CIL Project Report: Bayesian Probabilistic Mixture-Rank Matrix Factorization

Ming-Da Liu Zhang, Ard Kastrati, Erik Alexander Daxberger
Group: `make`
Department of Computer Science, ETH Zürich, Switzerland

***Abstract*—In this paper, we develop a novel collaborative filtering (CF) method based on low-rank matrix approximation (LRMA). We employ a Gaussian mixture model (GMM) to characterize user-item ratings as a mixture of LRMA models of different ranks and propose a fully Bayesian treatment of the model (hyper-)parameters. This is opposed to learning them via maximum a-posteriori (MAP) estimation, which is prone to overfitting and is often trapped in local minima. To this end, we show how the predictive distribution of our *Bayesian probabilistic mixture-rank matrix factorization* (BPMRMF) model can be efficiently computed via Markov chain Monte Carlo (MCMC). We empirically demonstrate the effectiveness of our approach compared to the state-of-the art.**

I. Introduction

Low-rank matrix approximation (LRMA) methods have witnessed tremendous popularity for collaborative filtering (CF) due to their simplicity and high accuracy [1], [2], [3].

However, while traditional LRMA methods (cf. Section II-A) use a fixed rank to approximate the rating matrix, in real-world rating matrices, e.g. Movielens or Netflix, internal submatrices of different ranks typically coexist (e.g. induced by the fact that users/items have a varying number of ratings), such that adopting a globally fixed rank for the user/item feature matrices will yield a poor approximation, resulting in suboptimal predictive performance [4].

To mitigate this issue, [4] recently proposed to use a mixture-rank matrix approximation (MRMA) model (cf. Section II-B) instead, in which user-item ratings are characterized as a mixture of LRMA models of different ranks. Since the popular approach of learning the model parameters via maximum a-posteriori (MAP) estimation is prone to overfitting if the hyperparameters controlling the generalization capacity of the model are not tuned carefully, [4] propose to estimate the parameters and hyperparameters simultaneously during training, which they realize by the iterated conditional modes (ICM) algorithm [5]. While [4] report promising empirical results, the MAP point estimates obtained via ICM turn out to be highly sensitive to initial values and often trapped in bad local optima [4].

To avoid these problems, we take inspiration from Bayesian probabilistic matrix factorization (BPMF) [6] to alternatively propose a fully Bayesian treatment of MRMA, in which model capacity is controlled automatically by integrating over all model parameters and hyperparameters, which significantly robustifies learning. To this end, we show how the analytically intractable predictive distribution of our *Bayesian probabilistic mixture-rank matrix factorization* (BPMRMF) model can be efficiently approximated using sampling-based Markov chain Monte Carlo (MCMC) methods. Since for large-scale problems, MCMC approaches are often computationally prohibitive, one of our key technical contributions lies in tackling the central challenge of divising proper prior distributions for all model hyperparameters so that sampling can be carried out efficiently (cf. Section III).

BPMRMF achieves significantly higher prediction accuracy than state-of-the-art CF techniques, which we empirically demonstrate on an item rating dataset (cf. Section IV).

II. Background and Notations

A. Low-Rank Matrix Approximation (LRMA)

Given a targeted rating matrix $R \in \mathbb{R}^{m \times n}$ over m users and n items, where $R_{i,j}$ denotes the rating of the i-th user on the j-th item, LRMA generally aims to to find two rank-k matrices $U \in \mathbb{R}^{m \times k}$ (called user feature matrix) and $V \in \mathbb{R}^{n \times k}$ (called item feature matrix), with $k \ll \min\{m, n\}$, such that $R \approx \hat{R} = UV^\top$. After obtaining U and V, rating $R_{i,j}$ can be estimated by the inner product $R_{i,j} = U_i V_j^\top$.

