A General Collaborative Filtering Framework based on Matrix Bordered Block Diagonal Forms

Yongfeng Zhang, Min Zhang, Yiqun Liu, Shaoping Ma
State Key Laboratory of Intelligent Technology and Systems
Department of Computer Science & Technology, Tsinghua University, Beijing, 100084, China
zhangyf07@gmail.com {z-m, yiqunliu, msp}@tsinghua.edu.cn

ABSTRACT

Recommender systems based on Collaborative Filtering (CF) techniques have achieved great success in e-commerce, social networks and various other applications on the Web. However, problems such as data sparsity and scalability are still important issues to be investigated in CF algorithms. In this paper, we present a novel CF framework that is based on Bordered Block Diagonal Form (BBDF) matrices attempting to meet the challenges of data sparsity and scalability. In this framework, general and special interests of users are distinguished, which helps to improve prediction accuracy in collaborative filtering tasks. Experimental results on four real-world datasets show that the proposed framework helps many traditional CF algorithms to make more accurate rating predictions. Moreover, by leveraging smaller and denser submatrices to make predictions, this framework contributes to the scalability of recommender systems.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Filtering; H.3.5 [**Online Information Services**]: Web-based services

General Terms

Algorithms, Performance, Experimentation

Keywords

Collaborative Filtering, Bordered Block Diagonal Form, Graph Partitioning

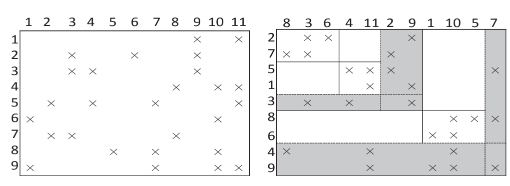
1. INTRODUCTION

This is an era of Big Data. The past years have witnessed a great explosion of data on the World Wide Web, which makes it difficult for users to find the information that they are really interested in on the internet. Recommender Systems attempt to meet these challenges by recommending the items that the users are potentially interested in, which is playing an important role on the Web that is becoming more and more personalized. With the ability to make recommendations without clear content descriptions of items, Collaborative Filtering (CF) [22] algorithms based on user-item rating matrices have been widely applied in various recommender systems thus far [22]. One of the core tasks of CF algorithms is to predict the scores that a user might rate on the items that he or she did not collect, and recommendations are then presented based on the predictions.

However, CF-based recommendation algorithms also suffer from the problems of data sparsity and systems scalability. Previous research attempting to address these problems mainly focus on various matrix-clustering [11, 26, 27, 37, 32] and community detection [6, 34, 23] techniques. Clustering-based approaches group users and/or items into clusters for CF. However, the clusters are usually difficult to interpret, and they usually assume that a user or an item should fall into one particular cluster, which might not be a reasonable assumption in reality. Community detection approaches based on user-item bipartite graphs attempt to improve accuracy and diversity by detecting user-item communities, but it is usually difficult for them to take advantage of various successful CF techniques on rating matrices.

In this work, we present a general CF framework based on Bordered Block Diagonal Form (BBDF) matrices. Figure 1 gives an example of the matrix structures to be leveraged in this framework. Figure 1(a) is a rating matrix, where each row/column/cross represents a user/item/rating, respectively. Figure 1(b) is a BBDF structure of the original matrix, where Row4, Row9 and Column7 are permuted to 'borders', and the remaining parts are thus permuted into two 'diagonal blocks'. Moreover, the permuting procedure is conducted recursively on the first diagonal block.

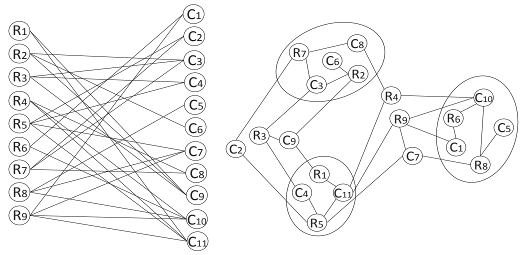
Permuting a sparse matrix into BBDF structure is equivalent to conducting Graph Partitioning by Vertex Separator (GPVS)-based community detection algorithms on its corresponding bipartite graph [1]. For example, Figure 2(a) is the bipartite graph for the matrix in Figure 1(a), and the GPVS result in Figure 2(b) corresponds to the BBDF matrix in Figure 1(b). In addition, the framework is compatible with any existing CF algorithm that is based on user-item rating matrices, which makes it capable of combining the



(a) Original Matrix (b) BBDF matrix

Figure 1: An example of BBDF Matrix.

