Local Matrix Approximation based on Graph Random Walk

Xuejiao Yang and Bang Wang*
Huazhong University of Science and Technology (HUST), Wuhan, China
yxj603@foxmail.com,wangbang@hust.edu.cn

ABSTRACT

How to decompose a large global matrix into many small local matrices has been recently researched a lot for matrix approximation. However, the distance computation in matrix decomposition is a challenging issue, as no prior knowledge about the most appropriate feature vectors and distance measures are available. In this paper, we propose a novel scheme for local matrix construction without involving distance computation. The basic idea is based on the application of convergence probabilities of graph random walk. At first, a user-item bipartite graph is constructed from the global matrix. After performing random walk on the bipartite graph, we select some user-item pairs as anchors. Then another random walk with restart is applied to construct the local matrix for each anchor. Finally, the global matrix approximation is obtained by averaging the prediction results of local matrices. Our experiments on the four real-world datasets show that the proposed solution outperforms the state-of-the-art schemes in terms of lower prediction errors and higher coverage ratios.

CCS CONCEPTS

• **Information systems** → **Recommender systems**.

KEYWORDS

Matrix factorization; local matrix construction; graph random walk; recommendation

ACM Reference Format:
Xuejiao Yang and Bang Wang. 2019. Local Matrix Approximation based on Graph Random Walk. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '19), July 21–25, 2019, Paris, France.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3331184.3331338

*Xuejiao Yang and Bang Wang are with the School of Electronic, Information and Communications, HUST. This work is supported in part by National Natural Science Foundation of China (Grant No: 61771209).

SIGIR '19, July 21–25, 2019, Paris, France

ACM ISBN 978-1-4503-6172-9/19/07…$15.00
https://doi.org/10.1145/3331184.3331338

1 INTRODUCTION

Recommendation systems have been playing an important role in the online social networks and e-commerce websites. Among various recommendation algorithms, the *matrix factorization* (MF) has gained lots of attentions since its great success in the Netflix contest [2]. For a *user-item rating matrix* $\mathbf{R} \in \mathbb{R}^{M \times N}$ with M users and N items, the MF technique is to predict the missing values in $\mathbf{R}$ by approximating $\mathbf{R}$ with another matrix $\hat{\mathbf{R}} = \mathbf{U}\mathbf{V}^{\mathsf{T}}$, where $\mathbf{U} \in \mathbb{R}^{M \times K}$ is a *user-factor matrix* and $\mathbf{V} \in \mathbb{R}^{N \times K}$ an *item-factor matrix*, with $K \ll \min(M, N)$. After performing a *global* MF on $\mathbf{R}$, a recommendation list for unrated items can be obtained according to the predicted ratings for each user.

Recently, a global rating matrix has become very large and sparse. Many studies have focused on how to convert the global matrix factorization task into several local matrix approximation tasks [1, 3, 4, 6]. For example, the DFC algorithm [4] randomly selects a subset of rows and columns from the global matrix to form many local matrices. It then performs a local MF on each local matrix and gets the final predictions by averaging the local approximations of the local matrices. Another important direction for decomposing the original rating matrix is based on the idea of anchor selection and neighborhood inclusion, as done in LLORMA [3] and CLLORMA [6]. They first select some of rated user-item pairs as anchors, and then find the neighbor users and items for each anchor, by which a local matrix is constructed.

Distance computation is needed for comparing the similarity in between users and items in the aforementioned algorithms when decomposing $\mathbf{R}$. However, engineering an appropriate feature as a user or item vector representation may become a challenging task. Although using the global MF can obtain user latent features from $\mathbf{U}$ and item latent features from $\mathbf{V}$, how to choose an appropriate feature dimensionality K again becomes a new challenge. Furthermore, there exist lots of distance measures, like the Euclidean distance, Pearson correlation, cosine similarity and etc., yet which one is the most appropriate for what kind of datasets is still an open problem. On the one hand, we are motivated from the recent findings for decomposing the original large matrix into many small local matrices. On the other hand, we would like to construct local matrices without involving distance computation.

