An Efficient Non-Negative Matrix-Factorization-Based Approach to Collaborative Filtering for Recommender Systems

Xin Luo, Mengchu Zhou, *Fellow*, *IEEE*, Yunni Xia, *Member*, *IEEE*, and Qingsheng Zhu

***Abstract*—Matrix-factorization (MF)-based approaches prove to be highly accurate and scalable in addressing collaborative filtering (CF) problems. During the MF process, the non-negativity, which ensures good representativeness of the learnt model, is critically important. However, current non-negative MF (NMF) models are mostly designed for problems in computer vision, while CF problems differ from them due to their extreme sparsity of the target rating-matrix. Currently available NMF-based CF models are based on matrix manipulation and lack practicability for industrial use. In this work, we focus on developing an NMF-based CF model with a single-element-based approach. The idea is to investigate the non-negative update process depending on each involved feature rather than on the whole feature matrices. With the non-negative single-element-based update rules, we subsequently integrate the Tikhonov regularizing terms, and propose the regularized single-element-based NMF (RSNMF) model. RSNMF is especially suitable for solving CF problems subject to the constraint of non-negativity. The experiments on large industrial datasets show high accuracy and low-computational complexity achieved by RSNMF.**

***Index Terms*—Collaborative filtering (CF), non-negative matrix-factorization (NMF), recommender system, single-element-based approach, Tikhonov regularization.**

I. Introduction

THE RAPID expansion of world-wide-web has brought people huge convenience as well as causing a serious problem of efficiency, due to the difficulty to filter desired information out of billions of bytes. Recommender systems, which can play the role of intelligent agents assisting people to locate preferences, become indispensable under such context. Started in early 1990s, research on recommender systems has shown great efficiency in solving the problem of information overload by connecting information actively, rather than passively, to potentially related people according to their information usage history [1]–[5].

A recommender system is based on three fundamental kinds of entities including users, items (e.g., movies, news), and user-item usage history (e.g., scores, comments). The main task is to figure out useful patterns describing the relationship between users and items from user-item usage history, and then make predictions for possible user-item links according to these patterns. To deal with this task, many models were proposed during the past 20 years. These models can be further grouped into several categories, among which a very important branch is collaborative filtering (CF) [1]–[3].

Inside a CF-based recommender, a user-item rating-matrix is usually the fundamental data source, where each entry is modeled according to the corresponding user-item usage history with high values usually denoting strong user-item preferences. Since only a finite item set can be operated by each user, this rating-matrix is usually very sparse with a mass of missing values. On the other hand, if these missing values are estimated appropriately, it is feasible to link people with their potential favorites. Therefore, CF transforms the problem of personalized-recommendation into missing-data-estimation, i.e., to estimate the unknown ratings based on known ones subject to globally high accuracy and other requirements [1]–[8].

The CF can be implemented through several approaches. The up-to-date progress in this area unveils that matrix-factorization (MF)-based models are highly accurate and scalable for many cases [1]–[3]. Their idea is to construct a low-rank estimate to the original rating-matrix. An MF-based model works by mapping both the items and users into the same latent feature space, training desired user/item features on existing ratings, and then generating predictions for unknown ratings heavily relying on the inner products of related user-item feature-vector pairs. Because of the usual low-rank of the rating-matrix caused by its extreme sparsity, the dimension of the latent feature space can be set low without impairing accuracy. Hence, the size of feature matrices in an MF-based model is linearly related to user and item counts [1]–[3]; whereby, the storage complexity is low and related issues are easy to resolve for real applications. Further, with a properly designed loss function, the prediction accuracy of MF-based models is very competitive [1]–[3], [9]–[19].

The singular value decomposition (SVD)-based recommender [9] is the earliest model of this kind. To carry out a standard SVD process, this model requires prefilling the unknown ratings artificially with the rating-data statistics (e.g., the global rating

Manuscript received September 26, 2013; revised January 11, 2014; accepted February 20, 2014. Date of publication February 26, 2014; date of current version May 02, 2014. This work was supported in part by the National Natural Science Foundation of China under Grant 61202347, Grant 61272194, and Grant 61103036; in part by the U.S. National Science Foundation under Grant CMMI-1162482; in part by the Fundamental Research Funds for the Central Universities under Grant CDJZR12180012; and in part by the Natural Science Foundation Project of CQ CSTC under Grant cstc2012jjA40016 and Grant cstc2012jjA40002. Paper no. TII-13-0680.

X. Luo, Y. Xia, and Q. Zhu are with the College of Computer Science, Chongqing University, Chongqing 400044, China (e-mail: luoxin21@cqu.edu.cn; xiayunni@hotmail.com; qszhu@cqu.edu.cn).

M. Zhou is with the Department of Electrical and Computer Engineering, New Jersey Institute of Technology, Newark, NJ 07102 USA (e-mail: zhou@njit.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TII.2014.2308433

average). The obtained low-dimensional matrices form a low-rank estimate of the original rating-matrix. Then, the unknown ratings in the original rating-matrix are predicted by the corresponding entries in this low-rank estimate. In a CF problem, the rank of the user-item rating-matrix is usually low and much less than its either dimension, due to its extreme sparsity. Therefore, the feature matrices in the SVD-based model share a low dimension which is decided by the low rank, and their storage complexity is low and far less than that of relationship-based models [7]–[9]. In other words, the SVD-based model has an advantage in storage efficiency, which is essential for many industrial applications.

Several MF-based techniques have been proposed and successfully applied to CF recommenders. Hofmann employs probabilistic latent semantic analysis to build a CF model [10]; Srebro *et al.* propose the maximum-margin MF-based model [11]; Kurucz *et al.* propose the expectation maximization (EM)-based MF model [12]. At the Netflix-prize-competition, the CF recommender based on regularized MF [13] has shown high performance. Since then, MF-based models have caused more attention, and several models have been developed and gained industrial applications. The representative works include biased SVD model [14], biased regularized incremental simultaneous MF model [15], SVD $++$ model [3], [16], probabilistic MF model [17], and nonparametric MF model [18]. Moreover, the idea of MF-based CF has also been employed to many related areas, e.g., social recommendation in social network service [19], content-based image tagging in image processing [20], QoS prediction in service computing [21], [22], video re-indexing in information retrieval [23], and mobile-user tracking in wireless sensor networks [24].

In general, MF-based recommenders are built through an unsupervised training process, where a global loss function is designed to model the difference between known ratings and their corresponding entries in a desired low-rank estimate. The parameters such as user/item features, which are used to form this low-rank estimate, are iteratively updated to fit the training data. During this training process, the constraint of non-negativity proves to be very important, since it can enable the learnt features to represent the actual meanings such as the user interests and community tendency more precisely [25]–[30].

By introducing the non-negative constraint into an MF process, Lee *et al.* [25], [26] first propose the algorithm of non-negative matrix-factorization (NMF). It is initially designed for learning part-based object representations in computer vision [25], [26], [31], [32]; and then applied to CF problems [27]–[30]. Essentially, NMF fulfills the constraint of non-negativity through adapting learning-rate; the learning-rate is rescaled to cancel out the negative components during the update of each parameter, and thus only the non-negative components remain. With the premise of convergence, the initially non-negative parameters can fit the training set as well as maintaining non-negativity after such a training process.

Unlike problems of computer vision, CF problems come with the characteristic of sparsity; the known entries in the rating-matrix are usually far less than the unknown ones. Therefore, the standard NMF process fails to deal with them without proper tuning. Suggested by Zhang *et al.* [27], this adaptation can be implemented through introducing either an EM procedure or indicator matrix into the NMF process. However, either strategy will lead to high computational complexity. Moreover, for industrial applications, frequent manipulations of matrices are lack of practicability [33], [34]. On the other hand, Tikhonov regularization has proved to be highly reliable in improving the accuracy of MF-based CF models [3], [13]–[18]; therefore, it is quite meaningful and significant to integrate this regularization strategy into the NMF-based CF model, which has been never seen in any prior research to our best knowledge.

In this work, we focus on developing an NMF-based CF model with a single-element-based approach, which can achieve high computational efficiency and prediction accuracy, as well as easy implementation. We first investigate the updated process depending on each involved feature rather than on the whole feature matrices, and then integrate Tikhonov regularization into this model to improve the prediction accuracy while ensuring non-negativity.

The rest of this paper is organized as follows. Section II gives the preliminaries, Section III states our methods, Section IV gives the experiments and results, and finally, Section V concludes this paper.

II. Preliminaries

In CF recommenders, historical user behaviors are usually modeled into a user-item rating-matrix, as defined in [1]–[3].

Definition 1: Given an item set I and a user set U, a *user-item rating-matrix* $\boldsymbol{R}$ is a $|U| \times |I|$ matrix where each element $r_{u,i}$ is proportional to user u's preference on item i.

Thus, the problem of CF can be regarded as a process of missing-data-estimation [1]–[3].

Definition 2: Let R_K and R_W denote the known and whole entry sets in R, respectively. Given $T \subseteq R_K$ as the training dataset, a *CF problem* is to construct an estimator $\hat{R}$ to generate the prediction $\hat{r}_{u,i}$ for each entry $(u,i) \in R_W$ and approximate $\min\left(\sum_{(u,i)\in R_W} |r_{u,i} - \hat{r}_{u,i}|\right)$.