24th ACM Conference on Hypertext and Social Media
1–3 May 2013, Paris, France



(a) Original Graph (b) After GPVS

Figure 2: An example of Graph Partitioning by Vertex Separator (GPVS)-based community detection.

advantages of community detection and matrix clustering techniques as well as making use of various CF algorithms.

In summary, the contributions of the paper are:

- To the best of our knowledge, this is the first time to attempt to investigate a general and easily interpretable structure of rating matrices under the background of CF recommendation tasks.
- The relationship between BBDF structures of rating matrices and community detection on bipartite graphs is investigated.
- A density-based algorithm is designed to transform rating matrices into BBDF structures and a general CF framework is proposed to make rating predictions.
- Both the efficiency and effectiveness of the proposed framework are verified through experimental studies.

The remainder of this paper will be organized as follows: section 2 reviews some related work, and section 3 introduces some preliminaries. Section 4 presents the proposed algorithms and framework. Experimental settings and results are shown in section 5, and section 6 concludes the work and provides some future directions.

2. RELATED WORK

Collaborative Filtering (CF) [22, 2] algorithms based on user-item rating matrices attempt to make recommendations by discovering and leveraging the knowledge of a user's preferences as well as the knowledge of others. They take advantage of the wisdom of crowds, and usually, they have no special requirements on items or domains.

User-based [28] and Item-based [29] CF algorithms are two best-known CF methods falling into the category of nearest neighbor approaches [22], which attempt to find the neighborhood of like-minded users or similar items to make rating predictions. Although simple to implement in practical systems, nearest neighbor approaches are usually unable to detect item synonymies and are also computationally expensive in real-world recommender systems.

The Matrix Factorization (MF) [16] approaches factorize a rating matrix into products of real-valued component matrices. Singular Value Decomposition (SVD) [31, 35] and Non-negative Matrix Factorization (NMF) [17, 38, 19] methods are typical MF algorithms investigated. However, the computationally expensive training components of these methods make them not scalable enough and impractical to conduct frequent model re-training. Incremental and distributed versions of SVD and NMF algorithms [4, 35, 30, 21, 10] partially alleviate this problem, but they are still not efficient enough because the effects of small updates to the rating matrix are not localized.

Various matrix clustering techniques have been investigated attempting to address the problems of efficiency, scalability and sparsity. User and item clustering methods [26] cluster user or item vectors first, and nearest neighbors of a user or item are restricted to its cluster. Some other matrix clustering techniques, such as co-clustering [7, 18, 11], ping-pang algorithm [27] and clustered low-rank approximation [33], cluster users and items at the same time, and the procedure of rating prediction takes advantage of these user-item clusters. By utilizing clusters, the scalability of recommender systems is usually improved, but clusters are usually difficult to interpret. In addition, these approaches usually force a user or item to fall into a single cluster, which might not be a reasonable assumption in real-world applications.

Recently, community detection techniques based on graphs have been investigated extensively with the rapid growth of social networks [20, 24, 23, 6, 25], which helps to improve both the accuracy and diversity of the recommender systems by extracting user or item communities.

In fact, permuting a sparse matrix into BBDF structure is equivalent to conducting community detection with Graph Partitioning by Vertex Separator (GPVS) algorithms on its corresponding bipartite graph [1, 3, 15]. Any collaborative filtering method can still be applied to the permuted matrix without any modification, but by leveraging user-item community information therein, more accurate and specific recommendations can be made.

3. PRELIMINARIES

Some preliminaries are presented in this section, which will be the basis of the BBDF permutation algorithm and the CF framework to be proposed in Section 4.

Definition 1. **Bordered Block Diagonal Form (BBDF).** *Matrix A is in Bordered Block Diagonal Form if:*

$$A = \begin{bmatrix} A_{11} & A_{12} & \cdots & A_{1k} & A_{1B} \\ A_{21} & A_{22} & \cdots & A_{2k} & A_{2B} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ A_{k1} & A_{k2} & \cdots & A_{kk} & A_{kB} \\ A_{B1} & A_{B2} & \cdots & A_{Bk} & A_{BB} \end{bmatrix} = \begin{bmatrix} D_1 & & & & C_1 \\ & D_2 & & & C_2 \\ & & \ddots & & \vdots \\ & & & D_k & C_k \\ R_1 & R_2 & \cdots & R_k & B \end{bmatrix} \quad (1)$$

