WEMAREC: Accurate and Scalable Recommendation through Weighted and Ensemble Matrix Approximation

Chao Chen[§,*], Dongsheng Li[†], Yingying Zhao[§], Qin Lv[‡], Li Shang[‡]
[§]Tongji University, Shanghai, 201804, P.R. China
[†]IBM Research - China, Shanghai, 201203, P.R. China
[‡]University of Colorado Boulder, Boulder, CO 80309, USA
{chench.resch, yyzhao.tj}@gmail.com, ldsli@cn.ibm.com, {qin.lv, li.shang}@colorado.edu

ABSTRACT

Matrix approximation is one of the most effective methods for collaborative filtering-based recommender systems. However, the high computation complexity of matrix factorization on large datasets limits its scalability. Prior solutions have adopted co-clustering methods to partition a large matrix into a set of smaller submatrices, which can then be processed in parallel to improve scalability. The drawback is that the recommendation accuracy is lower as the submatrices only contain subsets of the user-item rating information.

This paper presents WEMAREC, a weighted and ensemble matrix approximation method for accurate and scalable recommendation. It builds upon the intuition that (sub)matrices containing more frequent samples of certain user/item/rating tend to make more reliable rating predictions for these specific user/item/rating. WEMAREC consists of two important components: (1) a *weighting* strategy that is computed based on the rating distribution in each submatrix and applied to approximate a single matrix containing those submatrices; and (2) an *ensemble* strategy that leverages user-specific and item-specific rating distributions to combine the approximation matrices of multiple sets of co-clustering results. Evaluations using real-world datasets demonstrate that WEMAREC outperforms state-of-the-art matrix approximation methods in recommendation accuracy (0.5–11.9% on the MovieLens dataset and 2.2–13.1% on the Netflix dataset) with 3–10X improvement on scalability.

Categories and Subject Descriptors

H.3.4 [**Systems and Software**]: User profiles and alert services

Keywords

R ecommendation; matrix approximation; weighted; ensemble

*Chao Chen and Dongsheng Li contributed equally to this work.

SIGIR'15, August 09 - 13, 2015, Santiago, Chile.

DOI: http://dx.doi.org/10.1145/2766462.2767718 .

1. INTRODUCTION

Collaborative filtering (CF), which predicts users' item ratings based on the ratings of other users with similar taste, has been shown to preform well in many recommender systems [1]. Among existing CF solutions, matrix approximation has become increasingly popular. It formulates the recommendation problem as missing entry prediction using existing entries in a user-item rating matrix, i.e., attempting to predict the missing entries in a partially observed matrix. Given m users and n items, the user-item rating matrix $M \in \mathbb{R}^{m \times n}$ is typically of low-rank, then M can be approximated by a r-rank matrix $\hat{M} = UV^T$, where U $\in \mathbb{R}^{m \times r}$ is the set of user features, V $\in \mathbb{R}^{n \times r}$ is the set of item features, and $r \ll \min(m, n)$. Then, the rating of the i-th user on the j-th item can be predicted by the inner product $U_i V_j^T$. Using matrix approximation, the user/item feature vectors are reduced to lower dimensions, which helps to address the "data sparsity" issue, a challenge to memory-based CF methods [1, 24]. Recent studies have shown that matrix approximation based CF methods outperform many other CF solutions [10, 17, 22, 28].

However, existing matrix approximation based CF methods exhibit poor scalability due to the high computation complexity of matrix factorization on large user-item rating datasets [10, 22, 16, 11, 28]. Recent work adopted co-clustering methods [8, 29, 26] to partition the large user-item rating matrix into a set of smaller submatrices, which can then be processed in parallel to improve system scalability. However, this usually leads to lower recommendation accuracy. Co-clustering tries to find coherent submatrices each of which contains a subset of users who share similar interests on a subset of items. In the ideal case, recommendations based on such submatrices can be as accurate as recommendations based on the original large matrix while requiring much less computation overhead. However, the submatrices obtained by co-clustering methods are not perfect, as a small fraction of user-item ratings may not follow the distribution of majority ratings. As a result, recommendation accuracy on such user-item ratings will degrade, affecting the overall recommendation accuracy. Our study shows that, for the MovieLens dataset with 1 million ratings, the recommendation RMSE (root mean square error) increases from 0.8645 to 0.9 when the co-clustering setting varies from 1×1 to 5×5. Therefore, a better matrix approximation solution that achieves both high accuracy and high scalability for recommendation is needed.

In this work, we have developed WEMAREC, a weighted and ensemble matrix approximation method for accurate and scalable CF-based recommendation. The intuition is

that, (sub)matrices only contain partially-sampled information of user/item/rating; if a submatrix contains more samples for a certain user or item or rating, this submatrix can probably make more reliable predictions on the specific user or item or rating. This applies not only to the submatrices generated from a single co-clustering process, but also to the multiple sets of submatrices generated using different co-clustering constraints. Since these submatrices contain different user-item rating information, an intelligent combination of these submatrices can yield better recommendation quality while still enjoy the benefit of high scalability due to parallel processing the co-clustering submatrices.

This work makes the following contributions: (1) identification of the unbalanced predication power on different users/items/ratings due to the partial information that is contained in (sub)matrices; (2) development of a submatrix-based weighting strategy to capture rating-specific prediction power and combine submatrices into the approximation of a single user-item rating matrix; (3) development of an ensemble matrix approximation method that uses different co-clustering constraints to generate and combine multiple sets of submatrices with different rating prediction power for different users and items; and (4) evaluation using two large-scale real-world datasets which demonstrates WEMAREC's improvement in recommendation accuracy and scalability over state-of-the-art matrix approximation techniques.

The rest of this paper is organized as follows. Section 2 formulates the problem. Section 3 describes the proposed WEMAREC method in detail. Section 4 analyzes the error bounds of the proposed method. Section 5 presents the evaluation results. Section 6 discusses the related work, and finally Section 7 concludes this work.

2. PROBLEM FORMULATION

This section provides necessary background for the matrix approximation problem. It then presents case studies to motivate the challenge faced by existing matrix approximation methods. The case studies presented in this section are conducted on the MovieLens (1M) dataset and the standard singular value decomposition (SVD) algorithm is used in matrix approximation.

2.1 Notations and Definitions

In this paper, upper case letters such as M, U, V denote matrices. For matrix $M \in \mathbb{R}^{m\times n}$, we denote M_{i*} as the i-th row vector, M_{*j} as j-th column vector, and M_{ij} as the entry in the i-th row and j-th column. We denote $\mathcal{M}$ as a submatrix of M, i.e., both the rows and columns in $\mathcal{M}$ are subsets of those in M. A r-rank approximation of M is denoted as $\hat{M} = UV^T$, where $U \in \mathbb{R}^{m\times r}, V \in \mathbb{R}^{n\times r}$ and $r \ll \min(m,n)$. In addition, $[n]$ denotes the list of $\{1,\ldots,n\}$, Ω denotes the set of observed entries in the user-item rating matrix M, i.e., $\forall(i,j) \in \Omega$, $M_{i,j} \neq 0$. Then, $|\Omega|$ is denoted as the total number of observed entries in M.

Three matrix norms are used in this paper. The Frobenius norm is denoted as:

$$||M||_F := \sqrt{\sum_{ij} M_{ij}^2}.$$

The nuclear norm is denoted as the sum of singular values:

$$||M||_* := \sum_k \sigma_k.$$

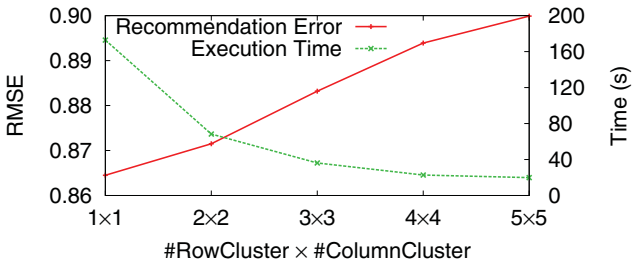

Figure 1: The tradeoff between recommendation accuracy and runtime efficiency when varying the number of co-clusters.