Similar to the other MF-based CF models, NMF also factorizes the original rating-matrix $\boldsymbol{R}$ into two rank-f matrices $\boldsymbol{P}$ and $\boldsymbol{Q}$; where $\boldsymbol{P}$ is $|U| \times f$, $\boldsymbol{Q}$ is $f \times |I|$ and $f \ll \min(|U|, |I|)$. Note that $\boldsymbol{P}$ and $\boldsymbol{Q}$ are interpreted as the user and item feature matrices that reflect the user and item characteristics contained in the rating data. This factorization process is implemented by minimizing the cost function measuring the difference between $\boldsymbol{R}$ and the rank-f estimate $\boldsymbol{PQ}$, e.g., the Euclidean distance or the Kullback–Leibler divergence [25], [26]. However, NMF carries out this factorization process subject to the non-negative constraint, i.e., $\boldsymbol{P}, \boldsymbol{Q} \geq 0$. Thus, with the loss function by Euclidean distance, the problem of NMF-based CF is given as follows [25], [26]:

$$\arg\min_{\boldsymbol{P},\boldsymbol{Q}} \underset{\boldsymbol{R}\leftrightarrow\boldsymbol{PQ}}{d} = \|\boldsymbol{R} - \boldsymbol{PQ}\|^2 \text{ s.t. } \boldsymbol{P}, \boldsymbol{Q} \geq 0. \quad (1)$$

This constrained-optimization problem is hard to solve in practice; yet NMF addresses this problem via manipulating the learning-rate skillfully. First, without a non-negative constraint,

the problem (1) can be solved by conventional optimization algorithms [25], [26]. For instance, with gradient decent, each desired parameter is obtained via the following training process [25], [26]:

$$\begin{aligned} p_{u,k} &\leftarrow p_{u,k} + \eta_{u,k}((\boldsymbol{R}\boldsymbol{Q}^T)_{u,k} - (\boldsymbol{P}\boldsymbol{Q}\boldsymbol{Q}^T)_{u,k}) \\ q_{k,i} &\leftarrow q_{k,i} + \eta_{k,i}((\boldsymbol{P}^T\boldsymbol{R})_{k,i} - (\boldsymbol{P}^T\boldsymbol{P}\boldsymbol{Q})_{k,i}). \end{aligned} \tag{2}$$

In (2), $p_{u,k}$ denotes the entry at the uth row and kth column in $\boldsymbol{P}$; $q_{k,i}$ denotes the entry at the kth row and ith column in $\boldsymbol{Q}$; these two parameters actually denote the kth feature of user u and the kth feature of item i, respectively. $\eta_{u,k}$ and $\eta_{k,i}$ denote the corresponding learning-rate. Given $p_{u,k}$ and $q_{k,i}$ initially non-negative, during the training process they may lose the non-negativity due to the negative component in (2). To maintain the non-negativity of $\boldsymbol{P}$ and $\boldsymbol{Q}$, NMF diagonally manipulate the learning-rate as follows [25], [26]:

$$\eta_{u,k} = \frac{p_{u,k}}{(\boldsymbol{P}\boldsymbol{Q}\boldsymbol{Q}^T)_{u,k}}, \quad \eta_{k,i} = \frac{q_{k,i}}{(\boldsymbol{P}^T\boldsymbol{P}\boldsymbol{Q})_{k,i}}. \tag{3}$$

With this transformation, the negative components in (2) are canceled out with the initial values of $p_{u,k}$ and $q_{k,i}$; thereby, the learning process is reformulated as follows:

$$p_{u,k} \leftarrow p_{u,k}\frac{(\boldsymbol{R}\boldsymbol{Q}^T)_{u,k}}{(\boldsymbol{P}\boldsymbol{Q}\boldsymbol{Q}^T)_{u,k}}, \quad q_{k,i} \leftarrow q_{k,i}\frac{(\boldsymbol{P}^T\boldsymbol{R})_{k,i}}{(\boldsymbol{P}^T\boldsymbol{P}\boldsymbol{Q})_{k,i}}. \tag{4}$$

With (4), the training process will also converge, as proved in [26]. This non-negative training process is an essential part of NMF. Its original form and extensions have shown high performance in computer vision [25], [26], [31], [32]. However, in a CF problem, the rating-matrix $\boldsymbol{R}$ is usually very sparse with huge number of unknown entries. Consequently, the update-rule (4) should be revised correspondingly. In [27], Zhang *et al.* suggest introducing an indicator matrix into the update process to fit the CF problem, which results in the weighted NMF (WNMF) model with the update rule given by

$$\begin{aligned} p_{u,k} &\leftarrow p_{u,k}\frac{((\boldsymbol{W}\circ\boldsymbol{R})\boldsymbol{Q}^T)_{u,k}}{((\boldsymbol{W}\circ\boldsymbol{P}\boldsymbol{Q})\boldsymbol{Q}^T)_{u,k}}, \\ q_{k,i} &\leftarrow q_{k,i}\frac{(\boldsymbol{P}^T(\boldsymbol{W}\circ\boldsymbol{R}))_{k,i}}{(\boldsymbol{P}^T(\boldsymbol{W}\circ PQ))_{k,i}} \end{aligned} \tag{5}$$

where $\boldsymbol{W}$ is a $|U| \times |I|$ matrix, of which the element is equal to 1 if the corresponding entry in $\boldsymbol{R}$ is known and 0 otherwise; symbol $\circ$ denotes the Hadamard product between two matrices. This strategy can deal with the sparse nature of the CF problem; however, it results in high computational cost. First, the integration of the indicator matrix brings more computational cost since Hadamard multiplications are additionally required. Second, the matrix multiplications are also taken on the whole rating-matrix $\boldsymbol{R}$, which is filled with sparse meaningful values and a huge number of zeros. Third, during the training process, it is required to iterate on each entry of $\boldsymbol{R}$ to check if it is equal to zero or otherwise.

In the following section, we will introduce a novel single-element-based approach to NMF-based CF model. The reason why we choose to design such a model is to exclude multiplications between large matrices; thus, the designed model can be applied to large datasets with high dimensions in real industrial applications. Furthermore, we will integrate Tikhonov regularization to achieve high-prediction accuracy.

III. A Single-Element-Based Approach

A. Single-Element-Based NMF for CF Problems

As discussed in [3], [13]–[18], when one considers the rating-matrix estimate in a CF problem, the Euclidean distance between the original rating-matrix $\boldsymbol{R}$ and low-rank estimate $\boldsymbol{P}\boldsymbol{Q}$ is equal to the sum squared error between the known entries in $\boldsymbol{R}$ and the corresponding estimates in $\boldsymbol{P}\boldsymbol{Q}$, given by

$$\begin{aligned} \underset{\boldsymbol{R}\leftrightarrow\boldsymbol{P}\boldsymbol{Q}}{d} &= \|\boldsymbol{R}-\boldsymbol{P}\boldsymbol{Q}\|^2 \approx \underset{\boldsymbol{R}\leftrightarrow\boldsymbol{P}\boldsymbol{Q}}{\varepsilon} \\ &= \sum_{(u,i)\in R_K}(r_{u,i} - p_{u,\cdot}q_{\cdot,i})^2 = \sum_{(u,i)\in R_K}\left(r_{u,i} - \sum_{k=1}^{f} p_{u,k}q_{k,i}\right)^2 \end{aligned} \tag{6}$$

where $p_{u,\cdot}$ denotes the uth row-vector of $\boldsymbol{P}$, $q_{\cdot,i}$ denotes the ith column-vector of $\boldsymbol{Q}$, respectively. Note that the last step of (6) is to expand the product between the one-row-matrix $p_{u,\cdot}$ and the one-col-matrix $q_{\cdot,i}$.

Therefore, employing gradient decent to seek for the optimal solution minimizing $\underset{\boldsymbol{R}\leftrightarrow\boldsymbol{P}\boldsymbol{Q}}{d}$, is equivalent to train each desired feature parameter $p_{u,k}$ and $q_{k,i}$ to minimize $\underset{\boldsymbol{R}\leftrightarrow\boldsymbol{P}\boldsymbol{Q}}{\varepsilon}$ in (6) [3], [13]–[18]. Note that in (6) only the known ratings and their corresponding estimates are considered. Hence, when we train $p_{u,k}/q_{k,i}$ we only have to consider the known rating set by user u/on item i. Therefore, an additive update-rule for this training process is derived as follows:

$$\begin{aligned} &\underset{p_{u,k},q_{k,i},(u,i)\in R_K}{\arg\min} \underset{\boldsymbol{R}\leftrightarrow\boldsymbol{P}\boldsymbol{Q}}{\varepsilon} \\ &\Rightarrow \begin{cases} p_{u,k} \leftarrow p_{u,k} - \eta_{u,k}\dfrac{\partial\varepsilon}{\partial p_{u,k}} \\ \qquad = p_{u,k} - 2\eta_{u,k}\left(-\displaystyle\sum_{i\in I_u} q_{k,i}\left(r_{u,i} - \sum_{k=1}^{f} p_{u,k}q_{k,i}\right)\right) \\ q_{k,i} \leftarrow q_{k,i} - \eta_{k,i}\dfrac{\partial\varepsilon}{\partial q_{k,i}} \\ \qquad = q_{k,i} - 2\eta_{k,i}\left(-\displaystyle\sum_{u\in U_i} p_{u,k}\left(r_{u,i} - \sum_{k=1}^{f} p_{u,k}q_{k,i}\right)\right) \end{cases} \end{aligned} \tag{7}$$

where I_u and U_i denote the item set rated by user u and the user set having rated item i. Note that (7) works by moving each involved feature to the opposite direction of the gradient to minimize $\underset{\boldsymbol{R}\leftrightarrow\boldsymbol{P}\boldsymbol{Q}}{\varepsilon}$. Since $\eta_{u,k}$ and $\eta_{k,i}$ in (7) are positive learning-rate-constants, with some notational abuse we treat $\eta_{u,k}$ and $\eta_{k,i}$ as the original $2\eta_{u,k}$ and $2\eta_{k,i}$, respectively, to lead to a concise form. $\sum_{k=1}^{f} p_{u,k}q_{k,i}$

denotes the estimate to $r_{u,i}$ and can be simplified by $\hat{r}_{u,i}$. Thus, (7) is reformulated to

$$\underset{p_{u,k},q_{k,i},(u,i)\in R_K}{\arg\min} \underset{\boldsymbol{R}\leftrightarrow\boldsymbol{PQ}}{\varepsilon} \Rightarrow \begin{cases} p_{u,k} \leftarrow p_{u,k} + \eta_{u,k} \sum\limits_{i\in I_u} q_{k,i}(r_{u,i} - \hat{r}_{u,i}) \\ q_{k,i} \leftarrow q_{k,i} + \eta_{k,i} \sum\limits_{u\in U_i} p_{u,k}(r_{u,i} - \hat{r}_{u,i}). \end{cases} \tag{8}$$