Namely, $A_{ij} = \mathbf{0}$ $(i \neq j, 1 \leq i, j \leq k)$. Each D_i $(1 \leq i \leq k)$ is a 'diagonal block'; $R = [R_1 \cdots R_k B]$ and $C = [C_1^T \cdots C_k^T B^T]^T$ are 'borders'. Recursively, each of the diagonal blocks D_i can also be in the BBDF structure. □

BBDF structure is a generalization of Block Diagonal Form (BDF) matrices, for example, $A = \mathrm{diag}(D_1 D_2 \cdots D_k)$, where the latter has no border. Intuitionally, a diagonal block in a BBDF matrix is a 'user-item community', with its users and items being its 'dominant' users and items. The borders can be viewed as 'super' users and items among communities. Super users are users whose tastes are relatively broad and fall into different communities, and super items are items favored by users from different communities.

Definition 2. **Graph Partitioning by Vertex Separator (GPVS)-based Community Detection.**

Consider an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. $Adj(v)$ denotes the set of vertices adjacent to v. For a vertex subset $\mathcal{V}' \subset \mathcal{V}$, $Adj(\mathcal{V}') = \{v_j \in \mathcal{V} - \mathcal{V}' : \exists v_i \in \mathcal{V}' s.t.\ v_j \in Adj(v_i)\}$.

$\mathcal{V}_S \subset \mathcal{V}$ is a vertex separator if the subgraph induced by $\mathcal{V} - \mathcal{V}_S$ has $k \geq 2$ connected components. Formally, GPVS is defined as $\Gamma_v = \{\mathcal{V}_1\mathcal{V}_2 \cdots \mathcal{V}_k; \mathcal{V}_S\}$, where $\mathcal{V}_i \neq \emptyset$, $\mathcal{V}_i \cap \mathcal{V}_S = \emptyset$, $Adj(\mathcal{V}_i) \subset \mathcal{V}_S$ for $1 \leq i \leq k$, $\mathcal{V}_i \cap \mathcal{V}_j = \emptyset$ for $1 \leq i < j \leq k$, and $(\bigcup_{i=1}^{k} \mathcal{V}_i) \cup \mathcal{V}_S = \mathcal{V}$. Note that $\mathcal{V}_S = \emptyset$ is allowed. $\square$

Intuitively, the removal of a vertex separator splits the graph into k connected components in GPVS. As has been shown by the example above, GPVS is a type of community detection algorithm that corresponds to BBDF structures of sparse rating matrices. The reader might refer to [1] for a mathematical proof of the property.

Definition 3. **Density.** *Let A be an $m \times n$ matrix, let $\mathrm{n}(A)$ be the number of non-zeros in A, and let $\mathrm{area}(A) = m \times n$ be the area of A. The density of A is $\rho(A) = \frac{\mathrm{n}(A)}{\mathrm{area}(A)}$, and the average density of k matrices $A_1 \cdots A_k$ is $\bar{\rho}(A_1 \cdots A_k) = \frac{\sum_{i=1}^{k} \mathrm{n}(A_i)}{\sum_{i=1}^{k} \mathrm{area}(A_i)}$. Let $\mathcal{G}$ denote the bipartite graph of A; then, $\rho(\mathcal{G}) \triangleq \rho(A)$, $\bar{\rho}(\mathcal{G}_1 \cdots \mathcal{G}_k) \triangleq \bar{\rho}(A_1 \cdots A_k)$.* $\square$

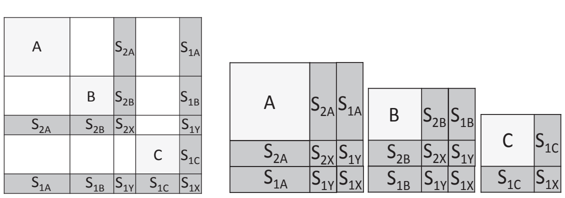
4. ALGORITHMS

4.1 BBDF Permutation Algorithm

In the BBDF permutation, a basic procedure is performed recursively, which is to permute some rows or columns to borders and to permute the remaining part to construct several diagonal blocks. This recursive framework is known as *George's nested dissection approach* [3, 14, 5], which has been widely used in fill-reducing ordering of sparse matrices. The basic step is equivalent to GPVS on the corresponding user-item bipartite graph, which has been shown in preliminaries. Perhaps the most widely known and used package for graph partitioning is Metis by Karypis [13], and we chose the core multilevel graph partitioning routine implemented in Metis as the basic GPVS algorithm.