The max norm is denoted as:

$$||M||_\infty = \max\{|M_{ij}|\}.$$

2.2 Existing Low-Rank Matrix Approximation

Two methods have been used for low-rank matrix approximation $\hat{M}$ of M, i.e., SVD and compressed sensing [5, 6]. The SVD method is based on minimizing the sum-squared distance — Frobenius norm:

$$\hat{M} = \arg\min_X \| I \otimes (M - X) \|_F \quad s.t. \;\; rank(X) = r, \quad (1)$$

where each I_{ij} is the indicator function that equals to 1 if M_{ij} is observed and equals to 0 otherwise. The compressed sensing method is based on minimizing the nuclear norm:

$$\hat{M} = \arg\min_X \| X \|_* \quad s.t. \quad \| I \otimes (M - X) \|_F < \epsilon. \quad (2)$$

As shown in [22], the problem defined in (1) is a difficult non-convex optimization problem and an iterative method may converge to a local minimum. In contrast to SVD, the problem defined in (2) is convex and can be casted as a semi-definite program [6].

Rating	Distribution	RMSE (w/o weighting)	RMSE (w/ weighting)
1	17.44%	1.2512	1.2533
2	25.39%	0.6750	**0.6651**
3	35.35%	0.5260	**0.5162**
4	18.28%	1.1856	**1.1793**
5	3.50%	2.1477	2.1597
Overall result		0.9517	**0.9479**

Table 1: Rating-specific RMSE when running SVD without (w/o) or with (w/) weighting on a submatrix.

2.3 Motivating Examples

Next, we present case studies to demonstrate the accuracy issue of existing co-clustering based matrix approximation methods, and provide insights on why user-item rating distribution can be leveraged to improve the recommendation accuracy.

2.3.1 *Scalability vs. Accuracy*

Co-clustering is an effective method to improve the scalability of matrix approximation based CF methods [8, 29], because these submatrices can be processed in parallel. In

addition, co-clustering tries to find coherent submatrices, each of which containing a subset of users who share similar interests on a subset of items. In the ideal case, users' common interests in each submatrix can be accurately predicted. Unfortunately, the submatrices obtained by co-clustering are not perfect. Within each submatrix, a subset of user-item ratings may not follow the distribution of majority ratings. As a result, the corresponding recommendation accuracy of those minority user-item ratings may be poor, which in turn affects the overall recommendation accuracy.

To evaluate how the recommendation accuracy varies with co-clustering granularity, we apply Bregman co-clustering [2] on the MovieLens dataset. As demonstrated in Figure 1, assuming the co-clustering submatrices are processed in parallel, the scalability of matrix approximation increases when the number of co-clusters increase from 1×1 to 5×5 ($k \times k$). On the other hand, the recommendation error (measured as RMSE) increases from 0.8645 to 0.9 as the number of co-clusters increases. Therefore, in order to utilize co-clustering based scalable matrix approximation methods in recommender systems, one must address the accuracy issue.

2.3.2 Accuracy vs. Rating Distribution

Although the submatrices obtained through co-clustering are not informative enough to build accurate recommendation models for all users and all items in each submatrix, they can still be utilized to build "weak" recommendation models which can accurately predict the common interests shared by users in the same submatrix. In this study, we analyze (1) which part of information in each submatrix represents users' common interests and thus can be utilized to build "weak" recommendation models and (2) how we can make these "weak" recommendation models more accurate when predicting users' common interests in a submatrix.

Table 1 shows the distribution of different ratings in a submatrix obtained from Bregman co-clustering, as well as the recommendation quality when applying standard SVD algorithm on the submatrix. As shown in the third column, the RMSE varies by the specific rating and lower RMSEs (i.e., better recommendation accuracy) are achieved for ratings that occur more frequently in the submatrix, such as 3 and 2 ratings. This makes sense because a learning model usually does a better job capturing samples that occur more frequently in the train data. Based on this observation, we can train "weak" recommendation models from the submatrices, which can at least make accurate recommendations on users' common interests, i.e., ratings that occur most frequently in the corresponding submatrix.

Given the biased prediction power of the "weak" models towards ratings that occur more frequently in a submatrix, we could potentially boost the recommendation accuracy by weighting the user-item ratings differently, assigning higher (lower) weights to ratings that occur more frequently (rarely) in the submatrix. The expectation is that we could obtain more accurate recommendations on the majority of the ratings and the overall recommendation accuracy can be improved. The fourth column in Table 1 shows the recommendation quality after adding such weights to differentiate the user-item ratings. Clearly, we can see that the "weak" recommendation model built by SVD with weighting can indeed make more accurate recommendations on the ratings that occur more frequently (i.e., 2, 3 and 4) than the SVD method without weighting. More importantly, the overall recommendation quality also increases after weighting,

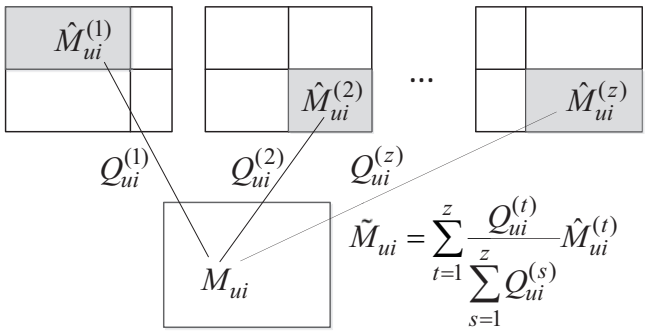

Figure 2: WEMAREC design overview. The original user-item rating matrix M is described by z low-rank matrices, which are based on z different $k \times l$ co-clustering settings. For all pairs $(u, i) \in [m] \times [n]$, the entry M_{ui} is computed based the corresponding entries in the co-cluster-based submatrices (denoted as shaded regions). The equation describes how to compute a unified matrix approximation $\tilde{M}$ from z co-clustering based approximations $\{\hat{M}_1, \hat{M}_2, \ldots, \hat{M}_z\}$.

i.e., RMSE decreases from 0.9517 to 0.9479. These results demonstrate that rating-specific weighting has the potential to boost the accuracy of more frequently-occurring ratings and enhance the overall accuracy as well.

3. WEMAREC ALGORITHM DESIGN

In this section, we present the design details of WEMAREC which can achieve both high recommendation accuracy and high scalability for matrix approximation based CF methods. As illustrated in Figure 2, WEMAREC consists of three key steps:

1. **Co-clustering and submatrices generation.** The original user-item rating matrix is first divided into a set of submatrices by Bregman co-clustering, so that the scalability issue can be addressed by factoring all submatrices in parallel. Also, different co-clusterings can be obtained by varying the constraints in Bregman co-clustering, which naturally offers us the ability to exploit the advantages of different co-clusterings to achieve better recommendation accuracy.

2. **Submatrices-based weighting and matrix approximation.** A new weighting strategy is proposed, which is computed based on each individual submatrix and assigns higher weights to ratings that occur more frequently in a given submatrix. The weighted submatrices from the same co-clustering setting are then used to generate a single matrix approximation.

3. **Ensemble of multiple matrix approximations.** Different co-clustering constraints can lead to different submatrices and thus different matrix approximations. Since each "weak" recommendation model can only make accurate recommendations on some of the user-item ratings, an ensemble strategy is proposed, which utilizes the advantages of different "weak" models to realize a "strong" recommendation model which can achieve high accuracy.

