Deep Matrix Factorization for Trust-Aware Recommendation in Social Networks

Liangtian Wan, *Member, IEEE*, Feng Xia, *Senior Member, IEEE*, Xiangjie Kong, *Senior Member, IEEE*, Ching-Hsien Hsu, *Senior Member, IEEE*, Runhe Huang, and Jianhua Ma

***Abstract*—Recent years have witnessed remarkable information overload in online social networks, and social network based approaches for recommender systems have been widely studied. The trust information in social networks among users is an important factor for improving recommendation performance. Many successful recommendation tasks are treated as the matrix factorization problems. However, the prediction performance of matrix factorization based methods largely depends on the matrixes initialization of users and items. To address this challenge, we develop a novel trust-aware approach based on deep learning to alleviate the initialization dependence. First, we propose two deep matrix factorization (DMF) techniques, i.e., linear DMF and non-linear DMF to extract features from the user-item rating matrix for improving the initialization accuracy. The trust relationship is integrated into the DMF model according to the preference similarity and the derivations of users on items. Second, we exploit deep marginalized Denoising Autoencoder (Deep-MDAE) to extract the latent representation in the hidden layer from the trust relationship matrix to approximate the user factor matrix factorized from the user-item rating matrix. The community regularization is integrated in the joint optimization function to take neighbours' effects into consideration. The results of DMF are applied to initialize the updating variables of Deep-MDAE in order to further improve the recommendation performance. Finally, we validate that the proposed approach outperforms state-of-the-art baselines for recommendation, especially for the cold-start users.**

***Index Terms*—Autoencoder, deep learning, matrix factorization, social networks, trust relationship.**

Manuscript received January 5, 2019; revised December 21, 2019 and August 16, 2020; accepted December 2, 2020. Date of publication December 11, 2020; date of current version March 17, 2021. This work is partially supported by National Natural Science Foundation of China (61 801 076, 61 872 054, 62 072 409), Fundamental Research Funds for the Central Universities (DUT20JC29), Zhejiang Provincial Natural Science Foundation (LR21F020003), and Fundamental Research Funds for the Provincial Universities of Zhejiang (RF-B2020001). Recommended for acceptance by Dr. Jie Li. *(Corresponding author: Feng Xia.)*

Liangtian Wan is with the Key Laboratory for Ubiquitous Network and Service Software of Liaoning Province, School of Software, Dalian University of Technology, Dalian 116 620, China (e-mail: wan.liangtian.2015@ieee.org).

Feng Xia is with the School of Engineering, IT and Physical Sciences, Federation University Australia, Ballarat, VIC 3353, Australia (e-mail: f.xia@ieee.org).

Xiangjie Kong is with the College of Computer Science and Technology, Zhejiang University of Technology, Hangzhou 310 023, China (e-mail: xjkong@ieee.org).

Ching-Hsien Hsu is with the College of Information and Electrical Engineering, Asia University, Taichung 41354, Taiwan, and also with Department of Medical Research, China Medical University Hospital, China Medical University, Taichung 404332, Taiwan (e-mail: robertchh@gmail.com).

Runhe Huang and Jianhua Ma are with the Faculty of Computer and Information Sciences, Hosei University, Tokyo 184-8584, Japan (e-mail: rhuang@hosei.ac.jp; jianhua@hosei.ac.jp).

Digital Object Identifier 10.1109/TNSE.2020.3044035

I. Introduction

DUE to the rapidly growing amount of information and explosive appearance of new services available in the web, the overloaded information prevents users from obtaining useful information conveniently [1]–[6]. How to help the overwhelmed users to select the interested part of online information is becoming an unprecedentedly important task. Both academic and industrial fields pay much attention to this problem. To satisfy this requirement, recommender systems have emerged as an effective mechanism to provide suitable recommendation for the costumers about what kinds of the items or persons that they may be potentially interested [7]. At present, there are many popular recommender systems such as the items recommendation in Amazon, the music recommendation in Last.fm, the movies recommendation in Netflix and the friends recommendation in Linked [8], [9], etc.

A. Motivation

Although recommender systems have been widely studied and used both in academic and industrial fields, some important problems still exist. First, only a small proportional items have been rated. The users' ratings on items may reflect the interest of users, and the users' historical rating data can generally be formalized as the user-item rating matrix, which usually influences the collaborative filtering-based recommender systems. The host information systems of recommender systems can provide a large amount of information, and thus the dimension of user-item rating matrix is generally very high. However, the users only visit a relative small number of items, and the number of ratings that the users assign to the items is rare. Thus a large number of ratings are lacked in the user-item rating matrix. As illustrated in [10], the available ratings for implementing recommendation are very rare. The classical user-based collaborative filtering approaches exploit the neighbors' ratings of the target user to predict his/her rating. When the user-item rating matrix is egregious sparsity, the target user's neighbors can hardly be identified. Thus the recommendation coverage would deteriorate dramatically. The classical item-based collaborative filtering approaches exploit the target users' ratings which already exist in the user-item rating matrix to achieve the recommendation. Due to the existing of sparsity, the users' historical rating data is too rare, which means that the essence of sparsity is information missing in the user-item rating matrix. The neighbors' items of

target items that have already been visited cannot be identified. The recommendation accuracy would deteriorate dramatically, and the item-based approaches even fail to recommend the items to target users. The recommendation quality of collaborative filtering approaches cannot be guaranteed without sufficient data of ratings assigned to items by users.

