Neural Metric Matrix Factorization

Boran Zheng\(^1\), Mingzhi Mao\(^2\)*

\(^1\)School of Data and Computer Science, Sun Yat-sen University, Guangzhou, China
\(^2\)School of Data and Computer Science, Sun Yat-sen University, Guangzhou, China

*Corresponding author e-mail: mcsmmz@mail.sysu.edu.cn

Abstract. Nowadays, matrix factorization (MF) has been widely adopted in industry and research as a classical collaborative filtering (CF) algorithm. Unfortunately, the dot product adopted by matrix factorization is against the triangle inequality, which is one of the main reasons why this model is opposed. To address the issue, we propose a recommendation algorithm combining metric learning and MF in the paper. It transforms users’ preferences into distances, using Euclidean distance which satisfies triangular inequalities instead of traditional dot products, then directly decomposes the distance matrix into latent factor matrices of users and items. A multi-layer feedforward neural network is adopted for learning the model. Extensive experiments on two real-world datasets show that the proposed model is obviously superior to some advanced models based on matrix factorization.

1. Introduction

In the era of information explosion, it is of vital importance to obtain useful information among massive data. Recommender systems are the primary means to address these problems and have been adopted widely in many fields, including social media sites, e-commerce and online news. Collaborative filtering (CF), as a classic recommendation method, has been widely studied and applied in both research and industry\([1,2]\). Among various collaborative filtering technologies, matrix factorization has turned to the most popular recommendation standard model on account of its scalability, simplicity and flexibility, as well as its excellent performance in high-profile competitions such as Netflix Prize\([3,4]\).

Matrix factorization (MF) decomposes user-item rating matrix into user latent factor matrix and item latent factor matrix, and calculates the dot product of the two matrices to get the predictions. Many studies have been devoted to improving this algorithm. Koren et al. \([12]\) combined it with neighbor-based models. Jamali et al. \([13]\) argued that there should be trust propagation in recommendation, and that users’ favorite items should be similar to those of their trustees. With the vigorous development of deep learning technology, the improvement of matrix decomposition algorithm ushered in a new upsurge. Li et al. \([14]\) applied denoising auto-coder to recommendation, and reconstructed users’ score by learning hidden structure and definite historical score. Wu et al. \([9]\) proposed a new method called collaborative denoising automatic coder (CDAE), which uses implicit feedback to model users’ preferences. The Neural Collaborative Filtering (NCF) algorithm proposed by He et al. \([8]\) uses multi-layer feedforward neural network to improve the performance.

However, the dot product is against the triangle inequality \([5, 6]\), which is one of the main reasons why the factorization models such as matrix factorization have been opposed. In addition, studies have shown that matrix factorization is prone to over-fitting for large latent vectors \([8, 9]\). On the other hand,
metric learning algorithm adopts Euclidean distance as similarity measure, which is fundamental for many machine learning algorithms such as KNN, SVM, K-means algorithm and so on. Applying metric learning algorithm to the recommendation system not only demonstrates its simplicity, but also demonstrates its effectiveness[7,10,11]. Among them, collaborative filtering learning (CML) proposed by Hsieh et al. [7] performs well in solving the problems faced by matrix factorization algorithm. However, it also has certain problems. If the vector space is overcrowded, the learning cluster may also suffer from overcrowding when measuring similarity, resulting in a suboptimal solution [11].

In order to address such problems, we propose a recommendation algorithm based on matrix factorization combined with distance metrics, and use multi-layer feedforward neural network for modeling. The basic idea is to use spatial distance to reflect users’ preferences and convert users’ preferences into distances using Euclidean distance instead of inner product. The distance matrix will be decomposed into latent factor matrices of users and items directly and build learning model through a multi-layer feedforward neural network. Finally, the distances between similar users and items are smaller, and the distances between different users and items are larger. The distances between users and items can be regarded as a recommendation standard. The experimental results show that our model has a good performance in top-n ranking recommendation.