Most graph partitioning algorithms try to balance the size of resulting subgraphs [1, 3], and usually require the expected number of subgraphs to be given in advance. However, this is not necessarily suitable for extracting user-item communities in CF tasks, since it is common that communities may not be evenly divided. In this study, we utilize the *density* of user-item communities to control the procedure of BBDF permutation because dense subgraphs are usually interpreted as actual communities. This approach has been widely used in community detection tasks [9].

The density-based BBDF permutation algorithm requires a parameter ρ as an input, which is a pre-defined requirement on the minimum average density of diagonal blocks. It conducts the basic procedure on a matrix and recurses on each of the resulting diagonal blocks until the density of a diagonal block has reached the density requirement ρ or the basic procedure cannot improve the average density any more. Algorithm 1 shows the procedure.



(a) BBDF matrix (b) Submatrices extracted

Figure 3: A toy example for extracting communities

Algorithm 1 BBDF-Permutation($A, \mathcal{G}, \rho$)

Require:
User-Item rating matrix A.
Bipartite graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}) = (\mathcal{R} \cup \mathcal{C}, \mathcal{E})$ of A. $\triangleright$ $\mathcal{R}$ and $\mathcal{C}$ are row and column vertex sets of $\mathcal{V}$, respectively.
Density requirement ρ.
Ensure:
Matrix A be permuted into BBDF structure.
1: $\rho_A \leftarrow \rho(A)$
2: **if** $\rho_A < \rho$ **then** $\triangleright$ else do nothing
3: $\Gamma_v \leftarrow \{\mathcal{V}_1\mathcal{V}_2 \cdots \mathcal{V}_k; \mathcal{V}_S\} \leftarrow$ GPVS($\mathcal{G}$)
4: Permute rows of A in order of $\mathcal{R}_1\mathcal{R}_2 \cdots \mathcal{R}_k\mathcal{R}_S$
5: Permute columns of A in order of $\mathcal{C}_1\mathcal{C}_2 \cdots \mathcal{C}_k\mathcal{C}_S$
6: $\bar{\rho} \leftarrow \bar{\rho}(D_1 D_2 \cdots D_k)$ $\triangleright$ D_i denotes the i-th diagonal block which corresponds to vertex set $\mathcal{V}_i = \mathcal{R}_i \cup \mathcal{C}_i$
7: **if** $\bar{\rho} > \rho_A$ **then** $\triangleright$ else do nothing
8: **for each** diagonal block D_i in A **do**
9: BBDF-Permutation($D_i, \mathcal{G}_{\mathcal{V}_i}, \rho$) $\triangleright$ $\mathcal{G}_{\mathcal{V}_i}$ is the subgraph induced by vertex set $\mathcal{V}_i$
10: **end for**
11: **end if**
12: **end if**

Note that, in the 7-th line, we do nothing if the average density of the resulting diagonal blocks is not improved compared with the original matrix, although some diagonals might not have reached the density requirement ρ. Such a diagonal block D is viewed as a sparse user-item community, and such a case occurs when the density requirement ρ is set too high. An important reason for using the average density to prevent such diagonals from recursion is that they would result in many small scattered communities with only a few users and items. Proper density requirements give better BBDF structures, and this issue will be discussed with the experimentation below.

4.2 BBDF-based Rating Prediction

A great advantage of the CF framework based on BBDF structure is that any CF algorithm can be used on a submatrix that is made up of diagonal blocks and borders. In this paper, we do not propose new CF algorithms on BBDF structures but instead propose a general CF framework, which makes use of user-item communities. By utilizing the community information, the prediction accuracy tends to be improved. Furthermore, conducting collaborative filtering on smaller and denser submatrices contributes to the scalability of CF algorithms.

The intuitive example shown in Figure 3 will be used to introduce the framework. For each diagonal block, we reconstruct its corresponding community by combining it with borders from different levels. In Figure 3 for example, three submatrices are constructed, which correspond to the diagonal blocks A, B and C. On each submatrix, any CF algorithm can be conducted to make rating predictions. There could be more than one prediction for user-item pairs from border crosses because they are shared by different submatrices. These predictions are averaged to construct the final prediction for the user-item pairs therein. It is a natural idea that the predictions can be averaged with different weights, but in this paper, we take only the average value. The averaging strategy could be investigated in future work.

5. EXPERIMENTS

5.1 Dataset Description

Experiments were conducted on four real-world datasets to validate the effectiveness of the proposed framework, which are MovieLens-100K, MovieLens-1M, Dianping and Yahoo! Music dataset.