Second, the challenge of cold-start exists in recommender systems, including the cold-start users and the cold-start items. When new users appear in the systems, no items has been assigned by them. There are no ratings for the new users can be utilized. From this perspective, the content-based collaborative filtering approaches do not work since the user profile cannot be constructed in recommender systems. For the user-based collaborative filtering approaches, the similar neighbors of target user cannot be identified in the recommender systems. In terms of the item-based collaborative filtering approaches, the item similarity can be determined by recommender systems. The ratings of items have not been assigned by new users, and thus the prediction cannot be completed. When new items appear in the systems, for the content-based collaborative filtering approaches, the model of new items can be constructed in recommender systems without ratings assigned by users. For collaborative filtering approaches, the similarity calculation and the predictions of ratings cannot be completed when the data deficiency of items' ratings exists. The cold-start problem exists during the whole life cycle of recommender systems, especially in the initial stage of construction of recommender systems. The new users and items are difficult for investigation without any prior information. The defect of cold-start is the excessive dependence of the rating data. Thus other information about users or items should be considered to design or improve the recommender systems.

Third, the trust relationship [11], [12] among the users has not been considered in traditional recommender systems. In real application, compared with other ordinary users, their friends' comments can better affect the users' decisions about which items that they are interested. Therefore, the traditional recommender systems cannot sufficiently provide accurate and reliable predictions since they only consider the user-item rating matrix. Thus new approaches and techniques are needed to address these problems.

In this paper, the trust relationship is integrated to solve the problem mentioned above. With the help of the trust relationship, the effect of cold-start problem can be relieved as well as the data sparsity. Although the neighbours of the cold-start users cannot be known accurately, the user's preference can be inferred via the friends-based social networks.

B. Contributions

In this paper, we propose several approaches based on matrix factorization for recommendation exploiting trust-aware relationship in social networks. Since the initialization is very important for matrix factorization-based approaches, we propose an initialization method based on deep learning, where Deep Matrix Factorization (DMF) is used for pertaining the initial feature matrices for our learning model. We propose a DMF-based model by integrating and exploiting the trust relationship for overcoming the data sparsity. Finally, a series of experiments verify the superiority of our proposed methods by extracting user data from online social networks Epinions and Flixster. From the experiment results, our proposed methods have higher recommendation accuracy compared with other baseline methods. In conclusion, the main contributions of this paper are listed as follows:

1) The deep learning techniques are integrated with the social trust relationship to improve the recommendation performance.

2) The Linear Representation DMF (LRDMF) and Non-Linear Representation DMF (NLRDMF) are adopted to improve the initialized accuracy of matrix factorization. The user preferences and the derivations of users on items are taken into consideration as social trust relationship, and they have been integrated in the DMF for overcoming data sparsity issue.

3) We propose a joint optimization function to enforce the user factor matrix as close as possible to the latent representation of the trust relationship in the hidden layers of the deep Marginalized Denoising Autoencoder (Deep-MDAE). In addition, we integrate the community regularization in the joint optimization function to take the neighbours' effects into consideration. The two factorized matrices obtained from DMF are utilized for initializing the updating variables of Deep-MDAE.

4) The data sets are extracted from Epinions and Flixster. From the recommendation results, our DMF trust based methods obtain higher recommendation accuracy compared with other methods, especially in the case of sparse data and cold-start users.

C. Organization of the Paper

This paper is organized as follows. The related work is discussed in Sectin 2. The problem formulation is given in Section 3. The proposed DMF algorithm is analyzed in Section 4. The proposed social trust based DMF method is presented in Section 5. The experiments are shown and analyzed in Section 6. The conclusions are drawn in Section 7.

D. Notations

In this paper, the operator $(\cdot)^{\dagger}$ and $(\cdot)^{T}$ stand for the pseudo-inverse and the transpose of a matrix, respectively. $\boldsymbol{I}_M$ stands for an $M \times M$ identity matrix. $\|\cdot\|_F$ denotes the Frobenius norm. $\odot$ stands for the Hadamard products. E stands for the expectation operator, and tr stands for the trace operator.

II. Related Work

Collaborative filtering has been widely used in recommender systems [13]. However, the sparse data and the cold-start problem exist in the collaborative filtering-based approaches [8]. Many frameworks have been proposed by researchers for solving the problems in the collaborative filtering-based approaches. The interpolation technique has been applied to fill the missing entries in the user-item rating matrix [14] for overcoming the problem of sparse data. The memory-based collaborative

filtering approaches are also known as the neighbour-based collaborative filtering approaches. The user-item rating matrix is utilized to generate recommendation items. First, various similarity metrics are adopted to calculate the similarity among different users or items. Then the neighbours of active users or target items can be found. The sum of weight of the neighbours belonging to active users or target items can be regarded as the prediction ratings. The typical memory-based collaborative filtering approaches are divided into user-based and item-based collaborative filtering approaches. However, the memory-based collaborative filtering approaches do not scale well to commercial recommender systems. In addition, the model-based collaborative filtering approaches provide a scalable solution for the scenario with relatively sufficient data.

Model-based collaborative filtering approaches utilize the machine learning skills to train prediction model. Users' behaviours are depicted by prediction model, and then model-based collaborative filtering approaches can predict the items' ratings assigned by users via the learned prediction model. However, the entries in user-item rating matrix are not all used. Typical model-based collaborative filtering approaches include Bayesian network-based approaches, clustering model-based approaches, potential semantic model-based approaches, limited Boltzmann machine-based approaches and association rules-based approaches.