3.1 Co-clustering & Submatrices Generation

Co-clustering is a popular technique which allows simul-

taneous clustering of both the rows and columns in a given matrix. By applying co-clustering methods on user-item rating matrix in recommender systems, users and items correspond to a co-cluster (submatrix) are highly correlated, i.e., these users will have similar opinions on these items. More formally, let submatrix $\mathcal{M} = \{M_{ui} \mid u \in \mathcal{U}, i \in \mathcal{I}\}$ denote the ratings of a subset of users $\mathcal{U}$ on a subset of items $\mathcal{I}$. If we properly choose a set of very similar users $\mathcal{U}$ and a set of very similar items $\mathcal{I}$, then $\mathcal{M}$ can be reconstructed by fewer number of parameters, i.e., lower rank. Such co-clustering can be beneficial in two aspects: 1) these submatrices can be approximated simultaneously via parallel computing so that high scalability can be achieved and 2) each submatrix will have lower rank than original user-item rating matrix, so that low-rank matrix approximation can be computed more efficiently for each submatrix.

In order to find such coherent submatrices in the user-item rating matrix, we consider to simultaneously partition all users and items into disjoint user clusters $\{\mathcal{U}_1, \ldots, \mathcal{U}_k\}$ and item clusters $\{\mathcal{I}_1, \ldots, \mathcal{I}_l\}$, and a co-cluster $(\mathcal{U}, \mathcal{I})$ corresponds to one desired submatrix. Bregman co-clustering [2] is adopted in this paper to achieve such partitioning. It views the co-clustering as a lossy data compression problem, and attempts to obtain as much information as possible about the original matrix with a few number of critical statistics for co-clusters, such as the row and column averages of each co-cluster. Following common approaches in Bregman co-clustering, we should firstly choose a set of statistics of original matrix that need to be preserved, e.g., the average rating of each user or the average rating of each item, etc., and each statistic can be viewed as a constraint. We denote $\mathcal{C}$ as the constraint set, and six of the most popular non-trivial constraint sets are described as follows:

$$\mathcal{C}_1 = \{\{I_r\}, \{I_c\}\},\ \mathcal{C}_2 = \{\{\hat{I}_r, \hat{I}_c\}\},$$
$$\mathcal{C}_3 = \{\{\hat{I}_r, \hat{I}_c\}, \{I_r\}\},\ \mathcal{C}_4 = \{\{\hat{I}_r, \hat{I}_c\}, \{I_c\}\},$$
$$\mathcal{C}_5 = \{\{\hat{I}_r, \hat{I}_c\}, \{I_r\}, \{I_c\}\},\ \mathcal{C}_6 = \{\{I_r, \hat{I}_c\}, \{\hat{I}_r, I_c\}\},$$

where I_r and I_c are the random variables of row and column indices, which take values over $\{1, \ldots, m\}$ and $\{1, \ldots, n\}$, respectively. $\hat{I}_r$ and $\hat{I}_c$ are the random variables of row and column clusters, which take values over $\{1, \ldots, k\}$ and $\{1, \ldots, l\}$ in a $k \times l$ co-clustering. More specifically, $\mathcal{C}_1$ means that the average values of each row and each column should be preserved, $\mathcal{C}_2$ means the average values of all entries inside each co-cluster should be preserved, and similarly for the other constraint sets. Besides the constraints, we also need to select appropriate Bregman divergence to evaluate a co-clustering, which is defined as follows: for $z_1, z_2 \in \mathbb{R}$, $d_\phi(z_1, z_2) = \phi(z_1) - \phi(z_2) - < z_1 - z_2, \bigtriangledown\phi(z_2) >$, where $\bigtriangledown\phi$ is the gradient of differential function ϕ. Two popular Bregman divergences are I-divergence and Squared Euclidean distance, defined respectively as follows:

$$d_\phi(z_1, z_2) = z_1 \log(z_1/z_2) - (z_1 - z_2), \phi(z) = z \log z \quad (3)$$
$$d_\phi(z_1, z_2) = (z_1 - z_2)^2, \phi(z) = z^2 \quad (4)$$

Finally, the row and column clustering will be achieved by an iterative meta algorithm, in which the row and column cluster updates can be obtained from the optimal Lagrange multipliers in parallel. The details of the meta algorithms for achieving Bregman co-clustering can be found in [7, 8]. Since the meta algorithms are not the contributions of this paper, details of these algorithms are omitted.

3.2 Submatrices-based Weighting and Matrix Approximation

As described earlier, performing standard matrix approximation on submatrices will not be accurate due to the fact that each submatrix only holds partial information of users/items/ratings. Therefore, we can only learn a "weak" recommendation model from each submatrix, which can make accurate recommendation on a majority of ratings in the corresponding submatrix. As demonstrated in the motivating examples (Section 2.3.2), by associating higher weights to ratings that occur more frequently in a given submatrix, we could potentially improve not only the recommendation accuracy for the frequent ratings, but also the overall recommendation accuracy.

Based on the idea above, we propose a new method for low-rank matrix approximation, which weighs the individual user-item rating differently based on the rating distribution in a submatrix. Specifically, we compute the probabilistic distribution of different ratings in each submatrix and construct a weighted norm by adding the probabilistic information. Since the weight is a function of the rating distribution, we can construct the weighting function $p(x) : \mathbb{F} \to \mathbb{R}$ with Taylor's formula as follows:

$$p(x) = f(\Pr[x]) \quad (5)$$
$$= C_0 + C_1 \Pr[x] + r_p \quad (6)$$
$$\approx 1 + \beta_0 \Pr[x] \quad (7)$$

Since $\Pr[x] \in [0, 1]$, the residual term r_p, which is super linear to $\Pr[x]$, can be omitted. Without loss of generality, we can assume that $C_0 = 1$ by scaling all the parameters, then there is only one unknown parameter β_0 in the weighting function. The value of β_0 should be trained such that optimal recommendation accuracy can be achieved. However, this is not straightforward, because recommendation is one-step further after matrix approximation. Therefore, we choose the optimal β_0 by brute force search, in which we check every β_0 value in the linear function $p(x)$. The sensitivity analysis of β_0 is presented in Section 5.2.

After defining the weighting function $p(x)$, we present the extended SVD and compressed sensing matrix approximation methods here, in which weight W_{ij} is $p(M_{ij})$ if the entry is observed and 0 otherwise. It should be noted that the standard low-rank matrix approximation methods, such as SVD, can be regarded as special cases of the proposed method by setting $\beta_0 = 0$.

Extension of SVD :

$$\hat{M} = \arg\min_X \ ||W \otimes (M - X)||_F \ \ s.t.\ \ rank(X) = r. \quad (8)$$

Extension of Compressed Sensing :

$$\hat{M} = \arg\min_X \ ||X||_* \ \ s.t.\ \ ||W \otimes (M - X)||_F < \epsilon. \quad (9)$$

The two optimization problems describe how to estimate $\hat{\mathcal{M}}$ from observed entries in an original submatrix $\mathcal{M}$. Then, missing entries in the original submatrix $\mathcal{M}$ can be obtained from $\hat{\mathcal{M}}$, i.e., user ratings on unrated items can be predicted from $\hat{\mathcal{M}}$ as in other matrix approximation methods.

After applying Bregman co-clustering on the user-item rating matrix M, a $k \times l$ co-clustering (k is the number of user clusters and l is the number of item clusters) can be obtained. Then, we can perform the proposed low-rank matrix approximation on each submatrix. Here, we present a gradient decent learning algorithm for approximating the

user-item rating matrix based on the $k \times l$ co-clustering. As described in Algorithm 1, the weight $p(x)$ for each entry in each submatrix is first computed. Then, the proposed low-rank matrix approximation is achieved by a gradient decent method with L_2 regularization. At last, we combine all the resulting submatrix approximations, so that the approximated matrix $\hat{M}$ can be obtained by re-locating each entry in the $k \times l$ submatrices.