B. Mixture-Rank Matrix Approximation (MRMA)

Following the idea of probabilistic matrix factorization (PMF) [2], MRMA [4] employs a probabilistic model with Gaussian noise to model the ratings, inducing the following conditional distribution over the observed ratings:

$$\Pr(R|U,V,\alpha,\beta,\sigma^2) = \prod_{i=1}^{m}\prod_{j=1}^{n}\left[\sum_{k=1}^{K}\alpha_i^k\beta_j^k\mathcal{N}(R_{i,j}|U_i^k {V_j^k}^\top,\sigma^2)\right]^{\mathbb{1}_{i,j}} \tag{1}$$

where K denotes the maximum rank considered, $\alpha^k \in \mathbb{R}^m$ and $\beta^k \in \mathbb{R}^n$ are the weight vectors of the rank-k matrix approximation model for all users and items, respectively, $U^k \in \mathbb{R}^{m \times k}$ and $V^k \in \mathbb{R}^{n \times k}$ are the feature matrices of the rank-k matrix approximation model for all users and items, respectively, and $\mathbb{1}_{i,j}$ is an indicator function which is 1 if $R_{i,j}$ is observed and 0 otherwise. The authors place zero mean isotropic Gaussian priors over U^k and V^k and Laplacian priors over α^k and β^k, such that when fixing their hyperparameters, MAP estimation in the model corresponds to least squares error minimization with ℓ_2 regularization on U and V and ℓ_1 regularization on α and β.

III. Bayesian Probabilistic Mixture-Rank Matrix Factorization (BPMRMF)

In order to perform Bayesian inference, we are required to evaluate the *predictive posterior distribution* [7]

$$\Pr(R^*_{i,j}|R) = \int \Pr(R^*_{i,j}|M)\Pr(M|R)dM \qquad (2)$$

where M denotes the set of model parameters and hyperparameters. Since the posterior $\Pr(M|R)$ cannot be computed analytically, we resort to numerical approximation techniques. The main idea is to view Eq. (2) as an expectation of $\Pr(R^*_{i,j}|M)$ over the posterior distribution $\Pr(M|R)$, which can then be approximated by averaging over samples drawn from the posterior. Since $\Pr(M|R)$ is too complex to sample from directly, we employ the widely-used Markov chain Monte Carlo (MCMC) indirect sampling technique. MCMC operates by drawing a sequence of samples from some proposal distribution such that each sample only depends on the previous one, thus forming a Markov chain [8]. As described in Section I, the main technical challenge to make it efficient is to find a proper model and proper prior distributions. However, the likelihood term $\Pr(R|M) = \Pr(R|U,V,\alpha,\beta,\sigma^2)$ given in Eq. (1) has $k^{n\times m}$ parameters, which implies a large computational cost for a rather small sample size of $n \times m$. Hence, using the model in Eq. (1) to generate ratings is infeasible in practice. We thus propose a different model, as described next.

A. The model

In order to simplify the likelihood in Eq. (1), we introduce auxiliary latent indicator variables $\mathcal{C} = \{\{c_{i,j}\}_{i=1}^n\}_{j=1}^m$ which allow us to identify the mixture component that every observation was generated from. Therefore, for each data sample $R_{i,j}$, we assume the existence of a corresponding variable $c_{i,j}$ indicating the mixture component (i.e., essentially the rank) which the observation was generated from. This induces the following Gaussian mixture model (GMM):

$$\begin{aligned}&\Pr(R|U,V,\alpha,\beta,\sigma^2)\\ &= \prod_{i=1}^{m}\prod_{j=1}^{n}\left[\sum_{k=1}^{K}\Pr(c_{i,j}=k)\mathcal{N}(R_{i,j}|U_i^k V_j^{k\top},\sigma^2)\right]^{\mathbb{1}_{i,j}} \qquad (3)\end{aligned}$$

where $c_{i,j}$ follows the discrete distribution $\Pr(c_{i,j}=k) \sim \alpha_i^k \beta_j^k$. As in [4], the priors for the user and the item feature vectors are assumed to be Gaussian, i.e.,

$$U_i^k \sim \mathcal{N}(\mu_{U^k},\Lambda_{U^k}), \quad V_j^k \sim \mathcal{N}(\mu_{V^k},\Lambda_{V^k}), \quad k = 1\dots K\ .$$

The key ingredient of our fully Bayesian treatment is to view the hyperparameters also as random variables. Hence we also choose prior distributions for the *hyperparameters* (which are commonly referred to as *hyper-priors*).