In the single-element-based update-rule (8), we can apply the principle of NMF to manipulate the learning-rate $\eta_{u,k}$ and $\eta_{k,i}$, for canceling out the negative components to maintain the non-negativity. Obviously, in (8) the negative components are $-\eta_{u,k} \sum\limits_{i\in I_u} q_{k,i}\hat{r}_{u,i}$ for $p_{u,k}$ and $-\eta_{k,i} \sum\limits_{u\in U_i} p_{u,k}\hat{r}_{u,i}$ for $q_{k,i}$. Hence, we can cancel them with the initial state of $p_{u,k}$ and $q_{k,i}$ in (8) by setting $\eta_{u,k} = \frac{p_{u,k}}{\sum\limits_{i\in I_u} q_{i,k}\hat{r}_{u,i}}$ and $\eta_{k,i} = \frac{q_{k,i}}{\sum\limits_{u\in U_i} p_{u,k}\hat{r}_{u,i}}$ to fulfill the non-negative constraint. After this adjustment, (8) is reformulated to

$$\underset{p_{u,k},q_{k,i},(u,i)\in R_K}{\arg\min} \underset{\boldsymbol{R}\leftrightarrow\boldsymbol{PQ}}{\varepsilon} \text{ s.t. } \boldsymbol{P},\boldsymbol{Q} \geq 0$$

$$\Rightarrow \begin{cases} p_{u,k} \leftarrow p_{u,k} + \dfrac{p_{u,k}}{\sum\limits_{i\in I_u} q_{i,k}\hat{r}_{u,i}} \left(\sum\limits_{i\in I_u} q_{k,i} r_{u,i} - \sum\limits_{i\in I_u} q_{k,i}\hat{r}_{u,i} \right) \\ \qquad = p_{u,k} \dfrac{\sum\limits_{i\in I_u} q_{k,i} r_{u,i}}{\sum\limits_{i\in I_u} q_{i,k}\hat{r}_{u,i}} \\ q_{k,i} \leftarrow q_{k,i} + \dfrac{q_{k,i}}{\sum\limits_{u\in U_i} p_{u,k}\hat{r}_{u,i}} \left(\sum\limits_{u\in U_i} p_{u,k} r_{u,i} - \sum\limits_{u\in U_i} p_{u,k}\hat{r}_{u,i} \right) \\ \qquad = q_{k,i} \dfrac{\sum\limits_{u\in U_i} p_{u,k} r_{u,i}}{\sum\limits_{u\in U_i} p_{u,k}\hat{r}_{u,i}}. \end{cases} \tag{9}$$

With update-rule (9) we can implement the NMF process to deal with a sparse rating-matrix $\boldsymbol{R}$ easily. Note that this single-element-based NMF (SNMF) model is functionally equivalent to that of WNMF in (5); however, the indicator matrix $\boldsymbol{W}$ is no longer required since the whole process is directly conducted on the known rating set; and thus the computational complexity is reduced significantly.

B. Integrating Tikhonov Regularization Into NMF-Based CF

Tikhonov regularization is a widely-employed strategy in machine learning tasks for regularizing the ill-posed problems to obtain high accuracy. In a CF problem, it proves to be useful in improving both prediction accuracy and convergence rate [3], [13]–[18]. Due to the sparse nature of the rating-matrix $\boldsymbol{R}$, when considering regularizing an MF-based CF model, a usual choice of regularizing terms is the Frobenius norm of feature matrices [3], [13]–[18]. Therefore, after integrating regularizing terms, the objective of NMF-based CF is changed into

$$\underset{\boldsymbol{P},\boldsymbol{Q}}{\arg\min} \underset{\boldsymbol{R}\leftrightarrow\boldsymbol{PQ}}{d_T} = \|\boldsymbol{R} - \boldsymbol{PQ}\|^2 + \lambda_{\boldsymbol{P}}\|\boldsymbol{P}\|_F^2 + \lambda_{\boldsymbol{Q}}\|\boldsymbol{Q}\|_F^2 \quad \text{s.t. } \boldsymbol{P},\boldsymbol{Q} \geq 0 \tag{10}$$

where $\lambda_{\boldsymbol{P}}$ and $\lambda_{\boldsymbol{Q}}$ are positive constant denoting the regularizing coefficients for $\boldsymbol{P}$ and $\boldsymbol{Q}$ respectively; $\|\cdot\|_F$ denotes the Frobenius norm of a matrix. Similar to (6), the problem (10) can be also transformed into sum squared error on R_K, given by

$$\begin{aligned} \underset{\boldsymbol{R}\leftrightarrow\boldsymbol{PQ}}{d_T} &= \|\boldsymbol{R} - \boldsymbol{PQ}\|^2 + \lambda_{\boldsymbol{P}}\|\boldsymbol{P}\|_F^2 + \lambda_{\boldsymbol{Q}}\|\boldsymbol{Q}\|_F^2 \\ &\approx \underset{\boldsymbol{R}\leftrightarrow\boldsymbol{PQ}}{\varepsilon_T} = \sum_{(u,i)\in R_K} ((r_{u,i} - p_{u,\cdot}q_{\cdot,i})^2 + \lambda_{\boldsymbol{P}}\|p_u\|_F^2 \\ &\quad + \lambda_{\boldsymbol{Q}}\|q_i\|_F^2) \\ &= \sum_{(u,i)\in R_K} \left(\left(r_{u,i} - \sum_{k=1}^{f} p_{u,k}q_{k,i} \right)^2 + \lambda_{\boldsymbol{P}} \sum_{k=1}^{f} p_{u,k}^2 \right. \\ &\quad \left. + \lambda_{\boldsymbol{Q}} \sum_{k=1}^{f} q_{k,i}^2 \right). \end{aligned} \tag{11}$$

Similar to (7), the additive update-rule for constrained optimization problem (11) without the non-negative constraint is given by

$$\underset{p_{u,k},q_{k,i},(u,i)\in R_K}{\arg\min} \underset{\boldsymbol{R}\leftrightarrow\boldsymbol{PQ}}{\varepsilon_T}$$

$$\Rightarrow \begin{cases} p_{u,k} \leftarrow p_{u,k} - \eta_{u,k} \cdot 2 \left(- \sum\limits_{i\in I_u} q_{k,i} \left(r_{u,i} - \sum\limits_{k=1}^{f} p_{u,k}q_{k,i} \right) \right. \\ \qquad \left. + \lambda_{\boldsymbol{P}} p_{u,k} \right) \\ q_{k,i} \leftarrow q_{k,i} - \eta_{k,i} \cdot 2 \left(- \sum\limits_{u\in U_i} p_{u,k} \left(r_{u,i} - \sum\limits_{k=1}^{f} p_{u,k}q_{k,i} \right) \right. \\ \qquad \left. + \lambda_{\boldsymbol{Q}} q_{k,i} \right). \end{cases} \tag{12}$$

Similar to (8), we reformulate (12) into

$$\begin{aligned} &\underset{p_{u,k},q_{k,i},(u,i)\in R_K}{\arg\min} \underset{\boldsymbol{R}\leftrightarrow\boldsymbol{PQ}}{\varepsilon_T} \\ &\Rightarrow p_{u,k} \leftarrow p_{u,k} + \eta_{u,k} \sum_{i\in I_u} (q_{k,i}(r_{u,i} - \hat{r}_{u,i}) - \lambda_P p_{u,k}) \\ &\qquad = p_{u,k} + \eta_{u,k} \sum_{i\in I_u} q_{k,i} r_{u,i} - \eta_{u,k} \sum_{i\in I_u} (q_{k,i}\hat{r}_{u,i} + \lambda_P p_{u,k}) \\ &\quad q_{k,i} \leftarrow q_{k,i} + \eta_{k,i} \sum_{u\in U_i} (p_{u,k}(r_{u,i} - \hat{r}_{u,i}) - \lambda_Q q_{k,i}) \\ &\qquad = q_{k,i} + \eta_{k,i} \sum_{u\in U_i} p_{u,k} r_{u,i} - \eta_{k,i} \sum_{u\in U_i} (p_{u,k}\hat{r}_{u,i} + \lambda_Q q_{k,i}). \end{aligned} \tag{13}$$

In (13) the negative components are $-\eta_{u,k} \sum\limits_{i\in I_u} (q_{k,i}\hat{r}_{u,i} + \lambda_{\boldsymbol{P}} p_{u,k})$ for $p_{u,k}$ and $-\eta_{k,i} \sum\limits_{u\in U_i} (p_{u,k}\hat{r}_{u,i} + \lambda_{\boldsymbol{Q}} q_{k,i})$ for $q_{k,i}$; then

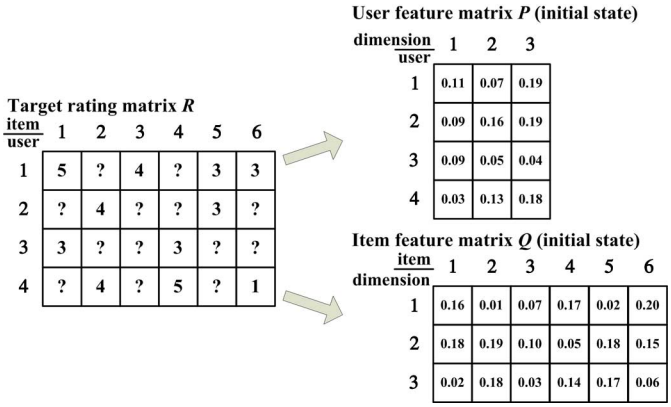


Fig. 1. Illustrative example for algorithm RSNMF.

following the principle of NMF, we set $\eta_{u,k} = \frac{p_{u,k}}{\sum_{i\in I_u}(q_{k,i}\hat{r}_{u,i}+\lambda_{\boldsymbol{P}}p_{u,k})}$ and $\eta_{k,i} = \frac{q_{k,i}}{\sum_{u\in U_i}(p_{u,k}\hat{r}_{u,i}+\lambda_{\boldsymbol{Q}}q_{k,i})}$ as

$$\arg\min_{p_{u,k},q_{k,i},(u,i)\in R_K} \varepsilon_{T \boldsymbol{R}\leftrightarrow\boldsymbol{PQ}} \text{ s.t. } \boldsymbol{P},\boldsymbol{Q}\geq 0$$

$$\Rightarrow \begin{cases} p_{u,k} \leftarrow p_{u,k} + p_{u,k}\dfrac{\sum_{i\in I_u}(q_{k,i}r_{u,i} - (q_{k,i}\hat{r}_{u,i}+\lambda_{\boldsymbol{P}}p_{u,k}))}{\sum_{i\in I_u}(q_{k,i}\hat{r}_{u,i}+\lambda_{\boldsymbol{P}}p_{u,k})} \\ \quad = p_{u,k}\dfrac{\sum_{i\in I_u} q_{k,i}r_{u,i}}{\sum_{i\in I_u}(q_{k,i}\hat{r}_{u,i}+\lambda_{\boldsymbol{P}}p_{u,k})} \\ \quad = p_{u,k}\dfrac{\sum_{i\in I_u} q_{k,i}r_{u,i}}{|I_u|\lambda_{\boldsymbol{P}}p_{u,k}+\sum_{i\in I_u} q_{k,i}\hat{r}_{u,i}} \\ q_{k,i} \leftarrow q_{k,i} + q_{k,i}\dfrac{\sum_{u\in U_i}(p_{u,k}r_{u,i} - (p_{u,k}\hat{r}_{u,i}+\lambda_{\boldsymbol{Q}}q_{k,i}))}{\sum_{u\in U_i}(p_{u,k}\hat{r}_{u,i}+\lambda_{\boldsymbol{Q}}q_{k,i})} \\ \quad = q_{k,i}\dfrac{\sum_{u\in U_i} p_{u,k}r_{u,i}}{\sum_{u\in U_i}(p_{u,k}\hat{r}_{u,i}+\lambda_{\boldsymbol{Q}}q_{k,i})} \\ \quad = q_{k,i}\dfrac{\sum_{u\in U_i} p_{u,k}r_{u,i}}{|U_i|\lambda_{\boldsymbol{Q}}q_{k,i}+\sum_{u\in U_i} p_{u,k}\hat{r}_{u,i}}. \end{cases} \quad (14)$$

With (14), we expect to implement an NMF-based CF model with higher accuracy compared to the nonregularized version. Based on the analysis in Section III, we summarize the work flow of the proposed regularized single-element-based NMF (RSNMF) model in Algorithm RSNMF.