Algorithm 1 Co-clustering-based Matrix Approximation

Input: All co-clustering submatrices $\mathcal{M}^{(t)} \subseteq M$ $(t \in [kl])$, rank r, learning rate v, regularization coefficient λ.
Output: Approximated user-item rating matrix $\hat{M}$.

for each $t \in \{1, \ldots, kl\}$ **in parallel do**
  // Computing weights
  Compute the rating distribution on $\mathbb{F}$ in $\mathcal{M}^{(t)}$.
  for each observed entry (u, i) in $\mathcal{M}^{(t)}$ **do**
    $W_{ui} = p(x)$, if $M_{ui} = x$.
  end for
  // Updating model
  Initialize $U^{(t)} \in \mathbb{R}^{m \times r}, V^{(t)} \in \mathbb{R}^{n \times r}$ randomly
  while not converged **do**
    for each observed entry (u, i) in $\mathcal{M}^{(t)}$ **do**
      $\triangle_{ui} = \mathcal{M}^{(t)}_{ui} - U^{(t)}_i (V^{(t)}_j)^T$
      for each $z \in \{1, \ldots, r\}$ **do**
        $U^{(t)}_{uz} = U^{(t)}_{uz} + v * (\triangle_{ui} * V^{(t)}_{iz} * W_{ui} - \lambda U^{(t)}_{uz})$
        $V^{(t)}_{iz} = V^{(t)}_{iz} + v * (\triangle_{ui} * U^{(t)}_{uz} * W_{ui} - \lambda V^{(t)}_{iz})$
      end for
    end for
  end while
end for
for each $(u, i) \in [m] \times [n]$ **do**
  Locate (u, i) in its corresponding submatrix and let the index of the submatrix be ξ.
  $\hat{M}_{ui} = U^{(\xi)}_u (V^{(\xi)}_i)^T$
end for
return $\hat{M}$

3.3 Ensemble of Multiple Matrix Approximations

As described in Section 3.1, a Bregman co-clustering consists of three components: a constraint set $\mathcal{C}$, a Bregman divergence d_ϕ, and the number of row clusters k and column clusters l. For convenience, we denote a 3-tuple $(\mathcal{C}, d_\phi, k \times l)$ as a co-clustering. It should be noted that different 3-tuples $(\mathcal{C}, d_\phi, k \times l)$ could lead to different matrix approximation results $\hat{M}$ because each entry in $\hat{M}$ is generated based on the corresponding co-cluster. In Section 3.1, we described six constraint sets and two Bregman divergences, which means that we can construct $6 \times 2 = 12$ different $\hat{M}$s by combining different $\mathcal{C}$ and d_ϕ given fixed k and l. Therefore, we propose an ensemble method to intelligently combine the user-item rating predictions obtained from multiple matrix approximations based on different co-clustering settings. Our goal is to further enhance the recommendation accuracy.

To recover a global approximation $\tilde{M}$ from z low-rank approximations $\hat{M}^{(t)}$ $(t \in [z])$, we adopt the weighted mean of $\hat{M}^{(t)}_{ij}$ as $\tilde{M}_{ij}$. The weight for the ensemble method should be determined by the confidences of both users and items. And a predicted rating from a "weak" recommendation model should be considered as more important if 1) the corresponding user frequently gave such rating to items before and 2) the corresponding item was frequently rated by such rating before. More formally, the i-th user (M_{i*}) and the j-th item (M_{*j}) can be viewed as discrete random variables over a finite-field $\mathbb{F}$ with unique distributions $\Pr(x; M_{i*})$ and $\Pr(x; M_{*j})$, $x \in \mathbb{F}$. Furthermore, the prediction of $\mathcal{M}_{ij} = x$ should be considered more confident if user i and item j have (been) rated x many times in M. Therefore, the ensemble weight $q(x) : \mathbb{F} \to \mathbb{R}$ can be regarded as a function of $\Pr(x; M_{i*})$ and $\Pr(x; M_{*j})$ as follows:

$$q(x) = f(\Pr[x; M_{i*}], \Pr[x; M_{*j}]) \quad (10)$$
$$= C_0 + C_1 \Pr[x; M_{i*}] + C_2 \Pr[x; M_{*j}] + r_q \quad (11)$$
$$\approx 1 + \beta_1 \Pr[x; M_{i*}] + \beta_2 \Pr[x; M_{*j}] \quad (12)$$

Again, by omit the small residual term r_q and scaling the constant term to 1, we can obtain the final ensemble weight as Equation 12. Then, the proposed ensemble method can be performed as follows:

$$\tilde{M}_{ui} = \sum_{t=1}^{z} \frac{Q^{(t)}_{ui}}{\sum_{s=1}^{z} Q^{(s)}_{ui}} \hat{M}^{(t)}_{ui}, \quad (13)$$

where $Q^{(t)}_{ui} = q(M_{ui})$. Based on Equation 13, we present Algorithm 2 to describe how to predict missing values in user-item rating matrix for recommendation. We can clearly see that the global matrix approximation $\tilde{M}$ can be efficiently computed because the computation for all entries in $\tilde{M}$ just requires weighted averaging.

Algorithm 2 WEMAREC_Ensemble (u, i)

Input: Resulting matrix approximations $\hat{M}^{(t)}$ $(t \in [z])$ from z different co-clusterings, u and i are the targeted user and item, respectively.
Output: The predicted rating of user u on item i: $\tilde{M}_{ui}$.

// Computing weights
for $t \in [z]$ **do**
  $Q^{(t)}_{ui} = q(\hat{M}^{(t)}_{ui})$
end for
return $\tilde{M}_{ui} = \sum_{t=1}^{z} \frac{Q^{(t)}_{ui}}{\sum_{s=1}^{z} Q^{(s)}_{ui}} \hat{M}^{(t)}_{ui}$

3.4 Running Time Analysis

The proposed weighted and ensemble matrix approximation method (WEMAREC) is faster than many state-of-the-art matrix approximation algorithms, although its overall computational complexity is nearly z times larger than solving a regularized SVD problem. The reasons why the proposed WEMAREC method can run faster are as follow: (1) Every co-cluster is independent from each other, and matrix approximation on each co-cluster can be computed in parallel; (2) standard low-rank algorithms have a computation complexity of $\Omega(rmn)$ per-iteration, whereas the proposed WEMAREC method significantly reduces the computation complexity to $\Omega(r|\mathcal{U}||\mathcal{I}|)$ per-iteration because user clusters and item clusters are not overlapping in Bregman co-clustering; and (3) the users inside each co-cluster are highly similar, and so are the items. Therefore, lower rank (r) is required to achieve accurate matrix approximation in the proposed method than other methods, which further reduces its running time. Besides theoretical analysis, we also analyze the scalability of the proposed WEMAREC method in Section 5.

4. ERROR BOUND ANALYSIS

This section analyzes the generalization error bounds of the proposed method. We use the root mean squared error (RMSE), one of the most widely adopted accuracy measures in recommender systems [1], as the evaluation metric:

$$\mathcal{D}(\hat{\mathrm{M}}) = \sqrt{\frac{1}{mn}\sum_{u=1}^{m}\sum_{i=1}^{n}(\hat{\mathrm{M}}_{ui} - \mathrm{M}_{ui})^2}$$

where $M \in \mathbb{F}^{m\times n}$ and $\Pr[\max(\mathbb{F}) \geq \hat{M}_{ui} \geq \min(\mathbb{F})] = 1$. Then, the following proposition establishes the error bound of the proposed weighted matrix approximation method, i.e., the RMSE of the weighted low-rank matrix approximation method on each co-cluster is bounded, so that we can still find optimum submatrix factorization for recommendation by optimizing the extended optimization problems (Equation 8 and 9).