2. Background

2.1. Matrix Factorization

The traditional matrix factorization method is based on a simple intuition: suppose there are $m$ users and $n$ items in the recommender system. Let $R_{mn}$ denotes the rating matrix and $p_u$ and $q_i$ denote the latent vector for user $u$ and item $i$, respectively. MF expects their dot product close to $R_{mn}$. In this way, the missing values in $R_{mn}$ in the real world can be filled. Squared loss function and stochastic gradient descent method are usually adopted to complete the training process,

$$J(\theta) = \frac{1}{2} \sum_{u=1}^{m} \sum_{i=1}^{n} (R_{u,i} - p_u^T q_i)^2 + \frac{\lambda}{2} (\|p_u\|^2 + \|q_i\|^2)$$

(1)

Where $\lambda$ is a regularization parameter, and $\|\cdot\|_2$ denotes the $L2$ norm. However, in real world, the explicit feedback of rating is often missing, not every user will rate every item. While implicit feedback such as like, follow, browsing frequency, collecting etc is more common. Information brought by implicit feedback can not be ignored, which is usually richer and more customer-friendly than explicit feedback [16, 17]. However, the matrix factorization algorithm can not be directly applied to implicit feedback. When user $u$ does not rate item $i$, it does not necessarily mean $u$ does not like $i$, it can be that the user is not aware of the item.

To solve the problems, Hu et al. [18] and Pan et al. [19] proposed Weighted Regularized Matrix Factorization (WRMF). As mentioned above, the negative sample in implicit feedback is not necessarily a real negative sample. Pan et al. [19] named the problem as a single class problem. A weight coefficient $c_{ui}$ is introduced to reduce the influence of negative samples, i.e.,

$$\sum_{u=1}^{m} \sum_{i=1}^{n} c_{ui} (R_{u,i} - p_u^T q_i)^2 + \frac{\lambda}{2} (\|p_u\|^2 + \|q_i\|^2)$$

(2)

For positive samples, the value of $c_{ui}$ is larger, while smaller for negative samples.

2.2. Metric Learning

The most common metric learning methods used in recommender systems are Euclidean distance and Mahalanobis distance. The Mahalanobis distance is effective to calculate the similarity between two unknown sample sets. To apply Mahalanobis distance in recommender systems, let matrix $A$ represents metric distance matrix, formula (3) calculates the Mahalanobis distance between two points $x$ and $y$,

$$d^2_A(x, y) = (x - y)^T A (x - y)$$

(3)

Where $A$ is a semi-positive definite matrix. To solve the problems above, the most original approach was to globally solve the following convex optimization problem,
\begin{equation}
\min_A \sum_{(x,y) \in S} d_A^2(x, y) \\
s.t. A \geq 0, \sum_{(x,y) \in S} d_A^2(x, y) \geq 1
\end{equation}

Where $S$ denotes the set where $x$ and $y$ are considered similar. Although Mahalanobis distance seems more reasonable than Euclidean distance in metric learning, Euclidean distance is more popular in recommender systems due to its simplicity. Khoshneshin et al. [20] for the first time combined distance metric learning methods with matrix factorization. CML proposed by Hsieh et al. is the latest competitive collaborative filtering algorithm that integrates distance metric learning. They also use European distance. CML refers to the ideas of Largest Margin Nearest Neighbor (LMNN) and Bayesian Personalized Ranking (BPR) [21]. CML omits pull operation in LMNN and only uses push loss function in LMNN. This makes the items that users don’t like pushed aside, but can not get the items that users like close. CML’s authors also point out that overcrowding and sub-optimal solutions may occur in this case.

3. Proposed Model

The discussion above shows that the purpose of using collaborative filtering algorithm to study recommender systems is not only to predict ratings by explicit feedback of users, but also to learn users’ preferences for different items from the interaction between users and items. In this section, we introduce the Neural Metric Matrix Factorization (NMMF), which is more scientific and effective for capturing users’ preferences for items.

3.1. Metric Learning

Suppose there are $m$ users $U = \{u_1, u_2, \cdots, u_m\}$ and $n$ items $I = \{i_1, i_2, \cdots, i_n\}$ in the recommendation system. Let matrix $R = [r_{ui}]_{mn}$ represents user $u$’s rating for item $i$. The matrix $R$ is usually sparse due to the lack of ratings in the real world, therefore it is usually transformed into an implicit user-item interaction matrix $Y$. Dai et al. [15] pointed out that there are usually two ways to accomplish this transformation,

\[
Y_{ui} = \begin{cases} 
0, & R_{ui} = \text{unk} \\
1, & \text{otherwise}
\end{cases}
\]

\[
Y_{ui} = \begin{cases} 
0, & R_{ui} = \text{unk} \\
R_{ui}, & \text{otherwise}
\end{cases}
\]