Matrix factorization is also a widely used method based on collaborative filtering [17]. In practice, we cannot directly factorize matrices in most of the cases because of its low recommendation accuracy. Deep learning [29] has been applied to improve the performance of matrix factorization, as well as to find an appropriate way to represent matrices in low dimension and relieve the data sparsity and cold-start problems partly. Trust information has been derived from [8] recently, and then the trust-aware recommendation becomes a developing area to enhance the recommendation performance of learning based methods.

A. Matrix Factorization Based Methods

Matrix factorization-based recommendation approaches [18] have been widely adopted by researchers. The main reason is that the matrix factorization technique can effectively deal with the user-item rating matrix with large scale. The matrix factorization technique assumes that users' behaviors are influenced by only a few implicit factors. Matrix factorization simultaneously maps the feature vectors of users and items into low dimension hidden feature space, in which the inner product between users and items can be directly calculated.

Constrained non-negative matrix factorisation (CNMF) incorporates the additional constraints as regularization of the error function on the prime problem [19]. However, there exists an apparent problem for CNMF, i.e., it concerns the problem in the global scope. However, for a local or pairwise situation, there is a lack of consideration. For solving this problem, a method named graph regularized nonnegative matrix factorization (GNMF) has been proposed in [20]. The geometrical information is represented by constructing a nearest neighbor graph, and the graph structure is incorporated into a new factorization objective function.

Recently, Relative Pairwise Relationships Non-negative Matrix Factorisation (RPR-NMF) has been proposed in [21]. The penalties imposed on relative pairwise relationship can be written as triplets. By adjusting the conditions of factors, RPR-NMF is able to implement on more recommendation issues conveniently.

Besides, there are various forms of matrix factorisation methods as well, such as nonnegative matrix factorization (NMF), SVD++, Bayesian probabilistic matrix factorization (BPMF), probabilistic matrix factorization (PMF) and maximum-margin matrix factorization (MMMF) [22]. Each of implicit eigenvectors of NMF constraint must be positive. The PMF uses probabilistic graph models with Gaussian noise to represent implicit eigenvectors of users and items. BPMF assumes that users' and items' hyperparameters are a priori, and they obey the Gaussian-Wishart distribution. The Markov chain Monte Carlo method has been performed for approximate reasoning. SVD++ generates recommendations based on both the explicit and the implicit effect of ratings.

B. Deep Learning Based Methods

Matrix factorization is an ideal way to integrate trust-aware recommendations. In fact, factorizing user-item rating matrix directly in a reasonable way is almost impossible. Because of the complex connections among users and items, we need a more effective approach to capture these connections. Based on deep learning, some architectural paradigms, including multilayer perceptron (MLP), autoencoder, recurrent neural network (RNN), convolutional neural network (CNN), restricted Boltzmann machine (RBM), neural autoregressive distribution estimation (NADE), adversarial networks, attentional models and deep reinforcement learning (DRL) have been proposed in [23].

Compared with traditional algorithms such as matrix factorization, the merits deep learning based methods consist of three aspects. 1) It can deal with nonlinear mapping efficiently, which can capture complex interactions among users and items. 2) It is useful for learning the underlying factors, that is convenient for us to extract key information from massive data. 3) It has improvement on sequence modeling.

Among all deep learning based recommendation methods, MLP is a simple but powerful idea to achieve desirable accuracy by approximating objective function. The functions of Neural network matrix factorization (NNMF) representing the sum of vectors are selected to learn, and the function is set as a feed-forward neural network [24]. Neural collaborative filtering (NCF) [25] has became a useful tool in recommendation systems recently, and it generalizes traditional matrix factorization to NCF. Researchers trained this network by adopting weighted square loss or binary cross-entropy loss functions.

Autoencoder is also a common technique in deep learning. Among the various variants of autoencoder, denoising autoencoder is the most studied one. Many researchers consider collaborative filtering from autoencoder aspect. User or item factor matrices are set as input [26], and they are desired to

recover in the output layer. The algorithm proposed in [27] has extended AutoRec proposed in [26] by denoising and has used side information to strengthen the robustness and refine the two difficulties we mentioned before. The autoencoding variational matrix factorization and graph convolutional matrix factorization autoencoder approaches have been proposed in [28]. A hierarchical Bayesian model has been proposed in [29], which runs the deep representation learning for the content information and collaborative filtering for the user-item rating matrix jointly. The recommendation performance has been improved significantly when deep learning is embedded in the recommendation systems [30].

Besides, CNN [31] and RNN [32] also achieve excellent performance in recommendation systems such as sequential recommendation. The DMF has been used for clustering by learning hidden representations in [33]. In addition, some works such as [34], [35] has tended to use both explicit and implicit information, or complete/missing data model to create a joint model for prediction with a new loss function. They obtain excellent recommendation results based on DMF. However, the trust relationship is not considered in the approaches mentioned above.