Note that in Algorithm RSNMF, we employ four auxiliary matrices, including $|U| \times f$ matrices *UserUP* and *UserDOWN*, and $f \times |I|$ matrices *ItemUP* and *ItemDOWN*, to cache the interim results. With this strategy, we can simply iterate over the known rating set R_K to obtain high computing efficiency.

Fig. 1 illustrates a toy example. The target rating-matrix $\boldsymbol{R}$ is a 4×6 matrix with many unknown entries, corresponding to the ratings by four users on six items. RSNMF is selected to build an MF-based recommender. The dimension of the latent feature space is set at three. Thus the desired user and feature matrices are 4×3 and 3×6, respectively. First, the user and item features are non-negatively initialized with random values. Then to carry out Algorithm NMF, the regularizing coefficients $\lambda_{\boldsymbol{P}}$ and $\lambda_{\boldsymbol{Q}}$ are set as positive constants, e.g., 0.06 for both; and auxiliary matrices *UserUP* and *UserDOWN* are initialized as 4×3 ones, *ItemUP* and *ItemDOWN* are initialized as 3×6 ones, in consistence with $\boldsymbol{P}$ and $\boldsymbol{Q}$.

Algorithm RSNMF

Process starts.

1. Initialize feature matrices P and Q non-negatively.
   Initialize feature dimension f.
   Initialize $\lambda_{\boldsymbol{P}}$, $\lambda_{\boldsymbol{Q}}$.
   Initialize *UserUP*, *UserDown*, *ItemUP*, *ItemDown*.
   Initialize training round $\mathrm{t} = 0, \max - \mathrm{training} - \mathrm{round} = \mathrm{n}$.
2. **while not** converge && $t \leq max - training - round$ **do**
   set *UserUP*, *UserDown*, *ItemUP*, $ItemDown = 0$.
3. **for** each rating $r_{u,i}$ in R_K
   $\hat{r}_{u,i} = \sum_{k=1}^{f} p_{u,k}q_{k,i}$;
4. **for** $k = 1$ to f
   $UserUP_{u,k} = UserUP_{u,k} + q_{k,i}r_{u,i}$.
   $UserDOWN_{u,k} = UserDOWN_{u,k} + q_{k,i}\hat{r}_{u,i}$.
   $ItemUP_{k,i} = ItemUP_{k,i} + p_{u,k}r_{u,i}$.
   $ItemDOWN_{k,i} = ItemDOWN_{k,i} + p_{u,k}\hat{r}_{u,i}$.
   end for
   end for
5. **for** each involved user u
6. **for** $k = 1$ to f
   $UserDOWN_{u,k} = UserDOWN_{u,k} + |I_u|\lambda_P p_{u,k}$.
   update$p_{u,k} = p_{u,k} \cdot (UserUP_{u,k}/UserDOWN_{u,k})$.
   end for
   end for
7. **for** each involved item i
8. **for** $k = 1$ to f
   $ItemDOWN_{k,i} = ItemDOWN_{k,i} + |U_i|\lambda_Q q_{k,i}$.
   update$q_{k,i} = q_{k,i} \cdot (ItemUP_{k,i}/ItemDOWN_{k,i})$.
   end for
   end for
   $t = t + 1$.
   end while

Process ends.

Before each training epoch of RSNMF, the auxiliary matrices should be reset with zeroes for recording the update of each feature. Then the known entry set R_K is traversed to record the corresponding feature update in auxiliary matrices according to (14). For instance, in the first training epoch, for feature $p_{2,1}$, it is related to ratings $r_{2,2}$ and $r_{2,5}$. Then during the traversal, the estimates $\hat{r}_{2,2}$ and $\hat{r}_{2,5}$ are computed as 0.0655 and 0.0629 according to (8); and $UserUP_{2,1}$ and $UserDOWN_{2,1}$ are computed as

$$\begin{aligned} UserUP_{2,1} &= q_{1,2}r_{2,2} + q_{1,5}r_{2,5} \\ &= 0.01 \times 4 + 0.02 \times 3 = 0.1 \\ UserDOWN_{2,1} &= q_{1,2}\hat{r}_{2,2} + q_{1,5}\hat{r}_{2,5} \\ &= 0.01 \times 0.0655 + 0.02 \times 0.0629 = 0.001913. \end{aligned} \tag{15}$$

Since user 2 has rated two items in $\boldsymbol{R}$, we have $|I_2| = 2$. Then the trained results for $p_{2,1}$ is given by

$$\begin{aligned} p_{2,1} &= (p_{2,1} \times UserUP_{2,1})/(UserDOWN_{2,1} + |I_2|\lambda_{\boldsymbol{P}} p_{2,1}) \\ &= (0.17 \times 0.1)/(0.001913 + 2 \times 0.06 \times 0.09). \end{aligned} \tag{16}$$

By analyzing Algorithm RSNMF, we obtain the following theorem.

Theorem 1: Given $|R_K| \gg \max\{|U|, |I|\}$, the asymptotic computational complexity of RSNMF is $\Theta(|R_K| \times t \times f)$.

Proof: First of all, we summarize the asymptotic complexity of each step in Algorithm RSNMF in Table I.

Based on Table I, we formulate the computational complexity of Algorithm RSNMF by

$$\begin{aligned} &T_{RSN} \\ &_{|R_K| \gg \max\{|U|,|I|\}} \\ &= t[2\Theta(|U| \times f) + 2\Theta(|I| \times f) + |R_K|(\Theta(f) + 4f \times \Theta(1)) \\ &\quad + (|U| \times f + |I| \times f)\Theta(1)] + 3\Theta(|U| \times f) \\ &\quad + 3\Theta(|I| \times f) + 5\Theta(1) \\ &= 5\Theta(t \times |R_K| \times f) + 3\Theta(t \times |U| \times f) + 3\Theta(t \times |I| \times f) \\ &\quad + 4\Theta(|R_K| \times f) + 3\Theta(|U| \times f) + 3\Theta(|I| \times f) + 5\Theta(1) \\ &\approx \Theta(|R_K| \times t \times f). \end{aligned} \tag{17}$$

Note that with $|R_K| \gg \max\{|U|, |I|\}$, the last step of (16) is by reasonably ignoring the lower-order-terms and constant coefficient in the computational complexity of Algorithm RSNMF.

IV. Experiments and Results

A. General Settings

1) Evaluation Metrics: Recommenders can be evaluated from several aspects, like the prediction accuracy and prediction coverage [35], [36]. In this work, we mainly care about the closeness of predictions to the actual ratings, since it can directly reflect whether or not the model has captured the essential characteristics of given data. Hence, we choose the root mean squared error (RMSE) and normalized mean absolute error (NMAE) as the evaluation metrics. RMSE is a widely used

TABLE I
Complexity Analysis of Algorithm RSNMF

Operation	Complexity
Initialize P non-negatively	$\Theta(\|U\|\times f)$
Initialize Q non-negatively	$\Theta(f\times\|I\|)$
Initialize f	$\Theta(1)$
Initialize λ_P	$\Theta(1)$
Initialize λ_Q	$\Theta(1)$
Initialize *UserUP*	$\Theta(\|U\|\times f)$
Initialize *UserDOWN*	$\Theta(\|U\|\times f)$
Initialize *ItemUP*	$\Theta(f\times\|I\|)$
Initialize *ItemDOWN*	$\Theta(f\times\|I\|)$
Initialize $t = 0$	$\Theta(1)$
Initialize *max-training-round* = n	$\Theta(1)$
while not converge && $t \le$ *max-training-round* do	t*
Set *UserUP* = 0	$\Theta(\|U\|\times f)$
Set *UserDOWN* = 0	$\Theta(\|U\|\times f)$
Set *ItemUP* = 0	$\Theta(f\times\|I\|)$
Set *ItemDOWN* = 0	$\Theta(f\times\|I\|)$
for each rating $r_{u,i}$ in R_K	$\|R_K\|$*
$\hat{r}_{u,i} = \sum_{k=1}^{f} p_{u,k}q_{k,i}$	$\Theta(f)$
for k = 1 to f	f*
$UserUP_{u,k} = UserUP_{u,k} + q_{k,i}r_{u,i}$	$\Theta(1)$
$UserDOWN_{u,k} = UserDOWN_{u,k} + q_{k,i}\hat{r}_{u,i}$	$\Theta(1)$
$ItemUP_{k,i} = ItemUP_{k,i} + p_{u,k}r_{u,i}$	$\Theta(1)$
$ItemDOWN_{k,i} = ItemDOWN_{k,i} + p_{u,k}\hat{r}_{u,i}$	$\Theta(1)$
end for	–
end for	–
for each involved user u	$\|U\|$*
for k = 1 to f	f*
$UserDOWN_{u,k} = UserDOWN_{u,k} + \|I_u\|\lambda_P p_{u,k}$	$\Theta(1)$
end for	–
end for	–
for each involved item i	$\|I\|$*
for k = 1 to f	f*
$ItemDOWN_{k,i} = ItemDOWN_{k,i} + \|U_i\|\lambda_Q q_{k,i}$	$\Theta(1)$
end for	–
end for	–
$t = t + 1$.	$\Theta(1)$
end while	–

metric for evaluating the statistical accuracy of recommenders [35], [36], which is given below

$$\text{RMSE} = \sqrt{\left(\sum_{(u,i)\in V} (r_{u,i} - \hat{r}_{u,i})^2\right) \Big/ |V|}$$

where V denotes the validation set. For a given recommender, low RMSE stands for high-prediction accuracy. Similar to RMSE, NMAE is also a frequently employed metric measuring the absolute difference between the target dataset and the prediction set as follows [35], [36]:

$$\text{NMAE} = \left(\sum_{(u,i)\in V} |r_{u,i} - \hat{r}_{u,i}|_{\text{abs}}\right) \Big/ |V|$$

where $|\cdot|_{\text{abs}}$ denotes the absolute value. Note that we use this symbol to distinguish from the cardinality of a set $|\cdot|$.