Where unk denotes the missing value in matrix $R$. The traditional matrix factorization algorithm decomposes the rating matrix $R$ or user-item interaction matrix $Y$ into user and item latent factor matrix, and learns users’ preferences for different items through inner product. We need to convert the user preference to distance at first. Zhang et al. [22] proposed a method in their model. The distance matrix can be obtained by,

\[
\text{Distance}(u, i) = \text{MaxSimilarity} - \text{Similarity}(u, i)
\]

Where MaxSimilarity represents the maximum of the rating matrix $R$ or user-item interaction matrix $Y$. For explicit feedback, the maximum value of rating matrix $R$ is usually 5; for implicit feedback, the maximum value of user-item interaction matrix $Y$ is usually 1.

Explicit/implicit feedback can be transformed into distance via the method above. As has been known to us, distance metric function should satisfy triangle inequality. And as has been mentioned in section 2, Mahalanobis distance and Euclidean distance are mainly adopted in recommender systems. And Euclidean distance has been widely used due to its simplicity and practicability. In order to avoid the computational complexity caused by the squared root, the squared Euclidean distance is usually adopted in practical research.

3.2. Metric Matrix Factorization

Based on the above theory, we improve the traditional matrix factorization algorithm and combine it with distance metric learning, finally establish a new matrix factorization model (MMF) to serve our algorithm.
We can convert the rating matrix into implicit feedback and get the user-item interaction matrix. In order to convert implicit feedback into distance, we adopt the following methods.

\[ Y_{ij} = \begin{cases} 0 & \text{if observed} \\ d_0 & \text{otherwise} \end{cases} \]

Where \( d_0 \) is the distance factor. According to discussion above, implicit feedback is converted to distance value 0 while there are interactions between users and items, and 1 otherwise. In order to increase the influence of user-item interaction, distance factor \( d_0 \) is adopted instead of 1, thus the results are more realistic.

Let \( u_i \) and \( v_j \) represent latent factor vectors of users and items respectively. We measure the distance between user and item with squared Euclidean Distance,

\[ D(i, j) = \| u_i - v_j \|^2 \]  

(6)

And we adopt pointwise loss to directly factorize the distance into user and item embeddings.

\[ J(\theta) = \sum_{i=1}^{m} \sum_{j=1}^{n} w_{ij} (Y_{ij} - D(i, j))^2 \]  

(7)

Where \( w_{ij} \) denotes the weight. In MMF we need to consider the unobserved user-item interaction. Due to the reason that we can not distinguish the unobserved user-item interaction is that the user does not like the item or the user does not know the item, we add the weight value \( w_{ij} \) to reduce the weight of these negative samples. For negative samples, the value of \( w_{ij} \) is smaller while larger for positive samples. \( w_{ij} \) is defined as: \( w_{ij} = 1 + \beta \omega_{ij} \). The value of \( \omega_{ij} \) can be obtained directly from the user-item interaction matrix (0 or 1), and \( \beta \) is a hyperparameter. By learning the loss function, the MMF model enables users to be closer to items they like and distant from items they don’t like.

### 3.3. Neural Metric Matrix Factorization

Nowadays, neural networks have become popular in many fields. Neural networks have also been proved to be able to fit any continuous function. Therefore, we apply it to the MMF model, expecting to learn the interaction function between users and items.

As mentioned in Section 3.2, distance matrix \( Y_{ij} \) can be constructed. Then we adopt deep neural network structure to project user vector and item vector into potential structure space. Specifically, the distance between user \( i \) and all items is expressed by \( p_i \), and the distance between the \( j \)th item and all users is expressed by \( q_j \). They can be regarded as the high dimensional vector representation of matrix \( Y_{ij} \), and also as the input vector of the input layer. In each layer, each input vector is mapped to another vector in the new space. For vector \( u_i \), assume that the weight of each layer is denoted by \( W_u, i = 1, 2, \cdots, L \). Biases of each layer are represented by \( b_u \). For vector \( v_j \), assume that the weight of each layer is denoted by \( W_v, j = 1, 2, \cdots, L \). Biases of each layer are represented by \( b_v \). And we use the ReLU as the activation function at the hidden layers and the output layer. Through our deep neural network structure, user vector \( p_i \) and item vector \( q_j \) are mapped to two latent factor vectors \( u_i \) and \( v_j \), respectively.