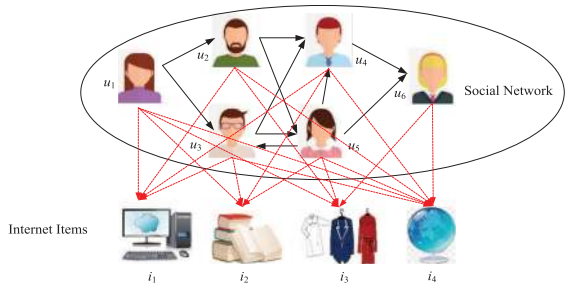
C. Trust-Aware Methods

Although the matrix factorization methods mentioned above can effectively deal with the user-item rating matrix with large scale, it cannot effectively solve the cold-start problem because of the intrinsic sparsity of the user-item rating matrix [36]. In recent years, some matrix factorization-based recommendation algorithms solve the cold-start problem by integrating additional sources of information. For example, users' tag information is integrated in a matrix factorization framework to improve the recommendation performance [37]. Based on the common interests among friends, users tend to accept recommendations from their friends. Many researchers have improved the quality of recommendation performance by integrating the information of social networks in a matrix factorization framework. The typical recommendation approaches based on social networks includes: social trust ensemble (STE) [38], SocialMF [39], TidalTrust [41] and MoleTrust [42]. SDAE proposed in [43] gives a joint objective function enforcing latent representations of social relationships and users to be as close as possible in the hidden layer of marginalized DAE. A deep learning based matrix factorization scheme for trust-aware recommendation has been proposed in [44]. Deep learning has been applied to initialize input, and a social trust ensemble learning model involving both influence of friends and communities has been adopted. Due to the efficiency of autoencoder, DAE is selected as the main method to solve the two problems stated above with trust information [45]. Users are described by not only the rating information but also the explicit trust information, and this method is named as TDAE. They also extract the implicit trust information to boost the performance, and the improved algorithm is called TDAE++. Besides, the trust relationships are also added to the input and output layer of autoencoder to map the nonlinear relationships. Now, how to distinguish a neighbor is trust-worthy? A part of researchers also propose models for neighbors selection, such as the availability evaluation module and the trust evaluation module in [46].

D. Similarity Metrics

Various similarity metrics have been applied in different scenarios, e.g., the user similarity metrics for friends recommendation in online social networks (OSNs) [47], [48], and the topological similarity metrics for link prediction and community detection [49]. The local similarity metrics utilize the local topological information to measure the similarity between nodes in networks, such as common neighbors (CN), Adamic-Adar (AA), resource allocation (RA) and preferential attachment (PA) [50]. The local path (LP) [51] index has been designed based on the path information. The widely used metric based on the structural similarity in networks is local random walk [52]. Random walk can to quantify relevance between nodes, and it is usually implemented for link prediction and recommendation tasks. In addition, a series of similarity indices including Jaccard similarity, cosine similarity, and Pearson correlation coefficient are used for measuring the interest similarity between users in OSNs [53]. Recently, the technology of network embedding has attracted lots of attention, such as DeepWalk and Node2vec [54]. They learn the low-dimensional vector representation of each node in networks [55], and then compute the similarity between vectors via different similarity metrics for the recommendation and prediction tasks.

In recent years, deep learning techniques can extract latent features and representations from the user-item rating matrix, which has been proved as an efficient method to improve the recommendation accuracy. Trust relationship matrix has been used for collecting the persons that the user trusts when this mechanism has been integrated in the recommender systems. The trust relationship matrix can be used for deducing the user preferences from his/her trust persons. This mechanism is extremely effective for the cold-start users, who have little information to deduce their preferences. Thus how to integrate the deep learning technique with the trust relationship becomes an important problem to solve. The matrix factorization technique is one of the important techniques for recommending the items to the target users in recommender systems. DMF has been used for improving item recommendation accuracy for target users to enhance the latent representations in hidden layers [34]. However, the trust relationship has not been used to further improve the recommendation performance. The autoencoder has been used for item recommendation [44], [56], which exploits the user-item rating matrix and the trust relationship matrix, respectively. However, none of them exploits both the deep structure for learning the model parameters of user-item rating matrix and the trust relationship matrix jointly. In this paper, we amalgamate the DMF of the user-item rating matrix into the autoencoder of the trust relationship matrix to achieve better recommendation performance in recommender systems.



(a)

	i_1	i_2	i_3	i_4
u_1	5	4	5	5
u_2	3		4	5
u_3	5	4		4
u_4	3	5		3
u_5		4	3	5
u_6			4	4

(b)

Fig. 1. Example of trust-aware recommendation. (a) Item-rating in a social network. (b) User-item rating matrix.

III. Problem Formulation

In recommender systems, we assume that $U = \{u_1, u_2, \ldots, u_M\}$ represents user set and $I = \{i_1, i_2, \ldots, i_N\}$ represents the item set. The ratings assigned by users on items are represented as a user-item rating matrix $\boldsymbol{R} \in \mathbb{R}^{M \times N}$. $\boldsymbol{R}_{m,n}$ is the rating of item n assigned by user m. Ratings are often represented as integers between 1 and 5. We normalize the ratings for mapping the ratings into the interval of $[0, 1]$. In a rating network of users and items, each user m has a set of neighbors N_m, and $t_{m,v}$ represents the social trust value that the user m assigns on user v in the range $[0, 1]$. If the value is zero, it means no trust relationship exists. Otherwise, it denotes full trust. In terms of the binary trust networks, the trust values among users are represented as a trust relationship matrix $\boldsymbol{T} \in \mathbb{R}^{M \times M}$. $\boldsymbol{T}_{m,v}$ in $\boldsymbol{T}$ represents the social relationship from user m to user v, and $\boldsymbol{T}$ is asymmetric.