Among these datasets, MovieLens-100K and MovieLens-1M are from the well-known MovieLens dataset[1]. We also collected a year's data from a famous restaurant rating web site DianPing[2] in China, and selected those users who made 20 or more ratings. The ratings also range from 1 to 5 like the MovieLens dataset. The Yahoo Music dataset [8] is from KDD Cup 2011[3], and its ratings range from 1 to 100. Statistics on these four datasets are presented in Table 1.

Table 1: Statistics of the four datasets

	ML-100K	ML-1M	Dianping	Yahoo! Music
#users	943	6,040	11,857	1,000,990
#items	1,682	3,952	22,365	624,961
#ratings	100,000	1,000,209	510,551	256,804,235
#ratings/user	106.045	165.598	43.059	256.550
#ratings/item	59.453	253.089	22.828	410.912
average density	0.0630	0.0419	0.00193	0.000411

These datasets are chosen because they have different sizes and densities. Additionally, two of them have more users than items, and the other two are the opposite. We expect to verify whether the framework works regardless of the size or density of datasets.

5.2 Algorithms and Evaluation Metrics

Four popular CF algorithms were experimented on using the framework. The User-based and Item-based CF algorithms are famous memory based approaches, while SVD and NMF are known to be famous matrix factorization approaches.

User-based: The Pearson correlation was used as user similarities as suggested in [28], and the neighborhood size is $k = 100$.

Item-based: Adjusted cosine similarity was used because it is reported to give the best performance in [29], and $k = 100$ is also used for the neighborhood size.

SVD: We use the popular SVD prediction strategy presented in [16]. Here, the number of factors k is 100, and the regularization coefficient λ is 0.015.

NMF: The most commonly used non-negative matrix factorization algorithm in [17] was used to make predictions, and we also used $k = 100$ and $\lambda = 0.015$.

To make a comparison with the literature, we used the Root Mean Square Error (RMSE) to measure the prediction accuracy in this work. For N rating-prediction pairs $\langle r_i, \hat{r}_i \rangle$, RMSE is defined as:

$$\text{RMSE} = \sqrt{\frac{\sum_{i=1}^{N}(r_i - \hat{r}_i)^2}{N}}$$

Five-fold cross validation was conducted on the MovieLens and Dianping datasets, and the average RMSE was calculated. For the Yahoo! Music dataset, we used its training set and validation set for training and evaluation, respectively.

[1]http://www.grouplens.org
[2]http://www.dianping.com
[3]http://www.sigkdd.org/kdd2011

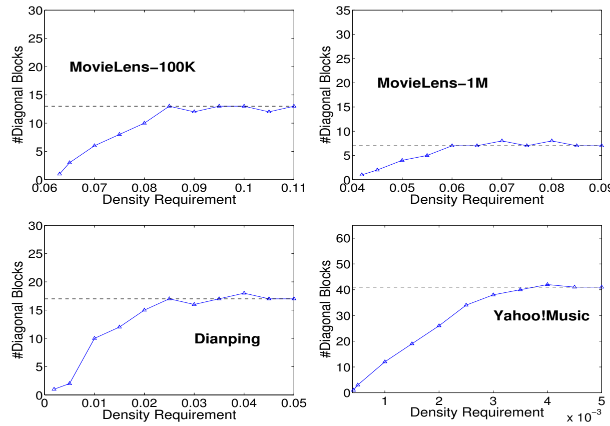


Figure 4: #communities v.s. density requirement

5.3 BBDF Algorithm Analysis

The most important parameter in the density based BBDF permutation algorithm is the density requirement ρ. Low density requirements lead to less and bigger user-item communities, and high density requirements result in more and smaller communities, as shown in Figure 5 where $\rho = 0.005$ and $\rho = 0.01$ are applied to Dianping dataset. An appropriate density requirement is important. If too low, user-item communities hidden in the original rating matrix cannot be extracted properly and completely. But if too high, it will result in many small scattered communities, which may lead to over fitting problems.

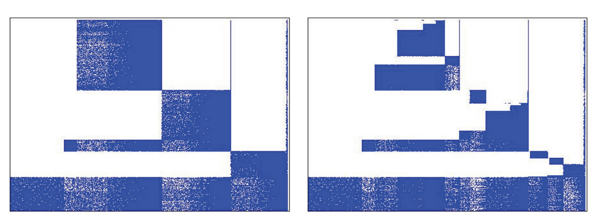
Figure 4 shows the relationship between number of extracted communities and density requirement on four datasets. We see that the number of communities rises at first, and tends to be stable after a certain density requirement, as a diagonal block will not be permuted recursively if its average density doesn't increase after graph partitioning.