We are also caring about the improvement in computational efficiency brought by the proposed single-element-based update-rule. We simply use the consumed time of each training epoch to denote the computational efficiency of each tested model. Note

that all tested models are implemented in MATLAB R2010B, and tested on a PC Server with a 2.5 GHz CPU and 16 GB Memory.

2) Tested Models: Three models are included in our experiment. First, the WNMF model [27] is implemented and tested as benchmark. Note that we choose WNMF as benchmark is because it is a widely adopted model which can enable the NMF process in a CF problem [27]–[30], [37]–[41]. Then, the proposed SNMF and RSNMF are compared with WNMF to validate if our strategies can bring positive effect on performance. Two phenomena are expected during the experiments.

1) With the single-element-based strategy of SNMF, the training time consumed by each training epoch should be shorter than that of WNMF.
2) After the integration of Tikhonov regularization, RSNMF should achieve higher prediction accuracy and computational efficiency compared to WNMF.

3) Datasets: Three datasets are employed in our experiments, as listed below.

a) D1: the MovieLens 1M dataset. MovieLens is a movie recommender-system developed by the GroupLens research group at the University of Minnesota. It collects the ratings by users on their watched movies to form the user-item rating-matrix, and then predicts the unknown entries based on known ones inside this matrix to recommend users with movies they have never seen and might be interested in.

   D1 contains 1 000 209 ratings applied to 3900 movies by 6040 users with a rating scale on the [0, 5], interval. The rating density of D1 is 4.25% only, as depicted in Fig. 2. During the experiment, we employ RSNMF to check whether the single-element-based approach and Tikhonov regularization can bring high efficiency and accuracy under the constraint of non-negativity.

b) D2: the Jester 1.1M dataset. D2 is collected through the joke-recommender Jester [42], and contains 1 186 324 continuous ratings in the scale of $[-10.00, 10.00]$ from 24 983 users on 100 jokes. Its rating density is 47.32%, which is relatively very high in CF problems.

c) D3: a subset of the Dating Agency dataset. The Dating Agency dataset is collected through the online dating recommender by Charles University. D3 is a subset of this dataset, and consists of 3 727 555 ratings in the scale of [1, 10] by 30 000 users on 126 048 profiles. D3 has the rating density of 0.09% only, which is very low compared to that of the other three datasets.

For keeping the experimental results comparable on all four datasets, we have mapped the ratings of D2 and D3 into the scale of [0, 5]. To obtain objective and unbiased results, we have employed the 80%–20% train-validation settings and 5-fold cross-validation technique.

B. Experimental Process

Since in RSNMF the regularizing coefficients λ_P and λ_Q control the extent of regularization, they surely affect the performance in prediction accuracy and convergence rate. Therefore, we first have conducted experiments for validating the effect of regularizing coefficients. On all four datasets, we have validated the performance of RSNMF with λ_P and λ_Q increasing from 0.01 to 0.15 stepping at 0.01. Note that for accelerating the searching process, we let $\lambda_P = \lambda_Q$. On each dataset, RSNMF is trained for 1000 epochs, and the RMSE varying tendency is recorded.

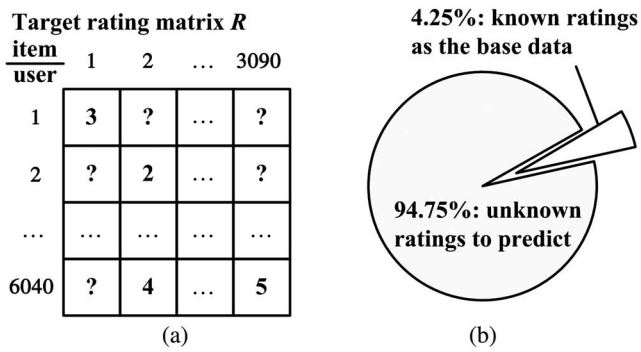


Fig. 2. Illustration of D1. (a) Rating-matrix of D1. (b) Pie diagram of known and unknown ratings.

After that, the involved three models are trained and compared on all four datasets. As proposed in [14]–[16], MF-based recommenders can be more accurate and efficient by integrating the user and item biases. Hence, during the test we employ this strategy for all three involved models. For a fair comparison, the training process of each model contains 1000 training epochs on each dataset. The RMSE changing tendency along with the average running time is recorded.

C. Results and Discussion

For RSNMF, the impact on prediction accuracy by λ_P and λ_Q on all four datasets is depicted in Fig. 3. We also depict the lowest RMSE of RSNMF on each dataset with different λ_P and λ_Q in Fig. 4. From both figures, we see that λ_P and λ_Q have vital effect on prediction accuracy of RSNMF.

On D1, after integrating the Tikhonov regularizing terms, RMSE of RSNMF continues to decrease with more training epochs; however, with too small or too big values of λ_P and λ_Q, RMSE deceasing rate becomes low. As shown in Fig. 3(a), with $\lambda_P = \lambda_Q = 0.01$ or $\lambda_P = \lambda_Q = 0.15$, the RMSE decreasing rate of RSNMF is apparently lower than in the case with $\lambda_P = \lambda_Q = 0.04$ or $\lambda_P = \lambda_Q = 0.12$. From Fig. 4(a), we see that with $\lambda_P = \lambda_Q = 0.06$, RSNMF obtains the lowest RMSE at 0.8532 on D1. In contrast, with $\lambda_P = \lambda_Q = 0.01$ and $\lambda_P = \lambda_Q = 0.15$, the best RMSE is at 0.8915 and 0.8877, respectively; which is about 4.49% and 4.04% higher than that in the case with $\lambda_P = \lambda_Q = 0.06$. The difference in prediction accuracy is quite significant. This phenomenon demonstrates that for RSNMF the regularizing coefficients should be chosen carefully to achieve high-prediction accuracy. For experiment on D2, the situation is very similar to that on D1, as shown in Figs. 3 and 4.

However, the situation is a bit different on D3. From Fig. 4(c), we find with $\lambda_P = \lambda_Q = 0.12$, RSNMF can obtain the lowest RMSE at 0.9684 on D3. However, this best RMSE is not significantly lower than that at 0.9703 with $\lambda_P = \lambda_Q = 0.15$. But compared to the RMSE at 1.0210 in the case with $\lambda_P = \lambda_Q = 0.01$, the best RMSE is about 5.43% lower. Note

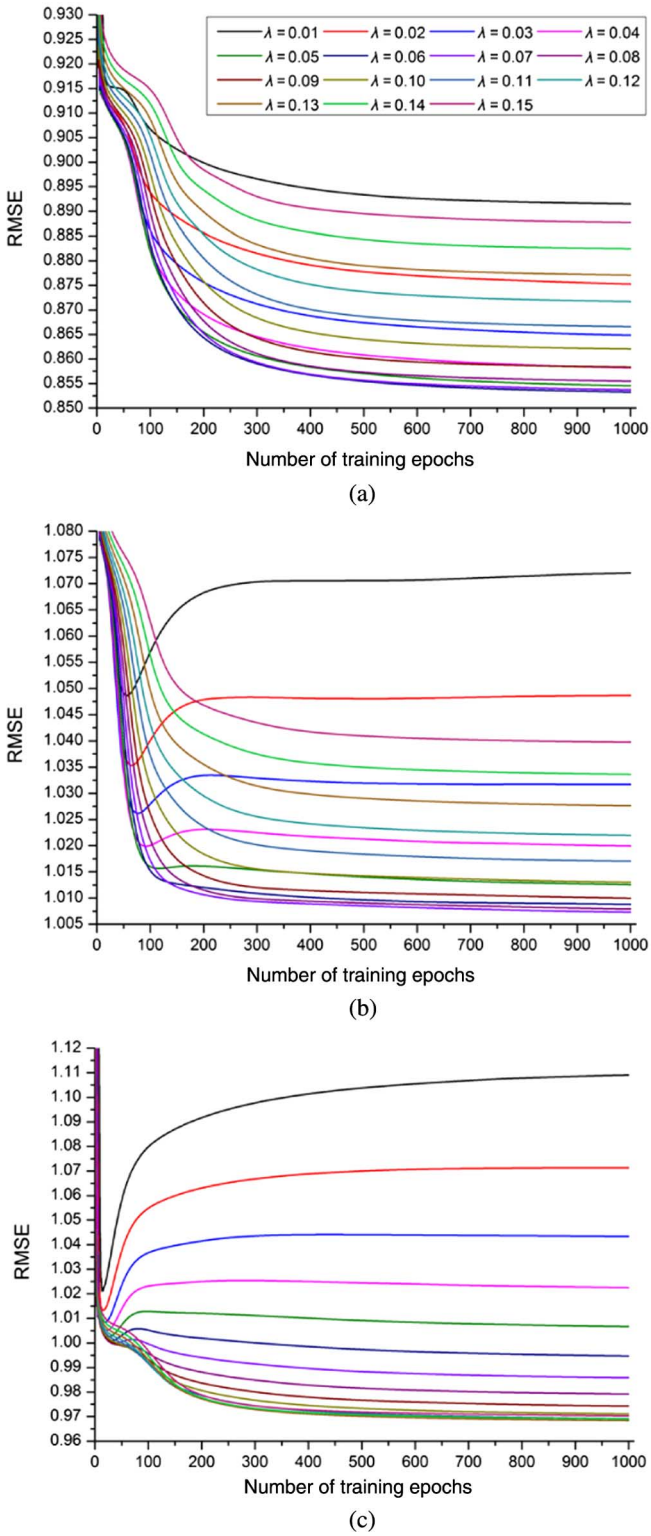


Fig. 3. Training process of RSNMF with different values of λ_P and λ_Q: (a) D1; (b) D2; and (c) D3.