The rating network consists of nodes and edges. Users and items are represented as nodes in a network. The edges between users represent their trust relationships, and the edge weights between users and items denote the ratings on the items assigned by the users. An example of social rating network is shown in Fig. 1(a), and the corresponding user-item rating matrix is shown in Fig. 1(b). It can be seen from the rating matrix that only part of the user-item rating matrix can be used for recommendation, and the other ratings are not known.

Therefore, the trust-aware recommendation task is described as follows: given a user m and an item n, we aim to predict the rating on item n from user m by using the user-item rating matrix $\boldsymbol{R}$ and the trust relationship matrix $\boldsymbol{T}$.

Next, we will introduce the basic matrix factorization approach from a probability perspective. It should be noted that the basic matrix factorization approach only use the known part of the user-item rating matrix $\boldsymbol{R}$ to predict the unknown part of the user-item rating matrix $\boldsymbol{R}$.

Matrix factorization is an efficient model for predicting missing values in a given matrix. This problem is also known as matrix completion [15], which has attracted increasing attentions from researchers in the field of recommender systems [16]. Matrix factorization model represents both users and items by using a low-dimensional latent feature space. i.e., the user-item rating matrix is modeled as a product of two user and item matrices with low rank. The scenario that the matrix factorization technique adopted is that the user-item interactions are influenced by a few key features, and the application of each feature will influence a user's interactive experience [17]. The trust-aware relationship is not used for predicting missing values in the user-item rating matrix.

The probabilistic linear model with Gaussian observation is adopted. The conditional distribution over the observed rating is defined as

$$p\left(R|P, Q, \sigma^2\right) = \prod_{m=1}^{M} \prod_{n=1}^{N} \left[\mathcal{N}\left(R_{m,n}|P_m^T Q_n, \sigma^2\right)\right]^{I_{mn}}, \quad (1)$$

where $\mathcal{N}(x|\mu, \sigma^2)$ is the probability density function of the Gaussian distribution, and it is determined by the mean μ and variance σ^2. I_{mn} is the indicator function, and it is equal to 1 if user m has rated item n, otherwise, it is 0. We assume that user and item feature vectors have zero-mean spherical Gaussian priors, and then the objective of matrix factorization is to maximize the posterior distribution over the user and item features, i.e., training and learning above latent variables by minimizing the following equation as follows

$$\begin{aligned} C = & \frac{1}{2} \min_{P,Q} \sum_{m=1}^{M} \sum_{n=1}^{N} I_{mn} \left(R_{m,n} - P_m^T Q_n\right)^2 \\ & + \frac{\lambda_P}{2} \|P\|_F^2 + \frac{\lambda_Q}{2} \|Q\|_F^2, \end{aligned} \quad (2)$$

where λ_P and λ_Q are both regularization terms for avoiding the overfitting of our model, and $\|\cdot\|_F^2$ is the Frobenius norm. We initialize $\boldsymbol{P}$ and $\boldsymbol{Q}$ randomly. Then we can perform the stochastic gradient descent technique [18] in $\boldsymbol{P}$ and $\boldsymbol{Q}$ to minimize the objective function given by (2). The update formulation is given as follows

$$\begin{aligned} P_m^{(t+1)} &= P_m^{(t)} - \gamma_1 \frac{\partial C}{\partial P_m^{(t)}}, \\ Q_n^{(t+1)} &= Q_n^{(t)} - \gamma_2 \frac{\partial C}{\partial Q_n^{(t)}}, \end{aligned} \quad (3)$$

where $\gamma_1 > 0$ and $\gamma_2 > 0$ are learning rates. A probabilistic foundation for regularizing the learned variables is given in [57], and some recent recommended approaches have adopted this form for the item recommendation in social networks [38]–[40].

IV. Deep Matrix Factorization for Recommendation Systems

As shown in Fig. 1(b), the matrix that we need to factorize is the user-item rating matrix, whose entries are assigned by the users (the corresponding column) to the items (the corresponding row). The two factorized matrices are corresponding to the users and items, which are called latent user and item

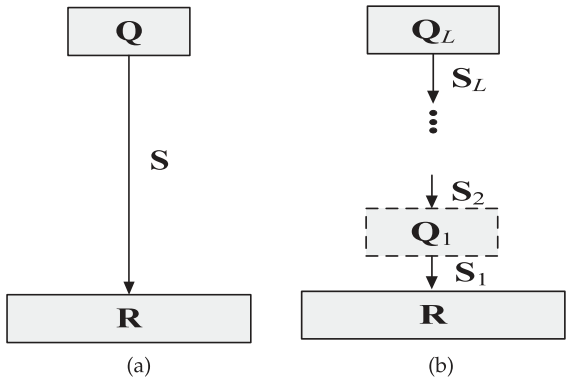


Fig. 2. Difference between traditional and the proposed Semi-NMF model. (a) Semi-NMF model. (b) Deep Semi-NMF model.

feature matrices, respectively. Learning models by matrix factorization is a mature approach for solving recommendation problem when only part of the user-item rating matrix can be used. In general, $\boldsymbol{P} \in \mathbb{R}^{K\times M}$ and $\boldsymbol{Q} \in \mathbb{R}^{K\times N}$ are latent user and item feature matrices, and the column vectors $\boldsymbol{P}_m \in \mathbb{R}^K$ and $\boldsymbol{Q}_n \in \mathbb{R}^K$ represent use-specific and item-specific latent feature vectors, respectively. Hence, we divide the user-item rating matrix into two sub-matrices $\boldsymbol{P}$ and $\boldsymbol{Q}$ with the constraints of K-dimensional features:

$$R \approx P^T Q. \tag{4}$$

The definition of NMF satisfies that factors $S = P^T$ (We define $S = P^T$ for the simplicity of the derivation in the following paragraph.) and Q are non-negative. The approach proposed in [58] has extended the applicability of NMF, which is called Semi-NMF. It is a NMF variant which both positive and negative signs can exist in the user-item rating matrix R and the first factor S. However, only positive signs can exist in the second factor Q.