PROPOSITION 1. *For any $M \in \mathbb{F}^{m\times n}$, $m, n > 2$, $\delta > 0$, with probability at least $1-\delta$ over choosing a subset Ω of entries in M uniformly,*

$$\mathcal{D}(\hat{M}) \leq \mathcal{D}_\Omega(\hat{M}) + \sqrt{\frac{\log\delta}{-2|\Omega|}(\max(\mathbb{F}) - \min(\mathbb{F}))^2}.$$

PROOF. Since the entries of Ω are chosen independently and uniformly, it is reasonable to assume each $loss(\hat{M}_{ui}; M_{ui}) = (M_{ui} - \hat{M}_{ui})^2$ is a random variable and satisfies

$$\Pr[\alpha^2 \geq loss(\hat{M}_{ui}; M_{ui}) \geq 0] = 1$$

which $\alpha = \max(\mathbb{F}) - \min(\mathbb{F})$. Hence, based on the Hoeffding Inequality, we have $\Pr[\mathcal{D}(\hat{M}) - \mathcal{D}_\Omega(\hat{M}) \geq \epsilon] \leq e^{\frac{-2|\Omega|\epsilon^2}{\alpha^2}}$. By setting $\epsilon = \sqrt{\frac{\log\delta}{-2|\Omega|}\alpha^2}$, we have

$$\Pr[\mathcal{D}(\hat{\mathrm{M}}) - \mathcal{D}_\Omega(\hat{\mathrm{M}}) \leq \sqrt{\frac{\log\delta}{-2|\Omega|}\alpha^2}] \geq 1-\delta$$

i.e.,

$$\Pr[\mathcal{D}(\hat{\mathrm{M}}) \leq \mathcal{D}_\Omega(\hat{\mathrm{M}}) + \sqrt{\frac{\log\delta}{-2|\Omega|}(\max(\mathbb{F}) - \min(\mathbb{F}))^2}] \geq 1-\delta$$

Therefore, the errors of the new problems are bounded. □

Next, we theoretically analyze the generalization error bounds of the proposed co-clustering based matrix approximation algorithm (Algorithm 1) and the ensemble method (Algorithm 2). Since the error bound of SVD based low-rank matrix approximation method has been well analyzed [19, 21], we focus on analyzing the error bound of compressed-sensing based method (defined in Equation 9) by using similar analysis techniques as in [5, 6]. As shown in [5, 6], we can recover a rank r matrix $M \in \mathbb{R}^{m\times n}(n \geq m)$ with probability at least $1-n^{-3}$, if the number of observed entries is $|\Omega| \geq C\mu^2 nr\log^6 n$, where C is a constant and μ is the strong incoherence parameter. However, this result is not applicable in our case, because the matrix M is approximated by multiple low-rank submatrices. Hence, we develop a new error bound based on a variant of the aforementioned conclusion.

The following analysis makes the following assumptions: (a) every submatrix $\mathcal{M}$ is a rank r matrix that satisfies the strong incoherent properties, and (b) the observed entries are uniformly distributed in submatrices such that the density of the observed entries ϱ in every submatrix is consistent with each other (i.e., $\varrho = \frac{|\Omega|}{mn}$). Without loss of generality, we assume $n \geq m$, and denote $\alpha = \max(\mathbb{F}) - \min(\mathbb{F})$, where $\mathbb{F}$ is the set of ratings. We start by analyzing the error bound of the co-clustering-based model $\hat{M}$ in Proposition 2. Then, based on Proposition 2, we proceed to derive an error bound on the global approximation $\tilde{M}$ in Proposition 3.

PROPOSITION 2. *If the density of the observed entries ϱ is large enough such that $|\Omega| \geq C\mu^2 nr\log^6 n$, then with probability of at leat $1-\delta$, $\hat{M}$ corresponding to a $k\times l$ co-clustering satisfies*

$$\mathcal{D}(\hat{M}) \leq \frac{(1+\beta_0)\alpha}{\sqrt{mn}}(4\sqrt{\frac{(2+\varrho)}{\varrho}(klm)} + 2kl),$$

where $\delta = (2kln)^{-3}$.

PROOF. For every user-item pair (u, i), an observation M_{ui} is equal to $\hat{M}_{ui} + Z$ where Z is a random variable whose absolute error is bounded by

$$\left\|W \otimes (M - \hat{M})\right\|_\infty \leq \left\|(1+\beta_0)(M - \hat{M})\right\|_\infty \leq (1+\beta_0)\alpha.$$

By applying Theorem 7 in [5] to matrix completion problem with bounded noise, we get with probability greater than $1-\upsilon^{-3}$ that every co-cluster-based approximation $\hat{\mathcal{M}}$ will satisfy

$$\left\|\mathcal{W} \otimes (\mathcal{M} - \hat{\mathcal{M}})\right\|_F \leq (1+\beta_0)\alpha(4\sqrt{\frac{\gamma(2+\varrho)}{\varrho}} + 2) \quad (14)$$

where $\upsilon = \max(|\mathcal{U}|, |\mathcal{I}|)$,$\gamma = \min(|\mathcal{U}|, |\mathcal{I}|)$. For one $k\times l$ co-clustering, there are kl different submatrices $\mathcal{M}_t, \gamma^{(t)}, t \in [kl]$, and obviously $\sum_{t\in[kl]}\gamma^{(t)} \leq m$. Using Cauchy-Schwarz inequality, we get

$$\sum_{t\in[kl]}\sqrt{\gamma^{(t)}} \leq \sqrt{kl\sum_{t\in[kl]}\gamma^{(t)}} \leq \sqrt{klm}. \quad (15)$$

Therefore, we can bound the approximation error as follows:

$$\begin{aligned}\left\|W \otimes (M - \hat{M})\right\|_F &\overset{(a)}{\leq} \textstyle\sum_{t\in[kl]}\left\|\mathcal{W}_t \otimes (\mathcal{M}_t - \hat{\mathcal{M}}_t)\right\|_F \\ &\overset{(b)}{\leq} \textstyle\sum_{t\in[kl]}(1+\beta_0)\alpha(4\sqrt{\frac{(2+\varrho)}{\varrho}\gamma^{(t)}} + 2) \\ &\overset{(c)}{\leq} (1+\beta_0)\alpha(4\sqrt{\tfrac{2+\varrho}{\varrho}(klm)} + 2kl)\end{aligned} \quad (16)$$

in which (a) holds due to the triangle inequality of Frobenius norm; and (b) holds due to (14); and (c) holds due to (15). Since for all (u, i) pairs, $\mathrm{W}_{ui} \geq 1$. Then, we have

$$\mathcal{D}(\hat{M}) = \frac{\left\|M - \hat{M}\right\|_F}{\sqrt{mn}} \leq \frac{\left\|W \otimes (M - \hat{M})\right\|_F}{\sqrt{mn}}. \quad (17)$$

Combining (16) and (17), we established the error bound of $\hat{M}$ as stated above. In order to adjust the confidence level, we take a union bound of the events $\left\|\mathcal{W} \otimes (\mathcal{M} - \hat{\mathcal{M}})\right\|_F \geq (1+\beta_0)\alpha(4\sqrt{\frac{\gamma(2+\varrho)}{\varrho}} + 2)$ for each submatrix $\mathcal{M}^{(t)}$, then we have

$$\sum_{t\in[kl]}\sqrt[3]{\upsilon^{(t)}} \leq \sqrt[3]{kl\sum_{t\in[kl]}\upsilon^{(t)}} \leq \sqrt[3]{2kln}.$$

i.e., the inequation in Proposition 2 holds with probabilities at least$1-\delta$ ($\delta = \sqrt[3]{2kln}$). □

PROPOSITION 3. *If Proposition 2 holds, then with probability of at leat $1-\delta$, the $\tilde{M}$ based on z different $k \times l$ co-clustering settings satisfies:*

$$\mathcal{D}(\tilde{M}) \leq \frac{(1+\beta_0)\alpha}{\sqrt{mn}}(4\sqrt{\frac{(2+\varrho)}{\varrho}(klm)} + 2kl),$$

where $\delta = z(2kln)^{-3}$.