As described previously, [4] place Laplacian priors over α_i and β_j to encourage sparsity. Since each indicator variable $c_{i,j}$ follows a categorical distribution, we instead define the priors of α^k and β^k to be the conjugate prior of the categorical distribution [7], namely the Dirichlet distribution[1]

$$\Pr(\alpha_i|\tau) = \mathrm{Dir}\left(\frac{\tau}{K},\dots,\frac{\tau}{K}\right) \sim \prod_{k=1}^{K}(\alpha_i^k)^{\frac{\tau}{K}-1}$$

(analogous for β_j). By using the Dirichlet distribution we can also encourage sparsity by setting the concentration parameter $\frac{\tau}{K}$ to much less than 1. In this way, the mass will be highly concentrated in a few components, leaving the rest with almost no mass.
For the Gaussian parameters

$$\Theta_U = \{\Theta_{U^k}\}_{k=1}^n, \text{where } \Theta_{U^k} = \{\mu_{U^k},\Lambda_{U^k}\}$$

(analogous for Θ_V) we choose the conjugate distributions as priors, as in [6]. Let $\Theta_0 = \{\mu_0,\beta_0,W_0,\nu_0\}$, then:

$$\begin{aligned}\Pr(\Theta_{U^k}|\Theta_0) &= \Pr(\mu_{U^k}|\Lambda_{U^k})\Pr(\Lambda_{U^k})\\ &= \mathcal{N}(\mu_{U^k}|\mu_0,(\beta_0\Lambda_{U^k})^{-1})\mathcal{W}(\Lambda_{U^k}|W_0,\nu_0)\end{aligned}$$

(analogous for Θ_{V^k}) where $\mathcal{W}$ denotes the Wishart distribution of a random $D\times D$ matrix λ with ν_0 degrees of freedom and a $D\times D$ scale matrix $\mathcal{W}_0$, i.e., $\mathcal{W}(\lambda|\mu_0,\nu_0) = \frac{1}{C}|\Lambda|^{(\nu_0-D-1)/2}\exp\left(-\frac{1}{2}\mathrm{Tr}(W_0^{-1}\Lambda)\right)$.

All these priors are chosen for mathematical convenience and interpretable expressiveness. They are conjugate priors which will allow us to analytically perform many of the marginalization steps (integrations) necessary to derive a sampler for this model.

Note that even the hyper-priors have several hyperparameters. These parameters should reflect our prior knowledge about the specific problem and are treated as constants during training. In fact, Bayesian learning is able to adjust them according to the training data, and varying their values (within in a reasonably large range) has little impact on the final prediction, as often observed in Bayesian estimation procedures [8]. They should simply encode our prior beliefs regarding class observation distribution shape and variability.

The graphical model for BPMRMF is shown in Figure 1.

B. Prediction

As in Eq. (2), the *predictive distribution* of the rating value $R^*_{i,j}$ is obtained by marginalizing over model parameters and hyperparameters:

$$\begin{aligned}\Pr(R^*_{i,j}|R) = \int &\Pr(R^*_{i,j}|U,V,\mathcal{C},\alpha,\beta,\Theta_U,\Theta_V)\\ &\Pr(U,V,\mathcal{C},\alpha,\beta,\Theta_U,\Theta_V|R)d\{U,V,\mathcal{C},\alpha,\beta,\Theta_U,\Theta_V\}\end{aligned}$$

Based on the graphical model given in Figure 1 and by using Bayes' rule, we can see that the posterior distribution has

[1]Note that the subscripts i,j correspond to the user and feature, superscript k corresponds to the k-th entry of the vectors α_i and β_j, and $\frac{\tau}{K}-1$ is the power.

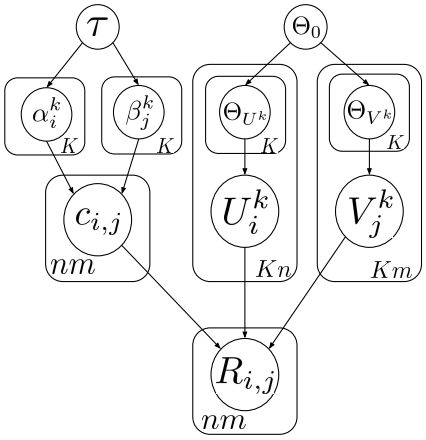

Figure 1. The graphical model corresponding to our Bayesian probabilistic mixture-rank matrix factorization (BPMRMF) method.