A. Linear Representations

In order to solve the recommendation problem, a Semi-NMF exploiting deep learning framework named as Deep Semi-NMF is proposed inspired from [57]. Based on the unsupervised learning pattern, the matrix can be factorized into multiple factors. The differences between traditional Semi-NMF and the proposed Deep Semi-NMF frameworks are shown in Fig. 2. As shown in Fig. 2(a), there is a linear transformation between the new representation Q and the original user-item rating matrix R deriving from Semi-NMF. As demonstrated in Fig. 2(b), multiple hidden representations of the identical hierarchy are learned by Deep Semi-NMF, which uncovers the final low-dimensional representation of the original user-item rating matrix. The cost function can be written as

$$\begin{aligned} C_{\text{deep}} =& \frac{1}{2}\|R - S_1 S_2 \cdots S_L Q_L\|_F^2 \\ &+ \frac{1}{2}\|S_1 S_2 \cdots S_L\|_F^2 + \frac{1}{2}\|Q_L\|_F^2. \end{aligned} \tag{5}$$

In general, training a deep neural network should cost lots of time. The factor matrices S_l and Q_l in the proposed Deep Semi-NMF framework need to be approximated in an accelerated way, and thus we pre-train each layer of this neural network. Then the initial approximation of factor matrices S_l and Q_l can be achieved. As reported in [59], the deep autoencoder networks have been pre-trained to reduce the training time greatly. For example, the original user-item rating matrix can be decomposed into $R = S_1 Q_1$, where $S_1 \in \mathbb{R}^{M\times k_1}$ and $Q_1 \in \mathbb{R}^{k_1\times N}$. In the similar way, the feature matrix can be decomposed into $Q_1 = S_2 Q_2$, where $S_2 \in \mathbb{R}^{k_1\times k_2}$ and $Q_2 \in \mathbb{R}^{k_2\times N}$. This procedure can be done until the $(l-1)$th feature matrix is decomposed into $Q_{L-1} = S_L Q_L$, where $S_L \in \mathbb{R}^{k_{L-1}\times k_L}$ and $Q_L \in \mathbb{R}^{k_L\times N}$. Thus we pre-train all the layers. Afterwards, the weight of each layer, i.e., S_l and Q_l, can be fine-tuned by exploiting alternating minimization, and then the reconstruction error of the model can be reduced dramatically.

1) Updating Step for Weight Matrix S: We fix the remaining weights for the lth layer, and S_l makes the cost function (5) achieve the minimum, i.e., the partial derivation on S_l is set to be zero

$$\frac{\partial C_{\text{deep}}}{\partial S_l} = 0. \tag{6}$$

Thus the updates for S_l can be expressed as

$$S_l = \frac{\Phi^\dagger R \tilde{Q}_l^\dagger}{I + \lambda_1 (\tilde{Q}_l^T \tilde{Q}_l)^{-1}}, \tag{7}$$

where $\Phi = S_1 S_2 \cdots S_{l-1}$, and $\tilde{Q}_l$ is the inference of the lth layer's feature matrix.

2) Updating Step for the Feature Matrix Q: Since the non-negativity of Q_l needs to be enforced, the feature matrix Q_l can be updated in a similar way as given in [58]. The feature matrix Q_l can be formulated as

$$Q_l = Q_l \odot \left[\frac{\left[\Phi^T R\right]^{\text{pos}} + \left[\Phi^T \Phi\right]^{\text{neg}} Q_l}{\left[\Phi^T R\right]^{\text{pos}} + \left[\Phi^T \Phi\right]^{\text{neg}} Q_l + \lambda_2 Q_l} \right]^{\eta}, \tag{8}$$

.

We set η to be a small number for preventing (8) to be zero. 0 takes place of matrix A with negative elements, and this matrix is defined as A^{pos}. 0 takes place of matrix A with positive elements, and this matrix is defined as A^{neg}:

$$\forall i, j, A_{ij}^{\text{pos}} = \frac{|A_{ij}| + A_{ij}}{2}, A_{ij}^{\text{neg}} = \frac{|A_{ij}| - A_{ij}}{2}. \tag{9}$$

At first, the Semi-NMF algorithm [58] has been used to approximate the factors greedily. Then the factors are fine-tuned until the convergence criterion is reached. In this paper, the maximum iteration number is fixed at 1000. If the difference between previous update and the current update is smaller than a threshold 10^{-6}, we stop the iteration. Thus we can train a Deep Semi-NMF model as described above. The pseudo code of the suggested training algorithm is summarized in Algorithm 1.

Thus the linear representation of Deep Semi-NMF (LRDMF) can be written as

$$R^{LR} = \hat{S}_L \hat{Q}_L. \tag{10}$$

Algorithm 1: Deep Semi-NMF.