PROOF. By Proposition 2, we bound the error of $\tilde{M}$ as follows:

$$\begin{aligned}
\left\|\tilde{M}-M\right\|_F &\overset{(a)}{\leq} \tfrac{1}{z}\left\|\textstyle\sum_{s\in[z]} Q^{(s)} \otimes (\tilde{M}-M)\right\|_F \\
&\overset{(b)}{=} \tfrac{1}{z}\left\|\textstyle\sum_{s\in[z]} Q^{(s)} \otimes (\hat{M}^{(s)}-M)\right\|_F \\
&\overset{(c)}{\leq} \tfrac{1}{z}\textstyle\sum_{s\in[z]} \left\|Q^{(s)} \otimes (\hat{M}^{(s)}-M)\right\|_F \\
&\overset{(d)}{\leq} \tfrac{1}{z}\textstyle\sum_{s}(1+\beta_0)\alpha(4\sqrt{\frac{2+p}{p}(klm)} + 2kl) \\
&= (1+\beta_0)\alpha(4\sqrt{\tfrac{2+p}{p}(klm)} + 2kl)
\end{aligned}$$

where (a) holds because every $\sum_{s\in[z]} Q^{(s)}_{ij} \geq \sum_{s\in[z]} 1 = z$; (b) holds due to Equation 13; (c) holds due to the triangle inequality of Frobenius norm; (d) holds due to Equation 16. Finally, by dividing $\sqrt{mn}$ from both sides, we conclude the proof. The confidence level is adjusted to $z(2kln)^{-3}$ using the union bound property as in Proposition 2. □

5. EXPERIMENTAL RESULTS

This section evaluates the proposed method on real-world datasets. The first study conducts sensitivity analysis. The proposed method consists of a set of parameters, i.e., rank of matrices r, and the number of row clusters k and column clusters l. This study evaluates how the recommendation accuracy of the proposed method is affected by these parameters. The second study compares the recommendation accuracy of the proposed method against six state-of-the-art matrix approximation based CF methods using PREA toolkit [13]. The third study evaluates the runtime scalability of the proposed method against standard SVD method.

5.1 Experiment Setup

The experimental study uses three real-world datasets that have been widely used for evaluating recommendation algorithms – 1) MovieLens 1M (10^6 ratings of $6,040$ users on $3,706$ items); 2) MovieLens 10M (10^7 rating of $69,878$ users on $10,677$ items); and 3) Netflix (10^8 rating of $480,189$ users on $17,770$ items). For each dataset, we split it into train and test sets randomly by setting the ratio between train set and test set as $9:1$. The results are presented by averaging the results over five different random train-test splits.

We use learning rate $v = 0.002$ for gradient decent method, $\lambda = 0.01$ for L_2-regularization coefficient, $\epsilon = 0.0001$ for gradient descent convergence threshold, and $T = 100$ for maximum number of iterations. The proposed method (WEMAREC) is compared against six state-of-the-art matrix approximation based CF methods, which are described as follows:

- NMF [Lee et al., NIPS' 01]: assumes the data and components are non-negative and every entry follows the Poisson distribution. Then the approximation is achieved by maximizing the log-likelihood.
- Regularized SVD [Paterek et al., KDD' 07]: is a standard matrix factorization method inspired by the effective methods of natural language processing, in which user/item features are estimated by minimizing the sum-squared error.
- BPMF [Salakhutdinov et al., ICML' 08]: is a Bayesain extension of probabilistic matrix factorization, in which the model is trained using Markov chain Monte Carlo methods.
- APG[1] [Toh et al., PJO 2010]: views the recommendation task as a matrix completion problem, and computes the approximation by solving a nuclear norm regularized linear least squares problem.
- DFC[2] [Mackey et al., NIPS' 11]: divides a large-scale matrix factorization task into smaller subproblems, and uses the techniques from randomized matrix approximation to combine the subproblem solutions.
- LLORMA [Lee et al., ICML' 13]: relaxes the low-rank assumption, and assumes that the original matrix is described using multiple low-rank submatrices, which are constructed using techniques from non-parametric kernel smoothing.

5.2 Sensitivity Analysis

We first show how WEMAREC performs with different combinations of parameters. MovieLens (10M) and Netflix datasets are used in this study with randomly selected 90% of data as training data and the rest 10% as test data.

Figure 3 presents the effects of the weighted function $p(x)$ (Eqn. 5) with β_0 varying in [0, 2.0] on three artificially selected datasets (from MovieLens (10M)) with different rating distributions. The detailed characteristics of the three selected datasets are presented in Table 2. As we can see, the RMSEs on all three datasets first decrease as β_0 increases from 0, and increases after the optimal accuracies are achieved. We also observe that optimal β_0s on more uneven datasets are smaller than those on more even datasets. The reason is that the frequency of different ratings are close on even datasets, so that a greater β_0 is required to make the weights of different ratings more different. Based on the above study, we choose $\beta_0 = 0.4$ for the following experiments because the submatrices generated by Bregman co-clustering are always uneven.

Figure 4 and 5 analyze the effects of Bregman co-clustering by changing the rank r and the numbers of row and column clusters k and l on MovieLens (10M) and Netflix dataset. We can see from the results that recommendation accuracies increase when the rank r increases from 5 to 20 in Figure 4. And the accuracies on the left (I-divergence) and middle (Euclidean-distance) are worse than those on the right (combination of these two distances), which indicates that the combination of different approximations $\hat{M}$ leads to better recommendation accuracy than both original ones. Figure 4 also shows that the recommendation accuracy decreases when k and l increase (from 2×2 to 3×3). This is due to the fact that each co-cluster based submatrix consists of less user-item ratings when k and l increase, resulting in insufficient training data for both model training and prediction. However, the accuracy first increases when k and l increase (from 2×2 to 2×3) in Figure 5, and then decrease when $k\times l = 3\times3$. This implies that larger dataset can be divided into more clusters to further improve the efficiency with improved accuracy. Based on this study, we choose rank $r = 20$ and $k\times l$ as 2×2 or 3×2 in the ensemble method, which offers the best recommendation accuracy for WEMAREC.

[1] `http://www.math.nus.edu.sg/~mattohkc/NNLS.html`
[2] `http://www.cs.ucla.edu/~ameet/dfc/`

	Rating = 1	Rating = 2	Rating = 3	Rating = 4	Rating = 5	Entropy
High	0.98%	3.14%	15.42%	40.98%	39.49%	1.174590
Median	3.44%	9.38%	29.25%	37.86%	20.06%	1.387499
Low	18.33%	26.10%	35.27%	16.88%	3.43%	1.445043

Table 2: Characteristics of three artificially selected datasets with different rating distributions (smaller entropy means that the dataset is more uneven).

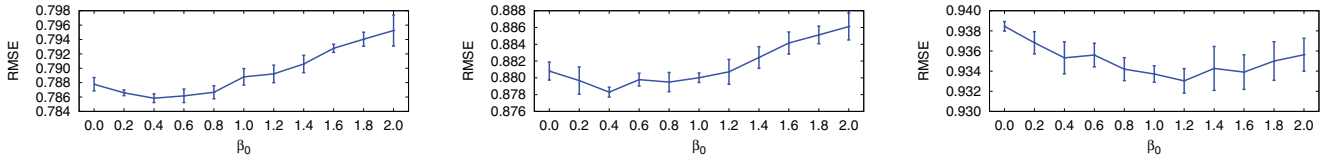

Figure 3: Effects of weighted function $p(x)$ on the performance of cocluster-based model over high-uneven(left), median-uneven(middle), low-uneven(right) datasets, with β_0 varying in $[0,\ 2.0]$.

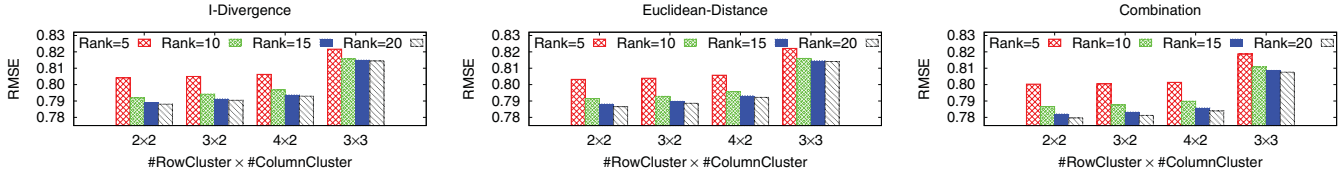

Figure 4: Effects of Bregman co-clustering on the performance of WEMAREC in I-divergence (left), Euclidean-distance (middle), and Combination of both two above distances (right), with the rank of submatrix varying in $\{5, 10, 15, 20\}$, the number of row and column clusters varying in $\{2 \times 2, 3 \times 2, 4 \times 2, 3 \times 3\}$ on MovieLens (10M).