In this paper, we propose a novel scheme to decompose the original rating matrix into many local matrices, called *random walk-based local matrix approximation* (RWLMA). We first construct a user-item bipartite graph for the original matrix $\mathbf{R}$. We then select some user-item pairs as *anchors* according to their convergence probabilities after performing a *random walk* (RW) on the bipartite graph. Then for each

anchor, we perform a *random walk with restart* (RWR) with the anchor user (item) as the restarting node to obtain the *correlation degrees* between this anchor user (item) and other users (items). Each non-anchor user-item pair is included into her top anchors' neighborhood, and for each anchor and her neighborhood, we construct a local matrix. The global matrix approximation is obtained by averaging the approximation results of local matrices. Experiments on four datasets including three public ones show that the proposed RWLMA can outperform the state-of-the-art ones in terms of lower prediction errors and higher coverage ratios.

2 THE PROPOSED METHOD

2.1 Anchor Selection

We propose to use a graph-based random walk to select anchors. Let $\mathbf{R} \in \mathbb{R}^{M \times N}$ denote a *user-item rating matrix* with M users and N items, where an element $r_{uv} \in \mathbf{R}$, if it exists, is a non-negative real value as the rating given to the vth item by the uth user. From $\mathbf{R}$, we construct a bipartite graph of users and items as follows: If r_{uv} exists, an undirected edge is established between u and v in the bipartite graph with the edge weight of r_{uv}. Motivated from the PageRank [5], we use a Markov chain to transform the anchor importance calculation task into a node convergence probability computation problem. Let $\mathbf{P}_{VU}$ be the probability transition matrix from items to users, which is obtained by column-normalizing the rating matrix $\mathbf{R}$. Let $\mathbf{P}_{UV}$ be the transition matrix from users to items, which is calculated by column-normalizing $\mathbf{R}^{\mathsf{T}}$.

We randomly initialize the probability vector of users and items as $\mathbf{u}^{(\mathbf{0})}$ and $\mathbf{v}^{(\mathbf{0})}$. To obtain the convergence probabilities, the proposed random walk algorithm is to iterative compute the following equations:

$$\mathbf{u}^{(t+1)} = (1-\alpha) \cdot \mathbf{P}_{VU}\mathbf{v}^{(t)} + \alpha \cdot \frac{1}{M} \tag{1}$$

$$\mathbf{v}^{(t+1)} = (1-\alpha) \cdot \mathbf{P}_{UV}\mathbf{u}^{(t)} + \alpha \cdot \frac{1}{N} \tag{2}$$

where $\mathbf{u}^{(t+1)}$ and $\mathbf{v}^{(t+1)}$, respectively, are the user and item probability vectors in the tth iteration. The parameter $\alpha \in (0,1)$ denotes the *random visit probability*. On the one hand, using α is equivalent to adding a very small weight for each node to connect with other nodes, thus guaranteeing the connectivity of the constructed bipartite graph. On the other hand, it is reasonable to assume that a user has some chance to select one item seemingly not within her previous interests. For example, in Eq. (1), a user node receives $(1-\alpha)$ probability of being visited from its connected item nodes, and α probability of being randomly visited by one of M users. Similar analysis applies in Eq. (2) for item transition.

The iteration terminates until the pairwise difference in between two iteration probability vectors is smaller than a predefined threshold. After the iteration termination, each node in the graph can obtain its convergence probability, which can reflect the importance of this node in the network to some extent. According to the convergence probabilities, we sort the list of users and the list of items in a decreasing order, respectively. Then we select the top A users and top A items from the sorted lists and randomly pair them to form in total A anchors. Let $\mathcal{A}$ denote the set of selected anchors.