the following block structure:

$$\begin{aligned}&\Pr(U,V,\mathcal{C},\alpha,\beta,\Theta_U,\Theta_V|R)\sim\\&\Pr(R|U,V,\mathcal{C})\Pr(U|\Theta_U)\Pr(V|\Theta_V)\\&\Pr(\Theta_U|\Theta_0)\Pr(\Theta_V|\Theta_0)\Pr(\mathcal{C}|\alpha,\beta)\Pr(\alpha|\tau)\Pr(\beta|\tau)\ .\end{aligned}$$

As explained in the beginning of this section, our aim is to use MCMC-based methods. Hence, an Monte Carlo approximation of the above equation is given:

$$\Pr(R^*_{i,j}|R,\Theta_0)\approx\frac{1}{L}\sum_{l=1}^{L}\Pr(R^*_{i,j}|U_i^{k^{(l)}},V_j^{k^{(l)}},{c_{i,j}}^{(l)}=k) \tag{4}$$

where (l) denotes the iteration in which U_i^k, V_j^k, $c_{i,j}$ are sampled. What remains to be done is sampling these parameters, which is described in the next section.

Finally, to obtain predictions that minimize the RMSE, we use the Bayes estimator with the mean squared error (MSE) risk function, which is equivalent to the expected value of the posterior[2] (i.e., the mean of a Gaussian mixture), i.e.,

$$\hat{R}_{i,j}=\frac{1}{L}\sum_{l=1}^{L}\sum_{k=1}^{K}\alpha_i^k\beta_j^k U_i^k {V_j^k}^\top \tag{5}$$

C. Inference

One of the simplest MCMC algorithms is the Gibbs sampling algorithm, which cycles through the latent variables, sampling each from its distribution conditioned on the current values of all other variables. Our model (i.e., its block structure and our choice of priors) was chosen in particular to make Gibbs sampling straightforward. We now finally present the required conditional distributions of every latent variable conditioned on all others.

[2]One can also use the sampled $c_{i,j}$ as in Eq. (4) instead of summing over all k's. However, since we also sample α and β, we use the other method instead, following the Bayes estimator.

- *Latent features* U, V [6]:

$$\Pr(U_i^k|R,V,\mathcal{C},\Theta_U,\sigma^2)=\mathcal{N}(U_i^k|\mu^*_{i,k},\Lambda^*_{i,k})$$

where

$$\Lambda^*_{i,k}=\Lambda_{U^k}+\frac{1}{\sigma^2}\sum_{j=1}^{m}[V_jV_j^\top]^{\mathbb{1}_{i,j,k}}$$

$$\mu^*_{i,k}=[\Lambda^*_{i,k}]^{-1}(\frac{1}{\sigma^2}\sum_{j=1}^{m}[V_jR_{i,j}]^{\mathbb{1}_{i,j,k}}+\Lambda_{U^k}\mu_{U^k})$$

$\mathbb{1}_{i,j,k}$ is an indicator function of the set of all observed ratings that are currently assigned to class k, i.e. $c_{i,j}=k$. The same applies to the conditional distributions over the item features V. This is analytically easily solvable due to the conjugate prior, i.e. we chose the prior of U and V to be Gaussian, which is the conjugate prior of the Gaussian itself [7].

- *Hyperparameters of* U, V [6]:

$$\begin{aligned}&\Pr(\mu_{U^k},\Lambda_{U^k}|U^k,\Theta_0)\\&=\mathcal{N}(\mu_{U^k}|\mu^*_{0,k},(\beta^*_{0,k}\Lambda_{U^k})^{-1})\mathcal{W}(\Lambda_{U^k}|W^*_{0,k},\nu^*_{0,k})\end{aligned}$$

where

$$\mu^*_{0,k}=\frac{\beta_{0,k}\mu_{0,k}+n\bar{U}^k}{\beta_{0,k}+n},\beta^*_{0,k}=\beta_{0,k}+n,\nu^*_{0,k}=\nu_{0,k}+n$$

$$[W_0^{k^*}]^{-1}=[W_0^k]^{-1}+n\bar{S}+\frac{\beta_0 n}{\beta_0+n}(\mu_0-\bar{U}^k)(\mu_0-\bar{U}^k)^\top$$

$$\bar{U}=\frac{1}{n}\sum_{i=1}^{n}U_i^k,\bar{S}=\frac{1}{n}\sum_{i=1}^{n}U_i^k{U_i^k}^\top$$

The same applies to the hyperparameters of V. Again, this is analytically solvable since the Gaussian-Wishart distribution is the conjugate prior of the multivariate normal distribution [7].