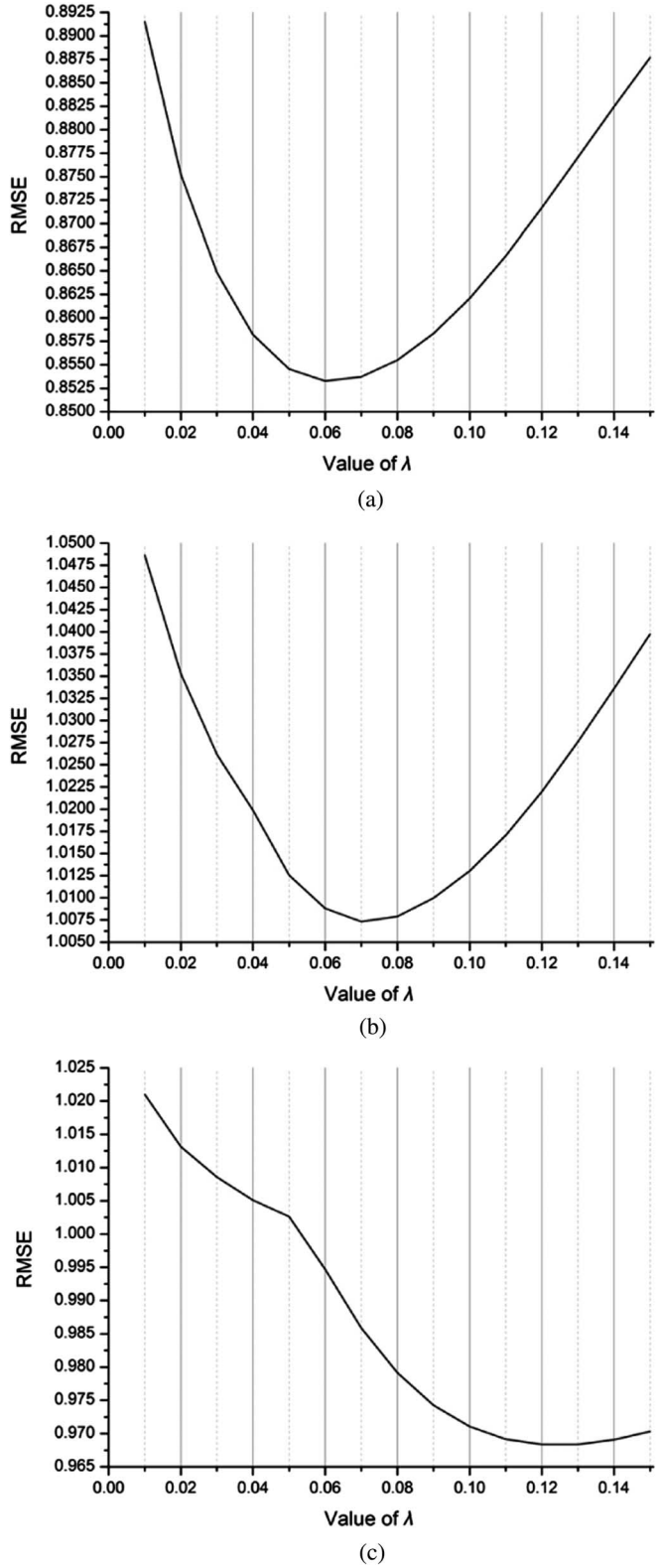


Fig. 4. Lowest RMSE of RSNMF with different values of λ_P and λ_Q: (a) D1; (b) D2; and (c) D3.

that when employing NMAE, the tuning process of λ_P and λ_Q is nearly the same and will result in the same optimal values; therefore, we present only the tuning process by RMSE.

During the tuning process of λ_P and λ_Q, we have identified another interesting phenomenon that for RSNMF the best choice for λ_P and λ_Q seems to be related to the size of the training data. Generally, with more training data, the optimal λ_P and λ_Q grow bigger. However, λ_P and λ_Q do not seem connecting with the rating density. For instance, the optimal values of λ_P and λ_Q on D1 and D2 are rather close, but D2's rating density is more than ten times that of D1's rating density.

TABLE II
FINAL SETTINGS OF REGULARIZING COEFFICIENTS FOR RSNMF

Dataset	λ_P and λ_Q
D1	0.06
D2	0.07
D3	0.12

Based on Figs. 3 and 4, we have picked the optimized regularizing coefficients for RSNMF on each dataset to draw the accuracy comparison between RSNMF, SNMF, and WNMF. These optimal values are summarized in Table II. Note that the tuning process of λ_P and λ_Q is a tedious task, which is unacceptable for online industrial applications. This problem can be addressed via offline tuning settings, or adopting empirical values. Naturally, the ideal way is to implement the auto-adaptation of the regularizing coefficients without introducing any sizable computational overhead. This clearly requires our future investigations.

Under these settings, the performance of RSNMF, SNMF, and WNMF are compared. The accuracy comparison of RSNMF, SNMF, and WNMF in RMSE and NMAE are depicted in Figs. 5 and 6, respectively. The lowest RMSE and NMAE by each model during the test are summarized in Tables III and IV. And the average time-consumption by one training epoch of each model is summarized in Table V. From these experimental results, we have the following findings.

1) The prediction accuracy of SNMF and WNMF are very close on all four datasets, as depicted in Figs. 5 and 6. And these two models also have the same converging rate, as shown in Tables III and IV. This is reasonable since they have the very similar training process in nature. WNMF fill the missing values in the original rating-matrix with zeros via drawing the Hadamard production between the indicator matrix $\boldsymbol{W}$ and rating-matrix $\boldsymbol{R}$; these zero-valued entries will not affect the training of each feature. On the other hand, SNMF does not take the missing values into consideration, and iterate on the known entries only. Therefore, same for these two models, only the known entries act on the training process. Meanwhile, the training rules of SNMF and WNMF also have the same effect, except for that they are in different mathematic forms of single-element-based operation and matrix manipulation. Hence, on the same training data, SNMF and WNMF obtain very similar training results and close accuracy.

   However, credited to the single-element-based approach, SNMF has much higher computational efficiency than WNMF does. This can be clearly seen from Table V.

   On D1, each training epoch of SNMF consumes 5376 ms on average, which is about 57% that of WNMF. On the other three datasets, the situation is very similar. This is because SNMF only requires to carry out the necessary updating operation connected to each existing entries, without the need to check if the current entry is nonzero standing for given ratings or otherwise. Hence, we can conclude that compared to WNMF, SNMF can maintain the prediction accuracy and non-negativity, as well as achieving higher computational efficiency.

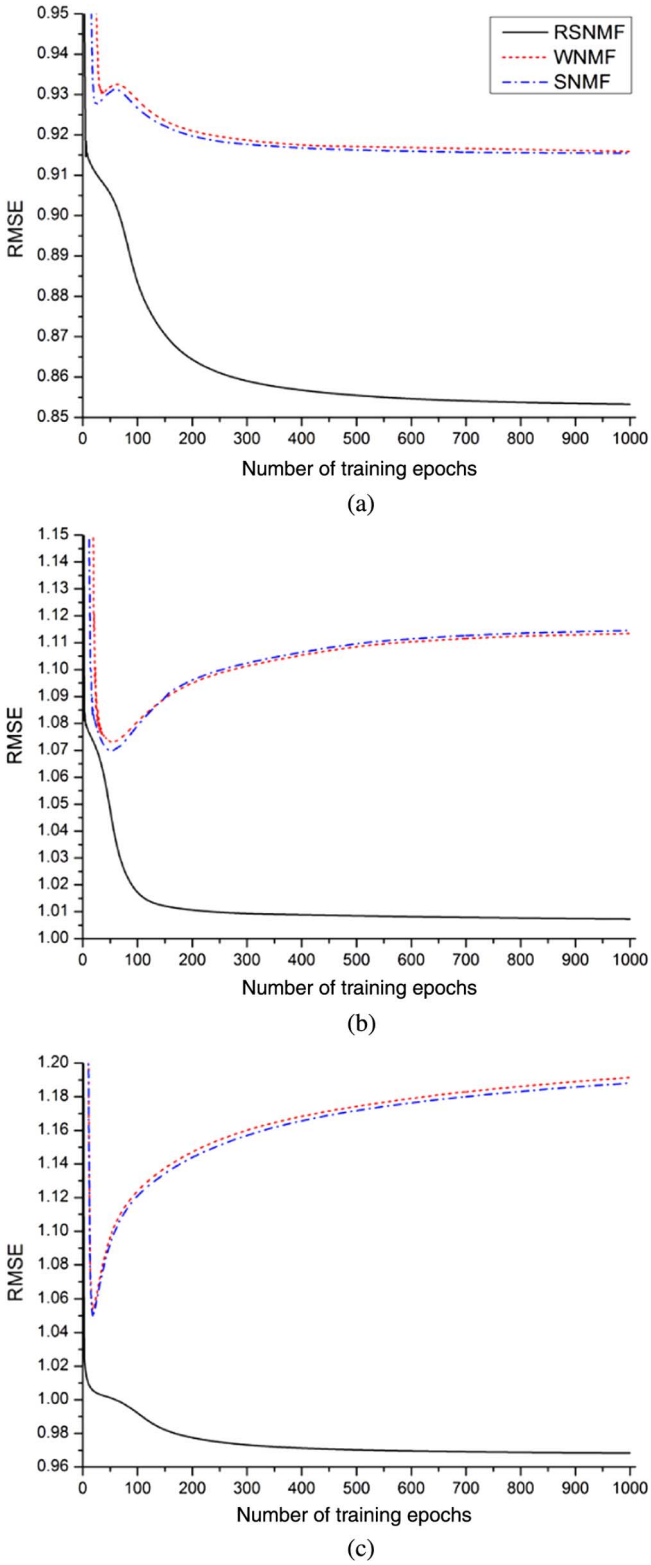


Fig. 5. RMSE comparison of RSNMF, SNMF, and WNMF: (a) D1; (b) D2; and (c) D3.