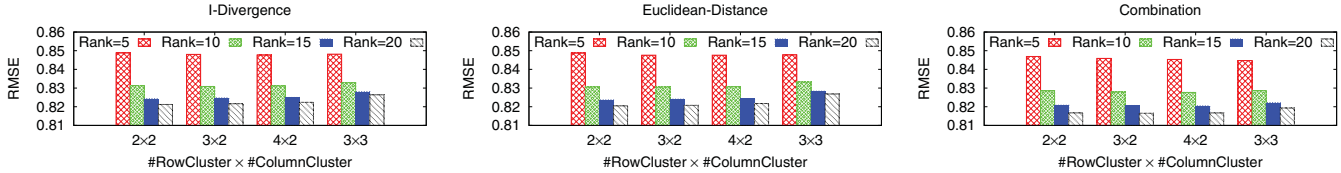

Figure 5: Effects of Bregman co-clustering on the performance of WEMAREC in I-divergence (left), Euclidean-distance (middle), and Combination of both two above distances (right), with the rank of submatrix varying in $\{5, 10, 15, 20\}$, the number of row and column clusters varying in $\{2 \times 2, 3 \times 2, 4 \times 2, 3 \times 3\}$ on Netflix.

Figure 6 analyzes the effect of ensemble weight function $q(x)$ (Eqn. 12) by selecting different β_1 and β_2. And we can see that the accuracies of the proposed weighted average methods always outperform the simple average method without weighting (i.e., $\beta_1 = 0.0, \beta_2 = 0.0$). It seems that larger β_1 will lead to better accuracy, but we also observe that the RMSE becomes stable when $\beta_1 > 40$. Therefore, we adopt $\beta_1 = 3.0$ and $\beta_2 = 40$ in the following experiments.

5.3 Recommendation Accuracy Comparisons

This study evaluates the accuracy of the proposed methods by comparing it with the six state-of-the-art matrix approximation based CF methods summarized in Section 5.1, i.e., NMF [11], Regularized SVD (RSVD) [16], BPMF [18], APG [25], DFC [14], LLORMA [12]. Each of the method is configured using the same parameters provided by the original paper. For the proposed WEMAREC method, we consider $2*2*2 = 8$ resulting matrix approximations, which are constructed by varying $\mathcal{C} \in \{\mathcal{C}_2, \mathcal{C}_5\}$, $d_\phi \in \{$*I-divergence*, *Euclidean-distance*$\}$ and $k \times l \in \{2 \times 2, 3 \times 2\}$. The MovieLens (10M) and Netflix datasets are used in this study.

Table 3 presents the RMSEs of all these matrix approximation based CF methods on the MovieLens (10M) and Netflix datasets. This study shows that the proposed WEMAREC method outperforms all the other six matrix approximation based CF methods on both the datasets. The reason why the proposed method can further improve the recommendation accuracy is due to 1) the new low-rank matrix approximation method can build more accurate models on submatrices because most often appeared ratings (major

	MovieLens (10M)	Netflix
NMF	0.8832 ± 0.0007	0.9396 ± 0.0002
RSVD	0.8253 ± 0.0009	0.8534 ± 0.0001
BPMF	0.8195 ± 0.0006	0.8420 ± 0.0003
APG	0.8098 ± 0.0005	0.8476 ± 0.0028
DFC	0.8064 ± 0.0006	0.8451 ± 0.0005
LLORMA	0.7851 ± 0.0007	0.8275 ± 0.0004
WEMAREC	**0.7769 ± 0.0004**	**0.8142 ± 0.0001**

Table 3: RMSE on MovieLens (10M) and Netflix of NMF (r=50) [11], Regularized SVD (r=50) [16], BPMF(r=30) [18], APG [25], DFC [14], LLORMA (r=20) [12], WEMAREC (r=20).

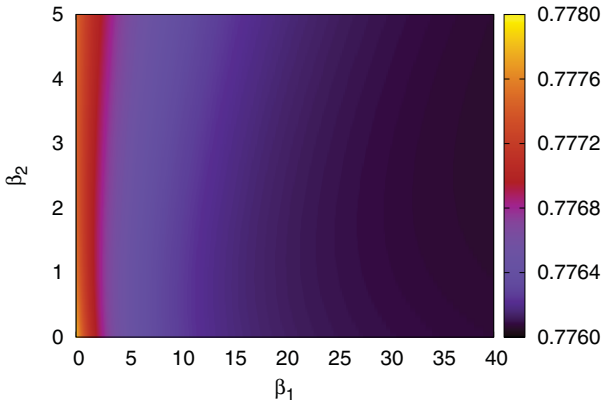

Figure 6: Effects of ensemble weighted function $q(x)$ on the performance of WEMAREC with different β_1 and β_2 on MovieLens 10M.

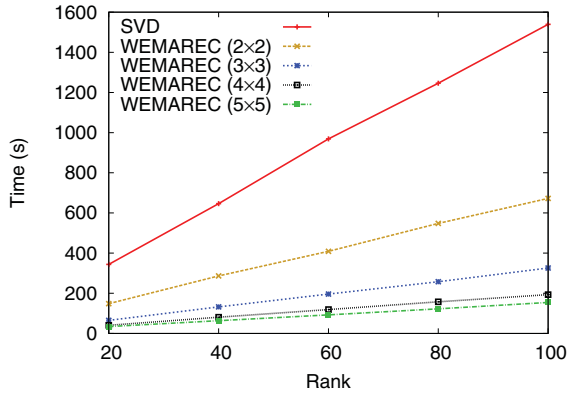

Figure 7: Efficiency comparisons of SVD method and WEMAREC method with different sizes of user-item rating matrix and different numbers of row clusters and column clusters.

interests) of users are treated more importantly; and 2) the ensemble method can effectively take advantage of the high-quality recommendation results from different co-clusterings to further improve the recommendation accuracy.

5.4 Scalability Analysis

This study evaluates how the WEMAREC method can speedup the recommendation process by leveraging parallel computing. The MovieLens (1M) dataset is used in this study. Leveraging parallel computing, all the submatrices are processed in parallel, and the execution time of WEMAREC is determined by that of the largest submatrix.

Figure 7 shows the execution time of WEMAREC by changing the co-clustering settings. The SVD method is included in this study for comparison purpose. This study shows that the execution time of SVD and WEMAREC both increases as the rank increases, and the performance of the WEMAREC method improves as the sizes of the submatrices decrease. WEMAREC outperforms SVD by 3X – 10X when the co-clustering setting varies from 2×2 to 5×5, and the speedup increases as the number of submatrices increases. This study demonstrates that WEMAREC can effectively improve the recommendation system scalability on large datasets.

6. RELATED WORK

Matrix factorization methods have been widely adopted in many applications [27], as well as recommendation systems [10]. Billsus et al. [4] initially introduced SVD to collaborative filtering context. Then, Srebro et al. [23] proposed a maximum-margin matrix factorization (MMMF) method, which can be formulated as a semi-definite programming problem for achieving matrix approximation based CF. Rennie et al. [17] investigated a direct gradient-based optimization method for achieving MMMF based CF, which can effectively improve the efficiency of MMMF method. Singh et al. [20] introduced a collective matrix factorization method based on relational learning to generalize existing matrix factorization methods and yielded new large-scale optimization algorithms for these problems. Yu et al. [28] proposed a non-parametric matrix factorization method, to make matrix approximation based CF methods applicable on large-scale datasets. Salakhutdinov et al. [15] extended matrix factorization to probabilistic algorithms by proposing a Probabilistic Matrix Factorization (PMF) method, which can scale linearly with the number of observations in the matrix. Based on the above work, a fully Bayesian treatment of PMF is present By Salakhutdinov et al. [18], which can train the user-rating matrix in recommender systems using Markov chain Monte Carlo methods.