- *Class probabilities* α, β

$$\Pr(\alpha_i|\mathcal{C},\beta,\tau)=\mathrm{Dir}(\bar{\tau}^1,\ldots,\bar{\tau}^K)$$

where $\{\bar{\tau}^k=\frac{\frac{\tau}{K}+c_i^k}{\tau+n}\}_{k=1}^K$, and c_i^k is the number of ratings C_{ij} in category k for all $\{c_{ij}\}_{j=1}^m$. This, again, is analytically solvable due to the Dirichlet distribution being the conjugate prior of the categorical distribution [7]. The same applies to β.

- *The classes*

$$\begin{aligned}&\Pr(c_{i,j}=k|R_{i,j},U_i,V_j,\sigma^2,\alpha_i,\beta_j)\\&=\Pr(R_{i,j}|c_{i,j},U,V,\sigma^2)\Pr(c_{i,j}|\alpha,\beta,\sigma^2)\\&=\exp\left((R_{ij}-U_i^k{V_j^k}^T)/\sigma^2\right)\alpha_i^k\beta_j^k\ .\end{aligned}$$

We can as well easily sample from this distribution by simply enumerating all values $c_{i,j}$ could take, normalizing, and then sampling from this discrete distribution.

Table I
COMPARISON OF RMSE AND RUNTIME BETWEEN BPMRMF AND FIVE STATE-OF-THE-ART CF ALGORITHMS.

Algorithm	Val. RMSE	Test RMSE	Time (sec)
SVD++	1.0071	1.0206	1900.36
NMF	1.0007	1.0133	34.94
SVD	0.9965	1.0119	**27.22**
SGD	1.1239	1.0022	160.01
BPMF	0.9800	0.9810	262.37
BPMRMF	**0.9780**	**0.9783**	444.24

Table II
COMPARISON OF RMSE AND RUNTIME BETWEEN BPMRMF MODELS WITH DIFFERENT SETS OF RANKS.

Ranks	Val. RMSE	Test RMSE	Time (sec)
$[10, 50]$	0.9804	0.9809	401.18
$[8, 9, 10, 50]$	0.9780	0.9783	444.24
$[2, 4, 6, 8, 10, 12]$	0.9802	0.9798	584.71
$[50, 60, 70, 80, 90, 100]$	0.9784	0.9793	2480.99
$[5, 6, 7, 8, 9, 10, 20, \ldots, 100]$	0.9779	0.9775	4178.45

IV. EXPERIMENTS

We now empirically demonstrate the effectiveness of our approach[3] as compared to the state-of-the-art. The training dataset we use for the assessment consists of 1M ratings from 10,000 users on 1,000 items. All ratings are integer values between 1 and 5. For validation purposes, we randomly split the training data into two sets, an actual training set comprising of 90% of the given data, and a validation set comprising of the remaining 10%. There is also a test set containing another 1,176,952 user/item pairs with the ratings withheld; the test set was not available to us, performance on it was evaluated via a Kaggle competition[4]. As a performance metric, we use the prediction error as measured by the root mean squared error (RMSE).

A. Accuracy Comparison

Table I compares the rating prediction accuracy between BPMRMF and five popular matrix approximation-based collaborative filtering algorithms[5], namely matrix approximation via stochastic gradient descent (SGD)[6], regularized singular value decomposition (SVD) [10], non-negative matrix factorization (NMF) [11], SVD with additional usage of bias and implicit data (SVD++) [12], and Bayesian probabilistic matrix factorization (BPMF)[7] [6]. For BPMRMF, we use $[8, 9, 10, 50]$ as the set of ranks due to reasons of efficiency, meaning that the prediction accuracy is not optimal. However, as shown in Table I, BPMRMF still significantly outperforms all other considered methods. This is likely because BPMRMF considers an ensemble of LRMA models, therefore adopting different rank approximations for different users/items, which not only allows for more suitable modeling of individual users or items, but also achieves a globally more accurate approximation. This confirms our claim that internal submatrices of different ranks indeed coexist in typical user-item rating matrices, which traditional fixed-rank matrix approximations fail to capture.