5.4 Prediction Accuracy

Experimental results on four datasets show that the proposed CF framework helps existing CF algorithms to improve their accuracy in a large range of density requirements.

The experimental results on the RMSE versus the density requirement is shown in Figure 6. Each sub-figure presents the performance of all of the four CF algorithms on the corresponding dataset. Note that the beginning point of each curve represents the base performance of the CF method on the dataset, and the density requirement of this point is set to the density of the whole rating matrix. As a result, points on a curve below its beginning point mean an improvement on the prediction accuracy, and vice versa.

We must note the fact that the very large number of users and items in the Yahoo! Music dataset makes it unrealistic for current hardware to conduct memory-based CF algo-



(a) BBDF $\rho = 0.005$ (b) BBDF $\rho = 0.01$

Figure 5: BBDF structures on Dianping dataset

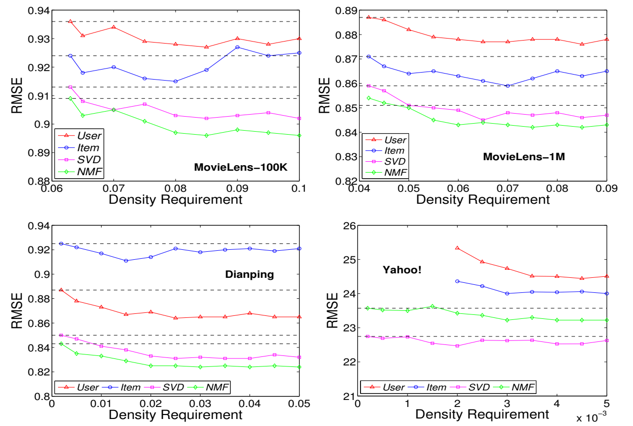


Figure 6: RMSE on four datasets based on BBDF structure given different density requirements

rithms. To the best of our knowledge, there is no report of direct user-based or item-based collaborative filtering on this dataset. Strategies such as KNN with SVD features [12] or parallelization [36] were attempted in KDD cup, but these strategies would make the results incomparable with other datasets. As a result, we did not perform KNN on the whole dataset. However, we found that our CF framework makes it possible to run KNN predictions on a standalone machine when the number of communities is more than 27, and experiments were performed in such cases.

Experimental results show that the proposed CF framework based on BBDF benefits user-based, SVD, and NMF algorithms consistently on all of the four datasets. However, mixed results were obtained on the item-based CF method. On MovieLens-1M, Dianping and Yahoo! Music dataset, its performance is slightly improved, but on MovieLens-100K, its accuracy rises at first and begins to drop while the density requirement continues to rise. Furthermore, a negative effect can even be introduced if the density requirement is set too high. This effect could be the reason that the item-based collaborative filtering algorithm utilizes historical ratings of a user, which is different from other CF algorithms that take advantage of neighborhood relationships. A user's historical ratings are reduced in the BBDF framework, especially on small datasets, which introduces bad predictions. However, it still benefits from BBDF structure when reasonable density requirements are set.

Table 2 presents the best performance of each CF method on each dataset based on the CF framework. When calculating the average RMSE on each dataset, five-fold cross-validation was conducted on MovieLens and DianPing dataset, and experiments were conducted five times on Yahoo! Music dataset. Standard deviations were ≤ 0.006 on MovieLens and DianPing and were ≤ 0.05 on Yahoo! Music.

Table 2: Best RMSE of the CF framework based on BBDF structures on the four datasets. The standard deviations are $\leq$ 0.006 on MovieLens and DianPing dataset and $\leq$ 0.05 on Yahoo! Music dataset.

Method	ML-100K		ML-1M		Dianping		Yahoo! Music	
	base	BBDF	base	BBDF	base	BBDF	base	BBDF
User-based	0.936	0.927	0.887	0.876	0.887	0.864	-	24.443
Item-based	0.924	0.915	0.871	0.859	0.925	0.911	-	24.002
SVD	0.913	0.902	0.859	0.845	0.850	0.831	22.746	22.469
NMF	0.909	0.896	0.854	0.842	0.843	0.824	23.572	23.223

5.5 Scalability and Efficiency

Experiments were conducted on a linux server with 8 core 3.1GHz CPU and 64GB RAM. For the BBDF permutation algorithm, we averaged the computational time consumed under different density requirements on each of the four dataset, as shown in Table 3.

