 17th Int. Conf. Web-Age Inf. Manage.*, 2016, pp. 244–256.

[16] F. Zhao and Y. Guo, "Improving top-n recommendation with heterogeneous loss," in *Proc. 25th Int. Joint Conf. Artif. Intell.*, 2016, pp. 2378–2384.

[17] Y. Shi, K. Alexandros, B. Linas, M. A. Larson, A. Hanjalic, and O. Nuria, "TFMAP: Optimizing MAP for top-n context-aware recommendation," in *Proc. 35th Annu. Int. SIGIR Conf. Res. Develop. Inf. Retr.*, 2012, pp. 155–164.

[18] D. Lian, C. Zhao, X. Xie, G. Sun, E. Chen, and Y. Rui, "GeoMF: Joint geographical modeling and matrix factorization for point-of-interest recommendation," in *Proc. 20th Int. ACM SIGKDD Conf. Knowl. Discovery Data Mining*, New York, NY, USA, 2014, pp. 831–840.

[19] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, "BPR: Bayesian personalized ranking from implicit feedback," in *Proc. 25th Conf. Uncertainty Artif. Intell.*, Jun. 2009, pp. 452–461.

[20] L. Guo, H. Jiang, X. Wang, and F. Liu, "Learning to recommend point-of-interest with the weighted Bayesian personalized ranking method in LBSNs," *Information*, vol. 8, no. 1, p. 20, 2017.

[21] T. Yuan, J. Cheng, X. Zhang, Q. Liu, and H. Lu, "A weighted one class collaborative filtering with content topic features," in *Proc. 19th Int. Conf. Multimedia Modeling*, Jan. 2013, pp. 417–427.

[22] S. Rendle, Z. Gantner, C. Freudenthaler, and L. Schmidt-Thieme, "Fast context-aware recommendations with factorization machines," in *Proc. 34th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, 2011, pp. 635–644.

[23] R. Devooght, N. Kourtellis, and A. Mantrach, "Dynamic matrix factorization with priors on unknown values," in *Proc. 21st Int. ACM SIGKDD Conf. Knowl. Discovery Data Mining*, Sydney, NSW, Australia, 2015, pp. 189–198.

[24] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *IEEE Comput.*, vol. 42, no. 8, pp. 30–37, Aug. 2009.

[25] R. R. Salakhutdinov and A. Mnih, "Probabilistic matrix factorization," in *Advances in Neural Information Processing Systems*, vol. 20. Red Hook, NY, USA: Curran Associates, 2008, pp. 1257–1264.

[26] S. Kalloori, F. Ricci, and M. Tkalcic, "Pairwise preferences based matrix factorization and nearest neighbor recommendation techniques," in *Proc. 10th ACM Conf. Rec. Syst.*, Boston, MA, USA, 2016, pp. 143–146.

[27] A. Krohn-Grimberghe, L. Drumond, C. Freudenthaler, and L. Schmidt-Thieme, "Multi-relational matrix factorization using Bayesian personalized ranking for social network data," in *Proc. 5th ACM Int. Conf. Web Search Data Mining*, 2012, pp. 173–182.

[28] I. Pilászy, D. Zibriczky, and D. Tikk, "Fast als-based matrix factorization for explicit and implicit feedback datasets," in *Proc. 4th ACM Int. Conf. Rec. Syst.*, 2010, pp. 71–78.

[29] H. Steck, "Training and testing of recommender systems on data missing not at random," in *Proc. 16th Int. ACM SIGKDD Conf. Knowl. Discovery Data Mining*, 2010, pp. 713–722.

[30] Y. Shi, A. Karatzoglou, L. Baltrunas, M. Larson, N. Oliver, and A. Hanjalic, "CLiMF: Learning to maximize reciprocal rank with collaborative less-is-more filtering," in *Proc. 6th ACM Int. Conf. Rec. Syst.*, 2012, pp. 139–146.

[31] W. Pan and L. Chen, "CoFiSet: Collaborative Filtering via Learning Pairwise Preferences over Item-sets," in *Proc. 6th ACM Int. Conf. Web Search Data Mining*, 2013, pp. 180–188.

[32] H. Gao, J. Tang, X. Hu, and H. Liu, "Exploring temporal effects for location recommendation on location-based social networks," in *Proc. ACM Conf. Recommender Syst.*, 2013, pp. 93–100.


**HONGMEI LI** (F'90) was born in Hengshui, Hebei, China, in 1990. She received the master's degree from PLA Army Engineering University, Nanjing, in 2015, where she is currently pursuing the Ph.D. degree.

Since 2008, she has been a Research Student with PLA Army Engineering University. She has co-authored two books and over ten articles. She holds one authorized software copyright. Her research interests include collaborative filtering recommendation, data analysis and mining, and natural language processing.

Dr. Li was a recipient of First Prize in the National Postgraduate Mathematical Modeling Contest in 2012.

**XINGCHUN DIAO** (M'64) was born in Taixing, Jiangsu, China, in 1964. He received the master's degree from the PLA National University of Defense Technology, Changsha.

He is currently a Researcher and a Ph.D. Supervisor. He is involved in the research of data quality control, data analysis, and mining for a long time. He is one of the co-sponsors of the Information Quality Research Group, China. He has authored three data quality translations and over 90 academic papers at important conferences and journals. He holds two authorized invention patents and two authorized software copyrights.

Prof. Diao holds six invention patents. He received ten provincial science and technology progress prizes. He enjoys the special allowance experts of the State Council. He has successively presided over a number of important scientific research projects.

**JIANJUN CAO** (M'75) was born in Yuncheng, Shandong, China, in 1975. He received the Ph.D. degree from the Ordnance Engineering College, Shijiazhuang, in 2008.

He is currently an Associate Research Fellow and a Master Supervisor. He is mainly involved in data quality control, data governance, data intelligence analysis, and application. He has presided over seven provincial key projects. He has authored over 80 academic papers up to now. He holds four authorized invention patents and four authorized software copyrights.

Dr. Cao was a recipient of the National Excellent Doctoral Dissertation Nomination Award. He received the China Postdoctoral Science Foundation Special Fund. He received four provincial and ministerial awards.

**QIBIN ZHENG** (M'91) was born in Lanzhou, Gansu, China, in 1991. He received the master's degree from PLA Army Engineering University, Nanjing, in 2016, where he is currently pursuing the Ph.D. degree.

Since 2008, he has been a Research Student with PLA Army Engineering University. He has co-authored three books and has authored over ten articles. He holds one authorized software copyright. His research interests include entity resolution, data mining, and data quality.