2.2 Neighborhood Construction

For each anchor, we next determine its neighbor users and neighbor items so as to construct a corresponding local matrix. To this end, we apply a graph-based random walk with restart to measure the relations between an anchor and other non-anchor user-item pairs as follows. Take anchor (u_a, v_a) as an example. We first use the anchor user u_a as a restart node to compute its *correlation degree* with other user nodes. We use the one-hot vector to initialize the user probability vector as $\mathbf{u}^{(0)}$. That is, $\mathbf{u}^{(0)}(i) = 1$, if $i = u_a$. Otherwise, $\mathbf{u}^{(0)}(i) = 0$. We define the user restart vector as $\mathbf{r}_U$. For user u_a, $\mathbf{r}_U(i) = 1$, if $i = u_a$. Otherwise, $\mathbf{r}_U(i) = 0$. Furthermore, we randomly initialize the item probability vector $\mathbf{v}^{(0)}$.

To obtain the convergence probabilities, the RWR algorithm is to iteratively compute the following equations:

$$\mathbf{u}^{(t+1)} = (1-\beta) \cdot \mathbf{P}_{VU}\mathbf{v}^{(t)} + \beta \cdot \mathbf{r}_U \tag{3}$$

$$\mathbf{v}^{(t+1)} = \mathbf{P}_{UV}\mathbf{u}^{(t)} \tag{4}$$

where $\mathbf{u}^{(t+1)}$ and $\mathbf{v}^{(t+1)}$, respectively, are the probability vectors that user and item nodes are visited in the tth iteration. The *restart probability* β denotes that in each iteration, there is β probability to return to the restart node u_a and $1-\beta$ probability to walk from items to users. The iteration terminates until the pairwise difference in between the two iteration probability vectors is smaller than a predefined threshold. Let $\mathbf{u}_a$ denote the convergence probability vector of user nodes, which represents the correlation degree between these user nodes and u_a.

Similarly, we perform the random walk with restart for each anchor item v_a as follows: We use the one-hot vector to initialize the probability vector of items as $\mathbf{v}^{(0)}$, and randomly initialize the probability vector of users as $\mathbf{u}^{(0)}$. Let $\mathbf{r}_V$ be the item restart vector: If $i = v_a$, $\mathbf{r}_V(i) = 1$; Otherwise, $\mathbf{r}_V(i) = 0$. The iteration process is as follows:

$$\mathbf{v}^{(t+1)} = (1-\beta) \cdot \mathbf{P}_{UV}\mathbf{u}^{(t)} + \beta \cdot \mathbf{r}_V \tag{5}$$

$$\mathbf{u}^{(t+1)} = \mathbf{P}_{VU}\mathbf{v}^{(t)} \tag{6}$$

Let $\mathbf{v}_a$ denote the convergence probability vector of item nodes, which represents the correlation degree between these item nodes and v_a.

For each anchor $(u_a, v_a) \in \mathcal{A}$, we now obtain its convergence probability vectors $\mathbf{u}_a$ and $\mathbf{v}_a$, by which we construct a user convergence matrix $\mathbf{C}_U \in \mathbb{R}^{M \times A}$ and an item convergence matrix $\mathbf{C}_V \in \mathbb{R}^{N \times A}$, respectively. The ath column in $\mathbf{C}_U$ is the user convergence vector $\mathbf{u}_a$ for the anchor (u_a, v_a), and the uth row in $\mathbf{C}_U$ is the user u's convergence probability under different restart anchor node u_a, which can reflect the relationship between the user u and each anchor user. We define a *local matrix scale control parameter* $\rho (0.5 < \rho < 1)$ for each user and each item. That is, a user can only be

assigned into the user neighborhood of $\rho \times A$ anchors. The same applies to each item.

For each user u, we select the top $\rho \times A$ anchors in the decreasing order of the element value in the uth row of $\mathbf{C}_U$. In the end, we construct a set of neighboring users $\mathcal{U}_a$ for each anchor a. Similarly, the ath column in $\mathbf{C}_V$ is the item convergence vector $\mathbf{v}_a$ for the anchor (u_a, v_a), and the vth row in $\mathbf{C}_V$ can reflect the relationship between the item v and each anchor item. We assign each item v into the top $\rho \times A$ anchors in the decreasing order of the element value in the vth row of $\mathbf{C}_V$ and construct a set of neighboring items $\mathcal{V}_a$ for each anchor a.