Table 3: Computational time of BBDF algorithm

Dataset	ML-100K	ML-1M	Dianping	Yahoo!
average time	85.6ms	1.61s	21.2s	42.6min

Experiments show that the computational time of the BBDF algorithm increases along with the scale of rating matrices, but the time used for BBDF permutation is small compared with the CF prediction algorithms. Moreover, once BBDF structures have been constructed, they help to decrease the total prediction time by conducting collaborative filtering on smaller submatrices, especially for user-based and item-based methods. We calculated the average speedups for each CF algorithm on each dataset, as shown in Table 4, where speedup is defined as:

$$Sp = \frac{T_{CF}}{T_{BBDF} + T_{BBDF\_CF}}$$

T_{CF} is the time used by a CF prediction algorithm on the whole matrix, T_{BBDF} is the time of the BBDF permutation algorithm, and T_{BBDF_CF} is the time of making CF prediction algorithms using the framework based on BBDF structures. In almost all of the cases, BBDF structures speed up rating prediction; the speedup is obvious especially for nearest neighbor CF methods and large datasets.

Table 4: Speedups by conducting collaborative filtering based on BBDF structures on four datasets.

Dataset	ML-100K	ML-1M	Dianping	Yahoo!Music
User-based	1.28	1.24	1.33	-
Item-based	1.15	1.21	1.39	-
SVD	1.10	1.16	1.30	1.46
NMF	1.07	1.20	1.26	1.54

6. CONCLUSIONS

In this paper, we investigated the relationship between BBDF structures and community detection on user-item bipartite graphs, and we proposed an algorithm that, in fact, need only one intuitional parameter density requirement to permute a matrix into BBDF structures. We further proposed a general collaborative filtering framework that is based on BBDF structures to make rating predictions. Experimental results show that, by utilizing user-item communities contained in these structures, the proposed framework benefits many CF algorithms improving their prediction accuracies, and at the same time contributes to system scalability, which means that BBDF structures tend to be a general and promising framework to improve the performance of existing CF algorithms.

7. ACKNOWLEDGEMENT

The authors thank Shi Feng for the fruitful discussions, and the reviewers for their constructive suggestions. This work was supported by Natural Science Foundation (60903107, 61073071) and National High Technology Research and Development (863) Program (2011AA01A205) of China.

8. REFERENCES

[1] C. Aykanat, A. Pinar, and U. V. Catalyurek. Permuting Sparse Rectangular Matrices into Block-Diagonal From. *SISC*, 2004.

[2] R. M. Bell and Y. Koren. Scalable Collaborative Filtering with Jointly Derived Neighborhood Interpolation Weights. *Proc. ICDM*, 2007.

[3] E. Boman and M. Wolf. A Nested Dissection approach to Sparse Matrix Partitioning for Parallel Computations. *Proc. AMM*, 2007.

[4] M. Brand. Fast online SVD revisions for lightweight recommender systems. *Proc. SIAM SDM*, 2003.

[5] T. N. Bui and C. Jones. Finding Good Approximate Vertex and Edge Partitions is NP-hard. *Inform. Process. Lett.*, 1992.

[6] J. Chen and Y. Saad. Dense Subgraph Extraction with Application to Community Detection. *TKDE*, 2012.

[7] I. S. Dhillon, S. Mallela, and D. S. Modha. Information Theoretic Co-clustering. *Proc. SIGKDD*, pages 89–98, 2003.

[8] G. Dror, N. Koenigstein, Y. Koren, and M. Weimer. The Yahoo! Music Dataset and KDD-Cup'11. *KDDCUP*, 2011.

[9] S. Fortunato. Community Detection in Graphs. *Physics Reports*, 486:75–174, 2010.

[10] R. Gemulla, E. Nijkamp, P. J. Haas, and Y. Sismanis. Large-scale Matrix Factorization with Distributed Stochastic Gradient Descent. *KDD*, 2011.

[11] T. George and S. Merugu. A Scalable Collaborative Filtering Framework based on Co-clustering. *Proc. ICDM*, 2005.

[12] M. Jahrer and A. Toscher. Collaborative Filtering Ensemble. *KDDCUP*, 2011.

[13] G. Karypis. Metis-A Software Package for Partitioning Unstructured Graphs, Meshes, and Computing Fill-Reducing Orderings of Sparse Matrices-Version 5.0. *University of Minnesota*, 2011.