2) After integrating the Tikhonov regularizing terms, the RMSE and NMAE of RSNMF continue decreasing with more training epochs, as shown in Figs. 5 and 6. Therefore, the lowest RMSE and NMAE of RSNMF always occur with 1000 training epochs during our experiment. However, with regard to SNMF and WNMF, their RMSE and

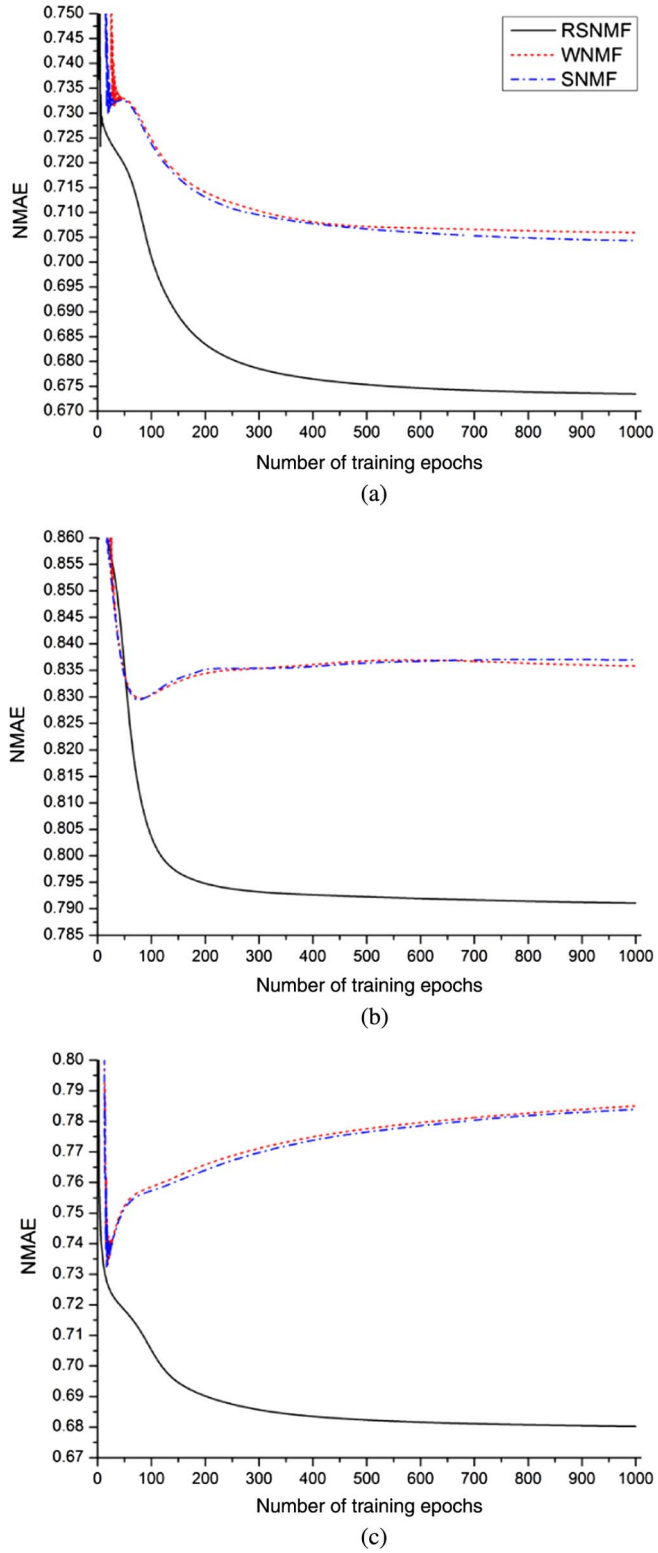


Fig. 6. NMAE comparison of RSNMF, SNMF, and WNMF: (a) D1; (b) D2; and (c) D3.

NMAE will increase after reaching the converging point, and thus their lowest RMSE and NMAE appears much earlier than those of RSNMF does. Hence, for drawing a fair comparison, in Tables III and IV we have also recorded the RMSE and NMAE of RSNMF at the converging point of SNMF and WNMF.

TABLE III
LOWEST RMSE OF RSNMF, SNMF, AND WNMF

Dataset	Lowest RMSE/training epochs		
	RSNMF	SNMF	WNMF
D1	0.8533/1000 (0.8556/490)	0.9158/490	0.9172/610
D2	1.0073/1000 (1.0479/50)	1.0699/50	1.0731/54
D3	0.9684/1000 (1.0058/18)	1.0498/18	1.0538/18

TABLE IV
LOWEST NMAE OF RSNMF, SNMF, AND WNMF

Dataset	Lowest NMAE/training epochs		
	RSNMF	SNMF	WNMF
D1	0.6734/1000	0.7043/1000	0.7060/1000
D2	0.7910/1000 (0.8124/77)	0.8294/79	0.8297/83
D3	0.6802/1000 (1.0058/18)	0.7320/20	0.7329/21

TABLE V
AVERAGE TIME CONSUMPTION FOR ONE TRAINING EPOCH OF RSNMF, SNMF, AND WNMF

Dataset	Consumed time for each epoch (ms)		
	RSNMF	SNMF	WNMF
D1	5671	5376	9433
D2	5783	5513	9716
D3	27 069	26 343	46 537

From Fig. 5 and Table III, we see that after integrating Tihkonov regularizing terms, RSNMF gains obvious advantage in prediction accuracy compared to WNMF and SNMF. On D1, the lowest RMSE of RSNMF is at 0.8533, which is about 6.82% lower than that of SNMF and 6.97% lower than that of WNMF. Meanwhile, RSNMF's RMSE at the converging point of SNMF is at 0.8556, still lower than the lowest RMSE of SNMF and WNMF. On the other three datasets, the situation is very similar.

3) When employing NMAE as the evaluation metric, RSNMF can also obtain obvious advantages compared with WNMF and SNMF, as shown in Fig. 6 and Table IV. For instance, on D1, the lowest NMAE of RSNMF is at 0.6734, which is about 4.39% lower than that of SNMF and 4.62% lower than that of WNMF. Note that the accuracy improvement brought by regularizing in NMAE is not as significant as in RMSE. This is because RMSE is more sensitive to the differences between two datasets. Due to the same reason, we see that WNMF and SNMF always need more epochs to obtain the lowest NMAE. Nonetheless, RSNMF can clearly outperform WNMF and SNMF in prediction accuracy when measured by either metric. This indicates that when addressing CF problems with NMF, the integration of Tikhonov regularization will bring strongly positive effect on improving the prediction accuracy while maintaining the non-negativity.
4) Will the integration of Tikhonov regularization impair the computational efficiency significantly? Fortunately, this does not happen, as recorded in Table V. RSNMF only requires a bit more time for each training epoch than SNMF

does. This is reasonable, since by comparing (9) and (14) we find that the update of each parameter in RSNMF only requires one more operation than that in SNMF, which takes the initial value of each parameter as the penalty factor. Therefore, the computational efficiency of RSNMF and SNMF are very close, as verified by the experimental results.

Based on the experimental results, we conclude that the employment of the single-element-based approach clearly improves the computational efficiency. The novel integration of Tikhnov regularization into the proposed SNMF results in much higher prediction accuracy, in context of NMF-based CF, with insignificant computational overhead. As shown in the experiment, RSNMF especially suits to address sparse matrices in context of recommender systems. Moreover, it can be extended to other related areas, e.g., user-side QoS prediction in service computing [21], [22], and other areas with the need of sparse-matrix-analysis subject to non-negativity.

V. Conclusion

This work focuses on developing NMF-based CF with a single-element-based approach. The main idea is to replace the loss function of standard distance with sum-squared-error, and investigate a non-negative update process depending on each involved feature parameter rather than on the whole feature matrices. With the single-element-based approach, we subsequently integrate the Tikhonov regularizing terms to improve the prediction accuracy under the premise of non-negativity. Based on these strategies, the SNMF and RSNMF are proposed, which are especially designed for CF problems. The experimental results on four large, industrial datasets well demonstrate that SNMF can obtain advantage in computational efficiency over the WNMF, an NMF-based CF model. Further, owing to the combination of single-element-based approach and Tikhonov regularization, RSNMF can outperform WNMF in terms of both computational efficiency and prediction accuracy. In addition, the single-element-based approach eases implementation of both SNMF and RSNMF, which is an important feature welcomed by industrial applications. As analyzed in the paper, the regularizing coefficients are vital to the prediction accuracy of RSNMF. However, the tuning process may be time-consuming and tedious. Thus, their automatic adjustment is an important issue in our future work. Another interesting study is to extend RSNMF with other loss functions such that more powerful solutions can be established for some significant industrial applications.

References

[1] G. Adomavicius and A. Tuzhilin, "Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions," *IEEE Trans. Knowl. Data Eng.*, vol. 17, no. 6, pp. 734–749, Jun. 2005.

[2] G. Adomavicius and Y. Kwon, "Improving aggregate recommendation diversity using ranking-based techniques," *IEEE Trans. Knowl. Data Eng.*, vol. 24, no. 5, pp. 896–911, May 2012.

[3] Y. Koren and R. Bell, "Advances in collaborative-filtering," in *Recommender Systems Handbook*, F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, Eds. New York, NY, USA: Springer, 2011, pp. 145–186.

[4] Y. Li, B. Cao, L. Xu, J. Yin, S. Deng, Y. Yin *et al.*, "An efficient recommendation method for improving business process modeling," *IEEE Trans. Ind. Informat.*, vol. 10, no. 1, pp. 502–513, Feb. 2014.

[5] J. de la Rosa, N. Hormazabal, S. Aciar, G. Lopardo, A. Trias, and M. Montaner, "A negotiation-style recommender based on computational ecology in open negotiation environments," *IEEE Trans. Ind. Electron.*, vol. 58, no. 6, pp. 2073–2085, Jun. 2011.

[6] Y. Cai, H. Leung, Q. Li, H. Min, J. Tang, and J. Li, "Typicality-based collaborative filtering recommendation," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 3, pp. 766–779, Mar. 2014.

[7] J. Breese, D. Heckerman, and C. Kadie, "Empirical analysis of predictive algorithms for collaborative-filtering," in *Proc. 14th Int. Conf. Uncertainty Artif. Intell.,* San Francisco, CA, USA, 1998, pp. 43–52.

[8] B. Sarwar, G. Karypis, J. Konstan, and J. Reidl, "Item based collaborative-filtering recommendation algorithms," in *Proc. 10th Int. Conf. World Wide Web,* Hong Kong, China, 2001, pp. 285–295.