Finally, we construct a local matrix for each anchor a based on the neighbor set $\mathcal{U}_a$ and $\mathcal{V}_a$ as follows: For each neighbor user $u \in \mathcal{U}_a$, we extract its corresponding uth row in the original rating matrix $\mathbf{R}$; And for each neighbor item $v \in \mathcal{V}_a$, we extract its corresponding vth column in $\mathbf{R}$. These extracted rows and columns then construct the local matrix $\mathbf{R}^a \in \mathbb{R}^{|\mathcal{U}_a| \times |\mathcal{V}_a|}$ for the anchor a.

We next discuss the coverage of local matrices. Let $\mathcal{P}$ and $\mathcal{P}_a$ denote the set of user-item pairs in $\mathbf{R}$ and in $\mathbf{R}^a$, respectively. For each element $p \in \mathcal{P}$, we define $\mathbb{I}(p) = 1$, if p appears in at least one of $\mathcal{P}_a$s; Otherwise, $\mathbb{I}(p) = 0$. The coverage of local matrices is defined as

$$\eta = \frac{1}{|\mathcal{P}|} \sum_{p \in \mathcal{P}} \mathbb{I}(p). \tag{7}$$

As we obtain the global predictions for missing values from those local predictions of local matrices, it is desirable to have complete coverage of the global rating matrix, that is, $\eta = 1$. We have the following lemma.

Lemma 2.1. *Our local matrices can achieve complete coverage of the global rating matrix* $\mathbf{R}$.

Proof. Proof is omitted due to limited space. □

2.3 Matrix Approximation

In each local matrix $\mathbf{R}^a$, we use the SVD algorithm[2] to predict ratings. The loss function in each local matrix is:

$$\arg \min_{(\mathbf{U}_a, \mathbf{V}_a)} \sum ([\mathbf{U}_a {\mathbf{V}_a}^T]_{u,v} - \mathbf{R}_{u,v})^2 + \lambda(||\mathbf{U}_a||^2 + ||\mathbf{V}_a||^2)$$

where $\mathbf{U}_a$ and $\mathbf{V}_a$, respectively, are the user-factor matrix and item-factor matrix of $\mathbf{R}^a$ after matrix factorization. We train each local matrix by the gradient descent method, and compute the prediction rating from user u to item v as:

$$\hat{\mathbf{R}}^a_{u,v} = [\mathbf{U}_a {\mathbf{V}_a}^T]_{u,v} \ (a = 1, 2, ..., A) \tag{8}$$

Finally, we obtain the global rating approximation $\hat{\mathbf{R}}_{u,v}$ by averaging the results $\hat{\mathbf{R}}^a_{u,v}$ from each local matrix to which (u, v) belongs.

3 EXPERIMENT

Experimental Settings: We conduct our experiments on four datasets: Ciao, Movielens-100K(ML-100K), Movielens-1M(ML-1M) and Zhihu Live(Zhihu). Movielens and Ciao are public datasets widely used in the rating prediction problem.

Table 1: Dataset Statistics

	#users	#items	#rating	density
Ciao	7375	106797	283119	0.04%
MovieLens-100K	943	1682	100000	6.3%
MovieLens-1M	6940	3952	1000209	3.65%
Zhihu Live	4482	3724	60285	0.01%

Zhihu Live is a real-time Q&A interactive product launched by Zhihu, a well-known online Q&A community in China. We craweled Zhihu Live website to build our dataset. In all the four datasets, the ratings are chosen from {1,...,5}. Table. 1 summarizes the statistics of the four datasets, where the *density* refers to the proportion of rated user-item pairs among all user-item pairs.

We compare the performance of the following algorithms:
SVD [2]: It is a kind of global matrix factorization method, which has been widely used for rating prediction.
LLORMA [3]: It splits the global rating matrix into many local matrices and obtains the global ratings through averaging the rating results from local matrix factorizations.
CLLORMA [6]: It improves the LLORMA by modifying its random anchor selection.
RWLMA: It is our proposed local matrix approximation based on graph random walk.