Input: $R \in \mathbb{R}^{M\times N}$, the size of different layers
Output: different layers with deletedthe parameter matrices. i.e., weight matrices S_i and feature matrices Q_i
Initialize different Layers
for Different layers**do**
$S_l, Q_l \leftarrow$Semi-NMF (Q_{l-1},layer(i))
end for
repeat
for Different layers
$\tilde{Q}_l \leftarrow cases\boldsymbol{Q}_l$ if $l = LS_{l+1}\tilde{Q}_{l+1}$ otherwise $cases$
$\boldsymbol{\Phi} \leftarrow \prod_{l=1}^{l-1} S_l$
$S_l \leftarrow \frac{\boldsymbol{\Phi}^\dagger R\tilde{Q}_l^\dagger}{I+\lambda_1(\tilde{Q}_l^T\tilde{Q}_l)^{-1}}$
$Q_l \leftarrow Q_l \odot [\frac{[\boldsymbol{\Phi}^T R]^{\text{pos}}+[\boldsymbol{\Phi}^T\boldsymbol{\Phi}]^{\text{neg}}Q_l}{[\boldsymbol{\Phi}^T R]^{\text{pos}}+[\boldsymbol{\Phi}^T\boldsymbol{\Phi}]^{\text{neg}}Q_l+\lambda_2 Q_l}]^\eta$
end for
untilStopping criterion is satisfied.

The Semi-NMF model actually is the special case of Deep Semi-NMF model with a single layer. The cost function can be written as

$$C_{\text{snmf}} = \frac{1}{2}\|R - SQ\|_F^2 + \frac{1}{2}\|S\|_F^2 + \frac{1}{2}\|Q\|_F^2, \tag{11}$$

subject to $Q \geq 0$. The pseudo code of Semi-NMF model with a single layer is summarized in **Algorithm 2**.

B. Non-Linear Representations

According to the neurophysiology, the human visual system will process it automatically in a hierarchical and non-linear way when person looking at an image. Specifically, the corresponding neurons in the brain process the complex image features sequentially [60]. The authors in [61] exploit an adaptable non-linear image representation algorithm to reduce the statistical and the perceptual redundancy of representation elements for image processing. Inspired from the way of image processing, we introduce the non-linear representation into deep Semi-NMF model for recommendation.

In the previous section, the original user-item rating matrix has been decomposed in a linear way. However, the latent attributes of the non-linearity are ignored in the model. Besides, the linear representation cannot account for the non-linear relationship efficiently. As a consequence, the non-linear functions should be introduced between the layers in order to extract feature for each latent attribute.

We utilize a non-linear function $g(\cdot)$ between every implicit representation $(Q_1, \ldots, Q_{L-1})$ for approximating the non-linear manifolds on which the user-item rating matrix R lies [33]. In other words, the Deep Semi-NMF model has an enhanced explainability by using a non-linear squashing function. Therefore, we can reconstruct the original user-item rating matrix in an explicit way. This method has been proved in [62] under the scenario of multilayer feedforward network structures. If the hidden units are provided sufficiently and explicitly, any interest function can be approximated by arbitrary squashing functions with any desired accuracy. Deep

Algorithm 2: Semi-NMF with a single layer.

Input: $R \in \mathbb{R}^{M\times N}$, the number of components K
Output: weight matrix $S \in \mathbb{R}^{M\times K}$ and feature matrix $Q \in \mathbb{R}^{K\times N}$
Initialize Q
repeat
$S \leftarrow \frac{\boldsymbol{\Phi}^\dagger R\tilde{Q}^\dagger}{I+\lambda_1(\tilde{Q}^T\tilde{Q})^{-1}}$
$Q \leftarrow Q \odot [\frac{[\boldsymbol{\Phi}^T R]^{\text{pos}}+[\boldsymbol{\Phi}^T\boldsymbol{\Phi}]^{\text{neg}}Q}{[\boldsymbol{\Phi}^T R]^{\text{pos}}+[\boldsymbol{\Phi}^T\boldsymbol{\Phi}]^{\text{neg}}Q+\lambda_2 Q}]^\eta$
untilStopping criterion is reached.

Semi-NMF is just an instance of multi-layer feedforward network.

It is straightforward to introduce non-linearity in Deep Semi-NMF model, and thus the lth feature matrix Q_l can be modified as

$$Q_l \approx g(S_{l+1}Q_{l+1}). \tag{12}$$

The cost function of the model is rewritten as

$$C^* = \frac{1}{2}\|R - S_1 g(S_2 g(\cdots g(S_L Q_L)))\|_F^2. \tag{13}$$

Remark 1: It should be noted that the model (13) is more general compared with model (5). The feature vectors of users and items of model (5) are assumed to have zero-mean spherical Gaussian priors. However, the model (13) does not have this constraint.