\[
\begin{align*}
    u_i &= h_L(\cdots h_2(W_u h_1(W_u p_i + b_u))) \\
    v_j &= h_L(\cdots h_2(W_v h_1(W_v q_j + b_v)))
\end{align*}
\]

(8)

Where \( h(*) \) denotes the activation function ReLU, which is defined as \( h(x) = \max(0,x) \). As before, we still use Squared Euclidean Distance to learn user latent factor vector \( u_i \) and item latent factor vector \( v_j \), then the final predictions can be expressed as follows:

\[ \hat{Y} = D(i, j) = \| u_i - v_j \|^2 \]  

(9)

For the loss function of NMMF model, we still use the same loss function as that of MMF model according to formula 7. Normally, the loss function of the model needs regular terms to prevent over-fitting, so we also take some measures. Usually, the \( L2 \) norm of \( u_i \) and \( v_j \) is used in matrix factorization algorithm, but it has no practical significance for our model. As has been mentioned in CML [7], \( L2 \) norm will pull each object to the origin, but in distance metric space, the origin has no specific meaning. Therefore, we use the constraints of \( \| u_i \|_2 \leq 1 \) and \( \| v_j \|_2 \leq 1 \) to regularize. This
method reduces the constraints on Euclidean spheres and limits the values of $u_i$ and $v_j$ in $L^2$ cell spheres. To achieve this, we can project all latent factor vectors to the unit ball at the beginning of each training.

4. Experiments

4.1. Preliminaries

Data description. To measure the ranking performance of our model NMMF on implicit feedback, we conduct experiments on two publicly accessed datasets:

a) MovieLens. It contains the rating data of users for movies. We choose the MovieLens-1m version, which contains a million ratings for 3,706 movies from 6,040 users.

b) Pinterest. Pinterest is a photo sharing website. Users can put their favorite pictures on their own sketchpads. Other users can view these sketchpads and pin (means like). The dataset is from Geng et al. [23] for image recommendation. Using the processing method in [8], a dataset consisting of 55, 187 users and 1, 500, 809 interactions is constructed.

Evaluation Methodology. In order to evaluate the ranking performance of our model, we adopt two widely used metrics: Hit Ratio (HR) and Normalized Discount Cumulative Gain (NDCG). HR is a commonly used measure of recall rate in Top-k recommendation, indicating whether the test item exists in the Top-k list, while NDCG calculates the hit position by assigning a higher score to the top ranking.

Baselines. We compare NMMF with the following methods:

i) ItemKNN. It is a classical item-based collaborative filtering algorithm.

ii) CML. It is a state-of-the-art MF method based on metric learning. It adopts the push function to push items that users do not like apart. Meanwhile weighted ranking loss and covariance matrix regularization are adopted.

iii) eALS. It is a state-of-the-art MF method for item recommendation, which optimizes the squared loss function, treating all unobserved interactions as negative instances and weighting them non-uniformly by the item popularity.

iv) NeuMF. It is a state-of-the-art MF method which combines multi-layer perceptron with MF.

Parameter settings. We implemented our proposed model based on Tensorflow. 80% of the dataset is for training and the rest is for testing. We adopt mini-batch Adam and randomly initialized model parameters with a Gaussian distribution (mean 0, σ 0.01). The default size of batches is set to 256, and the default value of learning rate is 0.001.

4.2. Performance Comparison

As can be seen from Table 1, NMMF model outperforms other algorithms on both datasets, including CML model based on metric learning method and neural network model. Compared with the matrix factorization method eALS, NMMF performs better on all data sets, which also shows the rationality of distance measure relative to dot product. For CML using the same metric learning method, NMMF increased by 5.60% on all data sets, which shows that NMMF model can reflect the relationship of distance and similarity between user and item better. The NMMF model is also slightly better than NeuMF using the same neural network model, which shows the feasibility and advantages of the metric learning method compared with the traditional matrix factorization method.
Table 1. RESULTS FOR MODELS OF HR@10 AND NDCG@10

| Method  | ML-1m | Pinterest |
|---------|-------|-----------|
|         | HR@10 | NDCG@10   | HR@10 | NDCG@10 |
| ItemKNN | 0.626 | 0.352     | 0.787 | 0.485   |
| eALS    | 0.642 | 0.372     | 0.846 | 0.520   |
| CML     | 0.657 | 0.384     | 0.839 | 0.534   |
| NeuMF   | 0.688 | 0.410     | 0.869 | 0.546   |
| NMMF    | 0.704 | 0.427     | **0.881** | **0.562** |

4.3. Model Analysis And Discussion

In this section, the influence of parameter settings on NMMF is studied. Due to limited space the experiments were conducted only on ML-1m dataset, and we only report two metrics (NDCG and HR).