For the hyper-parameter settings, we set the learning rate $\mu = 0.01$, L2-regularization coefficient $\lambda = 0.001$, maximum number of iterations as 50 for the matrix factorization. The number of anchor points is fixed as 50 according to LLORMA [3] and CLLORMA [6]. For our scheme, we set the parameters as follows: $\alpha = 0.2$, $\beta = 0.5$ and $\rho = 0.7$. All the experiment results are averaged through the standard 5-fold cross-validation. To evaluate the performance of rating prediction, we apply the commonly used metrics: *Mean Absolute Error* (MAE) and *Root Mean Square Error* (RMSE). Obviously, the more accurate the prediction, the lower value of MAE and RMSE.

We observe from our experiments that for LLORMA and CLLORMA, some user-item pairs in the test datasets cannot be included into any of their constructed local matrices. In this case, we use the SVD to predict the ratings for such dangling pairs. We define a *Coveage* (Cvg) performance metric: $Coverage = \frac{|\mathcal{D}_{inc}|}{|\mathcal{D}_{tst}|}$, where $\mathcal{D}_{tst}$ is the test dataset and $\mathcal{D}_{inc}$ is the set of testing user-item pairs that can be included into at least one local matrix. Furthermore, as the size of a local matrix impacts on the local matrix training efficiency, we define a metric of *Normalized Local Matrix Average Size* (NLMAS) as $NLMAS = \frac{1}{|\mathcal{D}_{trn}|} \cdot \frac{\sum_{a=1}^{A} |\mathcal{D}^a_{trn}|}{A}$, where $\mathcal{D}_{trn}$ is the training set and $\mathcal{D}^a_{trn}$ the set of training user-item pairs in the local matrix of anchor a.

Experiment Results: Table 2 presents the experiment results for the four datasets. We can first observe that the SVD algorithm performs the worst in almost all cases. This is not unexpected as it performs a global matrix factorization over the original user-item rating matrix, which is very sparse in all the four datasets. All the other methods are based on

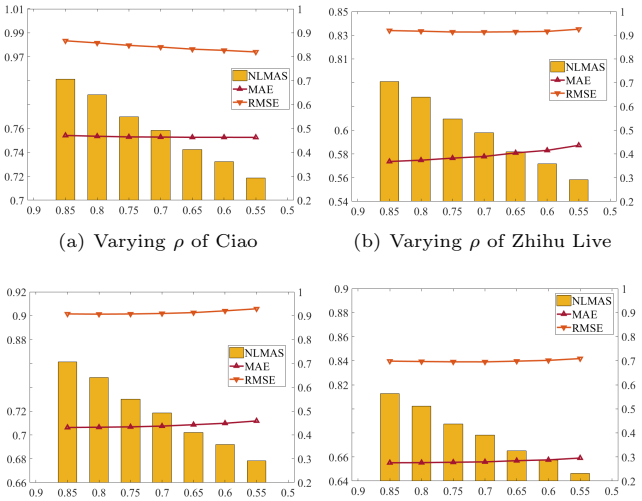
Table 2: Performances comparison on four datasets

Dataset	Metrics	SVD	LLORMA	CLLORMA	RWLMA
Ciao	MAE	0.7768	0.7636	0.7631	**0.7528**
			1.70%	1.76%	3.09%
	RMSE	1.0341	1.0090	1.0084	**0.9781**
			2.43%	2.49%	5.42%
	Cvg	–	70.63%	68.74%	**100%**
	NLMAS	–	84.34%	92.66%	**49.16%**
ML-100K	MAE	0.7434	0.7099	0.7111	**0.7074**
			4.51%	4.34%	4.84%
	RMSE	0.9603	0.9104	0.9119	**0.9019**
			5.20%	5.04%	6.08%
	Cvg	–	99.2%	98.5%	**100%**
	NLMAS	–	72.92%	75.04%	**49.15%**
ML-1M	MAE	0.6740	0.6600	0.6613	**0.6559**
			2.08%	1.88%	2.69%
	RMSE	0.8723	0.8469	0.8487	**0.8391**
			2.91%	2.71%	3.81%
	Cvg	–	99.4%	99.5%	**100%**
	NLMAS	–	51.52%	51.63%	**48.82%**
Zhihu	MAE	0.5796	0.5814	**0.5739**	0.5778
			-0.31%	0.98%	0.31%
	RMSE	0.8655	0.8632	0.8489	**0.8327**
			0.27%	1.92%	3.79%
	Cvg	–	97.7%	95.3%	**100%**
	NLMAS	–	89.14%	95.09%	**48.92%**