In addition to single matrix factorization, ensemble methods have also been investigated in the literature. The Netflix Prize winners Bell et al. [3] and Koren et al. [9] utilized the combination of memory-based and matrix factorization methods to improve recommendation accuracy. Different from the above work, Mackey et al. [14] introduced a Divide-Factor-Combine (DFC) framework, in which the expensive task of matrix factorization is randomly divided into smaller subproblems which can be solved in parallel using an arbitrary base matrix factorization algorithms. Lee et al. [12] proposed a local low-rank matrix approximation (LLORMA) method, which generalized the DFC method in a way that a metric structure is used on the original matrix and the matrix partitions are constructed by kernel smoothing.

The DFC method and LLORMA method share similar idea with our method in that ensemble methods are adopted to boost recommendation accuracy. A significant difference between DFC and LLORMA is the construction of submatrices. In DFC, each submatrix is constructed by random sampling, while in LLORMA the submatrix is made of nearest neighbors within certain range. Different from both of them, the submatrices in our method are constructed via partitional co-clustering, so that each submatrix has low-parameter structure with less users and items, i.e., the submatrices in our method are of lower rank. Therefore, matrix approximation on such submatrices can be performed more efficiently. In addition, a submatrix-based weighting strategy is proposed to capture rating-specific prediction power of each submatrix, so that more accurate recommendations can be generated on more frequent samples in each submatrix. Therefore, the overall recommendation accuracy can be improved in each submatrix. Finally, the ensemble strategy in the proposed method can leverage both user-specific and item-specific rating distributions to combine the approximation matrices, which considers much more information compared with simple averaging method in DFC and kernel

smoothing method in LLORMA. Moreover, the proposed ensemble method is more efficient than the LLORMA method, because the kernel distance used in LLORMA method is based on the cosine distances between the rows of factor matrices, requiring extra singular value decompositions.

7. CONCLUSIONS

Matrix approximation methods have shown great success in recommender systems. However, tradeoff between scalability and accuracy must be made for most existing matrix approximation based CF methods. In this paper, a weighted and ensemble matrix approximation method (WEMAREC) is proposed to improve both recommendation accuracy and scalability. In WEMAREC, the user-item rating matrix is partitioned into a set of submatrices, which are then processed in parallel to improve system scalability. To optimize recommendation accuracy, a submatrix-based weighting strategy and an ensemble strategy are proposed. The weighting strategy improves the accuracy of the majority ratings of individual submatrices. The ensemble strategy improves the overall recommendation accuracy by combining multiple sets of co-clustering results based on the user-specific and item-specific rating distributions. Experimental study on MovieLens and Netflix datasets demonstrates that the proposed method can outperform state-of-the-art matrix approximation based CF methods on recommendation accuracy and scalability.

8. ACKNOWLEDGMENT

This work was supported in part by the National Natural Science Foundation of China under Grant No. 61233016, and the National Science Foundation of USA under Grant Nos. 1251257, 1334351 and 1442971.

9. REFERENCES

[1] G. Adomavicius and A. Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engeering*, 17(6):734–749, 2005.

[2] A. Banerjee, I. Dhillon, J. Ghosh, S. Merugu, and D. S. Modha. A generalized maximum entropy approach to bregman co-clustering and matrix approximation. *The Journal of Machine Learning Research*, 8:1919–1986, 2007.

[3] R. Bell, Y. Koren, and C. Volinsky. Modeling relationships at multiple scales to improve accuracy of large recommender systems. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 95–104. ACM, 2007.

[4] D. Billsus and M. J. Pazzani. Learning collaborative information filters. In *Proceedings of the 15th International Conference on Machine Learning*, pages 46–54, 1998.

[5] E. J. Candès and Y. Plan. Matrix completion with noise. *Proceedings of the IEEE*, 98(6):925–936, 2010.

[6] E. J. Candès and T. Tao. The power of convex relaxation: Near-optimal matrix completion. *IEEE Transactions on Information Theory*, 56(5):2053–2080, 2010.

[7] I. S. Dhillon, S. Mallela, and D. S. Modha. Information-theoretic co-clustering. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 89–98, 2003.

[8] T. George and S. Merugu. A scalable collaborative filtering framework based on co-clustering. In *The Fifth IEEE Intl. Conference on Data Mining*, pages 625–628. IEEE, 2005.

[9] Y. Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 426–434, 2008.

[10] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009.

[11] D. D. Lee and H. S. Seung. Algorithms for non-negative matrix factorization. In *Advances in neural information processing systems*, pages 556–562, 2001.

[12] J. Lee, S. Kim, G. Lebanon, and Y. Singer. Local low-rank matrix approximation. In *Proceedings of The 30th International Conference on Machine Learning*, pages 82–90, 2013.

[13] J. Lee, M. Sun, and G. Lebanon. Prea: Personalized recommendation algorithms toolkit. *The Journal of Machine Learning Research*, 13(1):2699–2703, 2012.

[14] L. W. Mackey, M. I. Jordan, and A. Talwalkar. Divide-and-conquer matrix factorization. In *Advances in Neural Information Processing Systems*, pages 1134–1142, 2011.

[15] A. Mnih and R. Salakhutdinov. Probabilistic matrix factorization. In *Advances in neural information processing systems*, pages 1257–1264, 2007.

[16] A. Paterek. Improving regularized singular value decomposition for collaborative filtering. In *Proceedings of KDD cup and workshop*, volume 2007, pages 5–8, 2007.

[17] J. D. Rennie and N. Srebro. Fast maximum margin matrix factorization for collaborative prediction. In *Proceedings of the 22nd international conference on Machine learning*, pages 713–719. ACM, 2005.

[18] R. Salakhutdinov and A. Mnih. Bayesian probabilistic matrix factorization using markov chain monte carlo. In *Proceedings of the 25th international conference on Machine learning*, pages 880–887. ACM, 2008.

[19] J. Shawe-Taylor, N. Cristianini, and J. S. Kandola. On the concentration of spectral properties. In *Advances in neural information processing systems*, pages 511–517, 2001.

[20] A. P. Singh and G. J. Gordon. Relational learning via collective matrix factorization. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 650–658. ACM, 2008.

[21] N. Srebro, N. Alon, and T. S. Jaakkola. Generalization error bounds for collaborative prediction with low-rank matrices. In *Advances In Neural Information Processing Systems*, pages 1321–1328, 2004.

[22] N. Srebro, T. Jaakkola, et al. Weighted low-rank approximations. In *Proceedings of the 20th International Conference on Machine Learning*, pages 720–727, 2003.

[23] N. Srebro, J. Rennie, and T. S. Jaakkola. Maximum-margin matrix factorization. In *Advances in neural information processing systems*, pages 1329–1336, 2004.

[24] X. Su and T. M. Khoshgoftaar. A survey of collaborative filtering techniques. *Advances in Artificial Intelligence*, Jan. 2009. Article ID 421425.

[25] K.-C. Toh and S. Yun. An accelerated proximal gradient algorithm for nuclear norm regularized linear least squares problems. *Pacific Journal of Optimization*, 6(615-640):15, 2010.

[26] B. Xu, J. Bu, C. Chen, and D. Cai. An exploration of improving collaborative recommender systems via user-item subgroups. In *Proceedings of the 21st International Conference on World Wide Web*, pages 21–30, 2012.

[27] J. Yan, M. Zhu, H. Liu, and Y. Liu. Visual saliency detection via sparsity pursuit. *IEEE Signal Processing Letters*, 17(8):739–742, 2010.

[28] K. Yu, S. Zhu, J. Lafferty, and Y. Gong. Fast nonparametric matrix factorization for large-scale collaborative filtering. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 211–218, 2009.

[29] Y. Zhang, M. Zhang, Y. Liu, and S. Ma. Improve collaborative filtering through bordered block diagonal form matrices. In *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval*, pages 313–322. ACM, 2013.