B. Effect of Considered Ranks in BPMRMF

Clearly, the performance of BPMRMF is determined by the set of ranks used for the underlying LRMAs. However, it is neither efficient nor necessary to consider all ranks in $[1, 2, \ldots, K]$, such that considering a subset of ranks is typically sufficient in practice. In this experiment, we investigate the trade-off between computational efficiency and predictive accuracy of different rank combinations.

Table II shows the result of our investigation, where we considered 5 different sets of ranks. We can observe that the RMSE typically decreases when more ranks are considered, which is intuitive since more ranks can model different users/items more appropriately. However, the computational effort required also significantly increases with the number of ranks considered. In practice, it is thus desirable to choose a moderate number of ranks, to achieve an optimal trade-off between efficiency and accuracy. Also, neither sets of all small ranks (e.g., $[2, 4, \ldots, 12]$) nor sets of all large ranks (e.g. $[50, 60, \ldots, 100]$) achieve an RMSE as low as that of sets containing both small *and* large ranks (e.g. $[8, 9, 10, 50]$). This demonstrates that a model with all small ranks *underfits* users/items with many ratings, while one with all large ranks *overfits* those with few ratings, such that only a mixture of both can achieve an optimal fit.

V. CONCLUSION

In this paper we proposed Bayesian probabilistic mixture-rank matrix factorization (BPMRMF), which describes user-item ratings using a mixture of LRMA models of different ranks to achieve better approximation and thus higher recommendation accuracy. We proposed a fully Bayesian treatment of the BPMRMF model parameters and hyperparameters, instead of adopting the suboptimal MAP learning approach typically employed. We demonstrated how MCMC methods can be exploited to efficiently approximate the predictive distribution of our model. Our experimental results showed that BPMRMF can achieve higher accuracy than state-of-the-art matrix approximation-based CF methods, further pushing the frontier of recommender systems.

[3]We provide our implementation of BPMRMF, including instructions on how to run it, at https://gitlab.ethz.ch/lming/make.

[4]All reported scores were achieved on the *private* test set. The competition can be found at https://www.kaggle.com/c/cil-collab-filtering-2018/.

[5]We use the Surprise library [9] for implementations of SVD++, NMF and SVD, and implemented SGD and BPMF ourselves. We tuned the parameters of all methods using 10-fold cross-validation, except for SVD++, where we kept the default parameters due to its high computational demand. Since SVD++ requires very careful tuning of its parameters, which is computationally prohibitive, it most of the time doesn't perform that well.

[6]This corresponds to the method described in Problem 3 of Exercise 4.

[7]Note that for a fixed rank k, the corresponding rank-k model in BPMRMF is equivalent to a rank-k BPMF model.

REFERENCES

[1] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, no. 8, pp. 30–37, 2009.

[2] A. Mnih and R. R. Salakhutdinov, "Probabilistic matrix factorization," in *Advances in neural information processing systems*, 2008, pp. 1257–1264.

[3] J. Lee, S. Kim, G. Lebanon, and Y. Singer, "Local low-rank matrix approximation," in *International Conference on Machine Learning*, 2013, pp. 82–90.

[4] D. Li, C. Chen, W. Liu, T. Lu, N. Gu, and S. Chu, "Mixture-rank matrix approximation for collaborative filtering," in *Advances in Neural Information Processing Systems*, 2017, pp. 477–485.

[5] J. Besag, "On the statistical analysis of dirty pictures," *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 259–302, 1986.

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

[7] A. Gelman, J. B. Carlin, H. S. Stern, D. B. Rubin, A. Vehtari, and D. B. Dunson, *Bayesian data analysis*, 2014.

[8] X. Liang, C. Xi, H. Tzu-Kuo, S. Jeff, and G. C. Jaime, "Temporal collaborative filtering with bayesian probabilistic tensor factorization."

[9] N. Hug, "Surprise, a Python library for recommender systems," http://surpriselib.com, 2017.

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

[11] X. Luo, M. Zhou, Y. Xia, and Q. Zhu, "An efficient non-negative matrix-factorization-based approach to collaborative filtering for recommender systems," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 2, pp. 1273–1284, May 2014.

[12] 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*. ACM, 2008, pp. 426–434.