local matrix construction and factorization, so that the data sparsity can be alleviated in each smaller local matrix to some extent. In these local methods, we further observe that the proposed RWLMA algorithm can achieve the best prediction performance in almost all cases, with one exception that its MAE is slightly larger than that of CLLORMA in the Zhihu Live dataset. The results suggest the superiority of the proposed anchor selection and neighborhood construction based on the graph random walk.

As for the *NLMAS* metric, we observe that the proposed RWLMA constructs smaller local matrices, compared with the other local matrix methods LLORMA and CLLORMA, which can help to reduce computation complexity in local matrix factorization. Furthermore, for the *Coverage* of local matrices, we can observe that the proposed RWLMA achieves 100% coverage in all cases, yet the LLORMA and CLLORMA cannot achieve 100% in many cases. They select neighbors from the perspective of anchors based on the distance between anchors and non-anchor pairs and have not provided a mechanism to ensure complete coverage.

Figs. 1 plots the proposed RWLMA performance against the local matrix scale control parameter ρ. It can be observed that as ρ decreases, the NLMAS becomes smaller, indicating that local matrices become smaller; Furthermore, the MAE and RMSE experience a slight degradation in three datasets, but even perform slightly better in the Ciao dataset. We argue that this slight degradation might be justified by a higher computation efficiency, as local training and prediction can be performed in smaller size local matrices.



(a) Varying ρ of Ciao (b) Varying ρ of Zhihu Live

(c) Varying ρ of Movielens-100K (d) Varying ρ of Movielens-1M

Figure 1: Performance for different choices of ρ.

4 CONCLUSION

In this paper, we have proposed the RWLMA scheme to solve the classic matrix approximation problem by constructing many small local matrices. In RWLMA, a bipartite graph for user-item pairs is first established. Based on the graph, we have proposed to use the RW for anchor selection and the RWR for neighborhood construction. We then construct local matrices for each of anchors and compute local predictions in each local matrix. Experiments on four datasets have shown that our proposed scheme can outperform the state-of-the-art schemes in terms of lower approximation errors and higher coverage ratios. We notice that the RWLMA is merely based on the original user-item rating matrix. Our future work shall investigate how to integrate some other system information for local matrix construction and factorization.

REFERENCES

[1] A. Beutel, A. Ahmed, and A. J. Smola, "Accams: Additive co-clustering to approximate matrices succinctly," in *Proceedings of the 24th International Conference on World Wide Web*, ser. WWW '15, no. 11, 2015, pp. 119–129.
[2] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, no. 8, pp. 30–37, 2009.
[3] J. Lee, S. Kim, G. Lebanon, Y. Singer, and S. Bengio, "Llorma: local low-rank matrix approximation," *The Journal of Machine Learning Research*, vol. 17, no. 1, pp. 442–465, 2016.
[4] L. Mackey, A. Talwalkar, and M. I. Jordan, "Divide-and-conquer matrix factorization," in *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2011, pp. 1134–1142.
[5] L. Page, S. Brin, R. Motwani, and T. Winograd, "The pagerank citation ranking: Bringing order to the web," Stanford InfoLab, Technical Report 1999-66, 1999.
[6] M. Zhang, B. Hu, C. Shi, and B. Wang, "Local low-rank matrix approximation with preference selection of anchor points," in *Proceedings of the 26th International Conference on World Wide Web Companion*, ser. WWW '17 Companion, no. 9, 2017, pp. 1395–1403.