[9] B. Sarwar, G. Karypis, J. Konstan, and J. Reidl, "Application of dimensionality reduction in recommender systems—A case study," in *Proc. ACM WebKDD,* Boston, MA, USA, 2000, pp. 285–295.

[10] T. Hofmann, "Latent semantic models for collaborative-filtering," *ACM Trans. Inf. Syst.*, vol. 22, pp. 89–115, 2004.

[11] N. Srebro, J. D. M. Rennie, and T. S. Jaakola, "Maximum-margin matrix-factorization," *Adv. Neural Inf. Process. Syst.*, vol. 17, pp. 1329–1336, 2005.

[12] M. Kurucz, A. Benczúr, and K. Csalogány, "Methods for large scale SVD with missing values," in *Proc. 13th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining,* San Jose, CA, USA, 2007, pp. 7–14.

[13] G. Gorrell, "Generalized Hebbian algorithm for incremental singular value decomposition in natural language processing," in *Proc. 11th Conf. Eur. Chapter Assoc. Comput. Linguist.,* Stroudsburg, PA, USA, 2006, pp. 97–104.

[14] A. Paterek, "Improving regularized singular value decomposition for collaborative-filtering," in *Proc. 13th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining,* San Jose, CA, USA, 2007, pp. 39–42.

[15] G. Takács, I. Pilászy, B. Németh, and D. Tikky, "Scalable collaborative-filtering approaches for large recommender systems," *J. Mach. Learn. Res.*, vol. 10, pp. 623–656, 2009.

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

[17] R. Salakhutdinov and A. Mnih, "Probabilistic matrix-factorization," *Adv. Neural Inf. Process. Syst.*, vol. 20, pp. 1257–1264, 2008.

[18] K. Yu, S. Zhu, J. Lafferty, and Y. Gong, "Fast nonparametric matrix-factorization for large-scale collaborative-filtering," in *Proc. 32nd ACM SIGIR Int. Conf. Res. Dev. Inf. Retrieval,* Boston, MA, USA, 2009, pp. 211–218.

[19] H. Ma, H. Yang, M. R. Lyu, and I. King, "SoRec: Social recommendation using probabilistic matrix-factorization," in *Proc. 17th ACM Int. Conf. Inf. Knowl. Manage.,* Napa Valley, CA, USA, 2008, pp. 931–940.

[20] Z. Ning, W. K. Cheung, Q. Guoping, and X. Xiangyang, "A hybrid probabilistic model for unified collaborative and content-based image tagging," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 7, pp. 1281–1294, Jul. 2011.

[21] J. Wu, L. Chen, Y.-P. Feng, Z.-B. Zheng, M.-C. Zhou, and Z. Wu, "Predicting quality of service for selection by neighborhood-based collaborative-filtering," *IEEE Trans. Syst. Man Cybern., Syst.*, vol. 43, no. 2, pp. 428–439, Mar. 2013.

[22] Z.-B. Zheng, H. Ma, M. R. Lyu, and I. King, "Collaborative web service qos prediction via neighborhood integrated matrix factorization," *IEEE Trans. Serv. Comput.*, vol. 6, no. 3, pp. 289–299, Jul./Sep. 2013.

[23] M.-F. Weng and Y.-Y. Chuang, "Collaborative video reindexing via matrix-factorization," *ACM Trans. Multimedia Comput. Commun. Appl.*, vol. 8, pp. 1–20, 2012.

[24] J. J. Pan, S. J. Pan, Y. Jie, L. M. Ni, and Y. Qiang, "Tracking mobile users in wireless networks via semi-supervised colocalization," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 3, pp. 587–600, Mar. 2012.

[25] D. D. Lee and S. H. Seung, "Learning the parts of objects by non-negative matrix-factorization," *Nature*, vol. 401, pp. 788–791, 1999.

[26] D. D. Lee and S. H. Seung, "Algorithms for non-negative matrix-factorization," *Adv. Neural Inf. Process. Syst.*, vol. 13, pp. 556–562, 2000.

[27] S. Zhang, W. Wang, J. Ford, and F. Makedon, "Learning from incomplete ratings using non-negative matrix-factorization," in *Proc. SIAM Int. Conf. Data Mining,* Bethesda, MD, USA, 2006, pp. 549–553.

[28] M. Wu, "Collaborative-filtering via ensembles of matrix-factorizations," in *Proc. KDDCup Workshop,* San Jose, CA, USA, 2007, pp. 43–47.

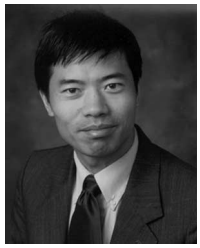
[29] G. Chen, F. Wang, and C. Zhang, "Collaborative-filtering using orthogonal nonnegative matrix tri-factorization," *Inf. Process. Manage.*, vol. 45, pp. 368–379, 2009.

[30] Q. Gu, J. Zhou, and C. Ding, "Collaborative-filtering: Weighted nonnegative matrix-factorization incorporating user and item graphs," in *Proc. SIAM Int. Conf. Data Mining,* Columbus, OH, USA, 2010, pp. 199–210.

[31] M. W. Berry, M. Browne, A. N. Langville, V. P. Pauca, and R. J. Plemmons, "Algorithms and applications for approximate nonnegative matrix-factorization," *Comput. Statist. Data Anal.*, vol. 52, pp. 155–173, 2007.
[32] C. Ding, L. Tao, and M. I. Jordan, "Convex and semi-nonnegative matrix-factorizations," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 1, pp. 45–55, Jan. 2010.
[33] F. Gruson, P. L. Moigne, P. Delarue, A. Videt, X. Cimetiére, and M. Arpillière, "A simple carrier-based modulation for the SVM of the matrix converter," *IEEE Trans. Ind. Informat.*, vol. 9, no. 2, pp. 947–956, May 2013.
[34] T. D. Nguyen and H.-H. Lee, "A new SVM method for an indirect matrix converter with common-mode voltage reduction," *IEEE Trans. Ind. Informat.*, vol. 10, pp. 61–72, 2014.
[35] J. Herlocker, J. Konstan, L. Terveen, and J. Riedl, "Evaluating collaborative-filtering recommender systems," *ACM Trans. Inf. Syst.*, vol. 22, pp. 5–53, 2004.
[36] G. Shani and A. Gunawardana, "Evaluating recommendation systems," in *Recommender Systems Handbook*, F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, Eds. New York, NY, USA: Springer, 2011, pp. 257–297.
[37] J. Yoo and S. Choi, "Weighted nonnegative matrix co-tri-factorization for collaborative prediction," in *Advances in Machine Learning*, Z.-H. Zhou and T. Washio, Eds., Berlin, Germany: Springer-Verlag, 2009, pp. 396–411.
[38] N. Zheng, Q. Li, S. Liao, and L. Zhang, "Which photo groups should I choose? A comparative study of recommendation algorithms in Flickr," *J. Inf. Sci.*, vol. 36, pp. 733–750, 2010.
[39] H. Ma, I. King, and M. R. Lyu, "Learning to recommend with explicit and implicit social relations," *ACM Trans. Intell. Syst. Technol.*, vol. 2, pp. 1–19, 2011.
[40] M. H. Aghdam, M. Analoui, and P. Kabiri, "Application of nonnegative matrix-factorization in recommender systems," in *Proc. 16th Int. Symp. Telecommun.,* 2012, pp. 873–876.
[41] K. Choi and Y. Suh, "A new similarity function for selecting neighbors for each target item in collaborative-filtering," *Knowl.-Based Syst.*, vol. 37, pp. 146–153, 2013.
[42] K. Goldberg, T. Roeder, D. Gupta, and C. Perkins, "Eigentaste: A constant time collaborative-filtering algorithm," *Inf. Retr.*, vol. 4, pp. 133–151, 2001.
[43] [Online]. Available: http://movielens.umn.edu/, accessed on Jan. 09, 2014.

Xin Luo received the B.S. degree in computer science from the University of Electronic Science and Technology of China, Chengdu, China, in 2005, and the Ph.D. degree in computer science from Beihang University, Beijing, China, in 2011.

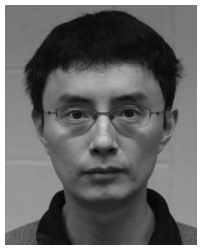
He joined the College of Computer Science, Chongqing University, Chongqing, China, in 2011, as an Assistant Professor, and is now an Associate Professor of Computer Science and Engineering. His research interests include recommender systems, social-network analysis, service computing, and bioinformatics.



Mengchu Zhou (S'88–M'90–SM'93–F'03) received the B.S. degree in control engineering from the Nanjing University of Science and Technology, Nanjing, China, in 1983; the M.S. degree in automatic control from the Beijing Institute of Technology, Beijing, China, in 1986; and the Ph.D. degree in computer and systems engineering from Rensselaer Polytechnic Institute, Troy, NY, USA, in 1990.

He joined the New Jersey Institute of Technology, Newark, NJ, USA, in 1990, as an Assistant Professor, and is now a Distinguished Professor of Electrical and Computer Engineering. His research interests include Petri nets, sensor networks, web services, transportation, and energy systems. He has published more than 250 journal papers (the majority in IEEE Transactions).

Dr. Zhou is the Founding Editor of the IEEE Press Book Series on Systems Science and Engineering. He is a Life Member of the Chinese Association for Science and Technology—USA and served as its President in 1999, and a Fellow of the International Federation of Automatic Control (IFAC) and the American Association for the Advancement of Science (AAAS).



Yunni Xia (M'12) received the B.S. degree in computer science from Chongqing University, Chongqing, China, in 2003, and the Ph.D. degree in computer science from Peking University, Beijing, China, in 2008.

He joined the College of Computer Science, Chongqing University, in 2008, as an Assistant Professor, and is now an Associate Professor of Computer Science and Engineering. His research interests include Petri nets, service computing, cloud computing, and web services.

Qingsheng Zhu received the B.S., M.S., and Ph.D. degrees in computer science from Chongqing University, Chongqing, China, in 1982, 1985, and 2000, respectively.

He was a Visiting Scholar with Oxford University, Oxford, U.K., and the University of London, London, U.K., from 1993 to 1994, and was a Visiting Professor with the University of Illinois, Chicago, IL, USA, from 2001 to 2002. Currently, he is a Professor of Computer Science and Engineering, Chongqing University, and the Director of the Key Laboratory of Software Theory and Technology, Chongqing, China. His research interests include service-oriented software engineering, data mining, and outlier detection.