[14] G. Karypis and V. Kumar. A Fast and High Quality Multilevel Scheme for Partitioning Irregular Graphs. *SISC*, 1998.

[15] J. Kim, I. Hwang, Y. H. Kim, and B. R. Moon. Genetic Approaches for Graph Partitioning: A Survey. *Proc. CECCO*, pages 473–480, 2011.

[16] Y. Koren, R. Bell, and C. Volinsky. Matrix Factorization Techniques for Recommender Systems. *Computer*, 42:30–37, 2009.

[17] D. D. Lee and H. S. Seung. Algorithms for Non-negative Matrix Factorization. *NIPS*, 2001.

[18] K. W. Leung, D. L. Lee, and W. Lee. CLR: A Collaborative Location Recommendation Framework based on Co-Clustering. *Proc. SIGIR*, 2011.

[19] T. Li, C. Gao, and J. Du. A NMF-based Privacy-Preserving Recommendation Algorithm. *Proc. ICISE*, 2009.

[20] W. Lin, X. Kong, P. S. Yu, Q. Wu, Y. Jia, and C. Li. Community Detection in Incomplete Information Networks. *Proc. WWW*, pages 341–349, 2012.

[21] C. Liu, H. Yang, J. Fan, L. He, and Y. Wang. Distributed Nonnegative Matrix Factorization for Web-scale Dyadic Data Analysis on MapReduce. *Proc. WWW*, pages 681–690, 2010.

[22] J. Liu, M. Z. Q. Chen, J. Chen, F. Deng, H. Zhang, Z. Zhang, and T. Zhou. Recent Advances in Personal Recommender Systems. *Jour. Info. and Sys. Sci.*, 5:230–247, 2009.

[23] F. Luo, J. Z. Wang, and E. Promislow. Exploring Local Community Structures in Large Networks. *Proc. WI*, 2006.

[24] T. Nakahara and H. Morita. Recommender System for Music CDs Using a Graph Partitioning Method. *Proc. KES*, 2009.

[25] J. Noel, S. Sanner, K. Tran, P. Christen, and L. Xie. New Objective Functions for Social Collaborative Filtering. *Proc. WWW*, pages 859–868, 2012.

[26] M. O'Connor and J. Herlocker. Clustering Items for Collaborative Filtering. *Proc. SIGIR Workshop*, 1999.

[27] S. Oyanagi, K. Kubota, and A. Nakase. Application of Matrix Clustering to Web Log Analysis and Access Prediction. *Proc. WEBKDD*, pages 13–21, 2001.

[28] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl. GroupLens: An Open Architecture for Collaborative Filtering of Netnews. *CSCW*, 1994.

[29] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Item-Based Collaborative Filtering Recommendation Algorithms. *Proc. WWW*, pages 285–295, 2001.

[30] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Incremental Singular Value Decomposition Algorithms for Highly Scalable Recommender Systems. *Proc. ICCIT*, 2002.

[31] B. M. Sarwar, G. Karypis, J. A. Konstan, and J. T. Riedl. Application of Dimensionality Reduction in Recommender Systems - a case study. *WebKDD*, 2000.

[32] B. Savas and I. S. Dhillon. Clustered Low Rank Approximation of Graphs in Information Science Applications. *Proc. SIAM SDM*, pages 164–175, 2011.

[33] B. Savas and I. S. Dhillon. Clustered Low Rank Approximation of Graphs in Information Science Applications. *Proc. SIAM SDM*, pages 164–175, 2011.

[34] F. Wang, T. Li, X. Wang, S. Zhu, and C. Ding. Community Discovery using Nonnegative Matrix Factorization. *Journal of DMKD*, 22, 2011.

[35] Q. Wang, J. Xu, H. Li, and N. Craswell. Regularized Latent Sematic Indexing. *Proc. SIGIR*, 2011.

[36] Y. Wu, Q. Yan, D. Bickson, Y. Low, and Q. Yang. Efficient Multicore Collaborative Filtering. *KDDCUP*, 2011.

[37] B. Xu, J. Bu, and C. Chen. An Exploration of Improving Collaborative Recommender Systems via User-Item Subgroups. *WWW*, pages 21–30, 2012.

[38] S. Zhang, W. Wang, J. Ford, and F. Makedon. Learning from Incomplite Ratings Using Non-negative Matrix Factorization. *Proc. SIAM SDM*, 2006.