**Negative Sampling Ratio.** As has been mentioned in section II, the data sample with implicit feedback can be regarded as a single class problem, so it involves the sampling rate of negative samples. Therefore, we set different sampling rates for experiments (neg1 means that the sampling rate is 10%, others are similar). As shown in Table 2.

Table 2. RESULTS FOR MODELS WITH DIFFERENT NEG ON ML-1M.

| neg  | HR@10   | NDCG@10   | HR@10   | NDCG@10 |
|------|---------|-----------|---------|---------|
| 1    | 0.6664  | 0.6863    | 0.6980  | 0.7035  |
| 2    | 0.3932  | 0.4136    | 0.4176  | 0.4256  |
| 3    | 0.6949  | 0.7036    | 0.6981  | 0.6922  |
| 4    | 0.6949  | 0.7036    | 0.6981  | 0.6922  |
| 5    | 0.6949  | 0.7036    | 0.6981  | 0.6922  |
| 6    | 0.4233  | 0.4281    | 0.4229  | 0.4140  |
| 7    | 0.4233  | 0.4281    | 0.4229  | 0.4140  |
| 8    | 0.4233  | 0.4281    | 0.4229  | 0.4140  |
| 9    | 0.4233  | 0.4281    | 0.4229  | 0.4140  |

From Table 2, we can see that higher sampling rate can promote the improvement of the model effect. When the sampling rate is higher than 4, the model tends to be stable. The optimal sampling rate is between 50% and 70%, while the performance of the model decreases when the sampling rate is higher than 70%, which shows that high sampling rate is not conducive to the NMMF model.

**Hidden Layers.** In order to figure out whether the deep learning method improves the model or not, we set up different layers of neural networks to carry out experiments. As shown in Table 3.

Table 3. RESULTS FOR MODELS WITH DIFFERENT HIDDEN LAYERS ON ML-1M.

| Method  | MLP-0 | MLP-1 | MLP-2 | MLP-3 | MLP-4 |
|---------|-------|-------|-------|-------|-------|
| HR@10   | 0.6843 | 0.6952 | 0.6995 | 0.7003 | 0.7018 |
| NDCG@10 | 0.4084 | 0.4185 | 0.4255 | 0.4228 | 0.4281 |

From Table 3, we can see that the NMMF model with more hidden layers performs better, which also shows the feasibility and effectiveness of deep learning. More intuitively, the NMMF model with four layers is obviously better than the MMF model without neural network (the NMMF model with 0 layers) and illustrates the best performance. So, the deep learning method is helpful to our model.

**Predictive Factors.** Besides that the number of layers of the neural network will affect the NMMF model, because the last layer of the neural network outputs the latent factor vectors of the last user and item, to a certain extent, it determines the quality of the model. Therefore, we use different predictive factors for the last layer of the neural network. We set the prediction factors to 8, 16, 32, 64.
Table 4. RESULTS FOR MODELS WITH DIFFERENT PREDICTIVE FACTORS ON ML-1M.

|         | 8     | 16    | 32    | 64    |
|---------|-------|-------|-------|-------|
| HR@10   | 0.6732| 0.6874| 0.6967| 0.7017|
| NDCG@10 | 0.3976| 0.4104| 0.4205| 0.4256|

From Table 4, we can see that the model with 64 as the predictor achieves the best results, which also shows that larger predictors are more helpful to improve the performance of the model.

5. Conclusion
In this paper, we propose a neural recommendation method based on distance metric learning and MF. It converts user preferences into distances and uses Euclidean distance instead of traditional dot product to reflect user preferences, then decomposes the distance matrix directly into latent factor matrices of users and items. Finally a multilayer feedforward neural network is used for modeling. Extensive experiments on two real-world datasets also show the superiority of our model.

In the future, we may adopt more powerful neural networks such as RNN and consider incorporating temporal dynamics to complete dynamic recommendation. We will also explore the impact of other implicit feedbacks such as browsing history, social relations and social information on model recommendation.

6. Acknowledgments
This work was financially supported by research project (35000—14070034).

References
[1] Greg Linden, Brent Smith, and Jeremy York. Amazon.com recommendations: Item-to-item collaborative filtering. Internet Computing, IEEE, 2003.
[2] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Item-based collaborative filtering recommendation algorithms. In WWW, pages 285 – 295, 2001.
[3] Ruslan Salakhutdinov and Andriy Mnih. Bayesian probabilistic matrix factorization using Markov chain Monte Carlo. In Proceedings of the 25th international conference on Machine learning. ACM, 880 – 887, 2008.
[4] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix Factorization Techniques for Recommender Systems. Computer 42, 8 (Aug. 2009), 30 – 37.
[5] P. Ram and A. G. Gray. Maximum inner-product search using cone trees. In KDD’12, pages 931–939. ACM, 2012.
[6] A. Shrivastava and P. Li. Asymmetric lsh (alsh) for sublinear time maximum inner product search (mips). In NIPS’14, pages 2321–2329, 2014.
[7] Cheng-Kang Hsieh, Longqi Yang, Yin Cui, Tsung-Yi Lin, Serge Belongie, and Deborah Estrin. Collaborative Metric Learning. In Proceedings of the 26th International Conference on World Wide Web (WWW’17), 2017.
[8] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. Neural Collaborative Filtering. In WWW2017.
[9] Yao Wu, Christopher DuBois, et al. Collaborative denoising auto-encoders for top-n recommender systems. In Proceedings of the Ninth ACM International Conference on Web Search and Data Mining, pages 153–162. ACM, 2016.
[10] Kilian Q. Weinberger and Lawrence K. Saul. Distance Metric Learning for Large Margin Nearest Neighbor Classification. J. Mach. Learn. Res. 10 (June 2009), 207–244.
[11] Yi Tay, Luu Anh Tuan, and Siu Cheung Hui. 2018. Latent Relational Metric Learning via Memory-based Attention for Collaborative Ranking. In WWW2018, Lyon, France, April 23-27, 2018.
[12] Y. Koren. Factorization meets the neighborhood: A multifaceted collaborative filtering model. In KDD, pages 426–434, 2008.
[13] Jamalim, Esterm. A matrix factorization technique with trust propagation for recommendation in social networks[C]//Proceedings of the Fourth ACM Conference on Recommender Systems. ACM, 2010: 135-142.

[14] Sheng Li, Jaya Kawale, and Yun Fu. Deep collaborative filtering via marginalized denoising autoencoder. In Proceedings of the 24th ACM International Conference on Information and Knowledge Management, pages 811–820. ACM, 2015.

[15] Xue H J, Dai X, Zhang J, et al. Deep Matrix Factorization Models for Recommender Systems[C]// Twenty-Sixth International Joint Conference on Artificial Intelligence. AAAI Press, 2017.

[16] D. Kelly and J. Teevan. Implicit feedback for inferring user preference: a bibliography. In ACM SIGIR Forum, volume 37, pages 18–28. ACM, 2003.

[17] T. Joachims, L. Granka, B. Pan, H. Hembrooke, and G. Gay. Accurately interpreting clickthrough data as implicit feedback. In SIGIR’05, pages 154–161. ACM, 2005.

[18] Y. Hu, Y. Koren, and C. Volinsky. Collaborative filtering for implicit feedback datasets. In ICDM’08, pages 263–272. IEEE, 2008.

[19] R. Pan, Y. Zhou, B. Cao, N. N. Liu, R. Lukose, M. Scholz, and Q. Yang. One-class collaborative filtering. In ICDM’08, pages 502–511. IEEE, 2008.

[20] Khoshneshin M, Street W N. Collaborative filtering via Euclidean embedding[C]//Proceedings of the Fourth ACM Conference on Recommender Systems. ACM, 2010: 87-94.

[21] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. BPR: Bayesian Personalized Ranking from Implicit Feedback. In UAI 2009.

[22] Zhang S, Yao L, Tay Y, et al. Metric Factorization: Recommendation beyond Matrix Factorization[J]. 2018.

[23] X. Geng, H. Zhang, J. Bian, and T.-S. Chua. Learning image and user features for recommendation in social networks. In ICCV, pages 4274 – 4282, 2015.