By using the chain rule, the derivation of lth feature layer can be described as

$$\begin{aligned}\frac{\partial C^*}{\partial Q_l} &= S_l^T \frac{\partial C^*}{\partial S_l Q_l} \\ &= S_l^T\left[\frac{\partial C^*}{\partial g(S_l Q_l)} \odot \nabla g(S_l Q_l)\right] \\ &= S_l^T\left[\frac{\partial C^*}{\partial Q_{l-1}} \odot \nabla g(S_l Q_l)\right].\end{aligned} \tag{14}$$

Therefore, the derivation of the first feature layer Q_1 is concordant with the model of one layer

$$\begin{aligned}\frac{\partial C^*}{\partial Q_1} &= \frac{1}{2}\frac{\partial \text{Tr}\left[-2R^T S_1 Q_1 + (S_1 Q_1)^T S_1 Q_1\right]}{\partial Q_1} \\ &= S_1^T S_1 Q_1 - S_1^T R \\ &= S_1^T(S_1 Q_1 - R).\end{aligned} \tag{15}$$

Similarly, the derivation for the weight matrices S_l can be expressed as

$$\begin{aligned}\frac{\partial C^*}{\partial S_l} &= \frac{\partial C^*}{\partial S_l Q_l} Q_l^T \\ &= \left[\frac{\partial C^*}{\partial g(S_l Q_l)} \odot \nabla g(S_l Q_l)\right] Q_l^T \\ &= \left[\frac{\partial C^*}{\partial Q_{l-1}} \odot \nabla g(S_l Q_l)\right] Q_l^T,\end{aligned} \tag{16}$$

and

$$\frac{\partial C^*}{\partial S_1} = \frac{1}{2}\frac{\partial \mathrm{Tr}\left[-2R^T S_1\tilde{Q}_1 + (S_1\tilde{Q}_1)^T S_1\tilde{Q}_1\right]}{\partial S_1} = (S_1\tilde{Q}_1 - R)\tilde{Q}_1^T. \tag{17}$$

By using these derivations, the cost function corresponding to the weight of the model can be minimized with gradient decent optimizations by using Nesterov's optimal gradient [63]. Based on the non-linear representation of Deep Semi-NMF (NLRDMF), the original user-item rating matrix can be written as

$$R^{NLR} = \hat{S}_1\hat{Q}_1. \tag{18}$$

C. Stochastic Optimization

Unfortunately, for Semi-NMF or NMF, it is difficult to compute for large datasets since the computational complexity of these algorithms would grow quadratically in proportional to the number of items n. In addition, the whole training dataset is required to be resided in main memory. In recent years, the stochastic optimization techniques have been proposed to mitigate these two problems. In each iteration, only a small portion of the dataset is processed. Thus several iterations are required to process the whole dataset, and this method is known as mini-batch [64]. The number of mini-batches is set to be H. The cost function of the stochastic Deep Semi-NMF can be expressed as

$$\tilde{C} = \frac{1}{2}\sum_{h=0}^{H-1}\left\|R^{[h]} - S_1 g(S_2 g(\cdots g(S_L Q_L^{[h]})))\right\|_F^2, \tag{19}$$

subject to $\forall h$, where $Q_h \geq 0$, and $R^{[h]}$ is the subset of the training set. i.e., a small batch of training set. $R^{[h]}$ contains $b = \frac{n}{H}$ examples. The stochastic optimization techniques such as Adam and SGD [65] are adopted for updating all the parameters in the Deep Semi-NMF model. This is an approximation implementation over the whole training set, but the stochastic optimization techniques take effect even for a small batch sizes (32 samples).

V. Social Trust Ensemble

The social trust networks will affect users' strategic decisions when users select items. In this section, we analyze this phenomenon, and propose an extended Deep Semi-NMF model based on users' trusted friends. First, we integrate the trust degree to improve the recommendation accuracy. Then, we exploit the autoencoder to extract the latent representation in the trust relationship matrix $T \in \mathbb{R}^{M\times M}$ to further enhance the recommendation accuracy.

A. Trust Degree Ensemble

As mentioned in Section III, a trust relationship matrix $T \in \mathbb{R}^{M\times M}$ is used for representing the trust values among users. Users tend to allocate the scores to their trusted friends in most social networks, and these scores correspond to the trust degrees between user-friend pairs. However, there is no explicit trust values measuring the trust degree in existing online social networks. Consequently, it is significant to construct a model to measure the trust degree of the social networks without trust values.

Generally speaking, most users tend to trust their friends, as well as the recommendations provided by their friends. However, target users may not well satisfy with the recommendations from their trusted friends since the difference between them exists, including their potential interests, preferences or habits. In this case, we take the trust degree into consideration, it contains the similarity and the social based trust degrees. The trust degree $\mathrm{trust}(v_i, v_j)$ assigned to v_i from v_j is calculated as

$$T_{v_i,v_j} = (1-\beta)T^I_{v_i,v_j} + \beta T^O_{v_i,v_j}, \tag{20}$$

where $T^I_{v_i,v_j}$ and $T^O_{v_i,v_j}$ stand for similarity and the social based trust degrees, respectively, and β is a weight coefficient.

The social behavior in a social network is usually modeled as the trust relationship, such as the followers of a person and the message forwarding in microblog. Thus the similarity based trust degree $T^I_{v_i,v_j}$ can be modelled as

$$T^I_{v_i,v_j} = \frac{\mathrm{sim}(v_i,v_j)\times \mathrm{trust}(v_i,v_j)}{\sum_{m\in S_m}\mathrm{sim}(v_i,v_j)\times \mathrm{trust}(v_i,v_j)}, \tag{21}$$

where $\mathrm{sim}(v_i, v_j)$ is the similarity between user v_i and user v_j, which is calculated via cosine similarity between their corresponding vectors P_{v_i} and P_{v_j}. $\mathrm{trust}(v_i, v_j)$ is the trust value assigned to v_j from v_i. If the trust value is given, we set $\mathrm{trust}(v_i, v_j) = 1$. i.e., it represents the existence of a social relationship from v_i to v_j. Otherwise, $\mathrm{trust}(v_i, v_j) = 0$. S_{v_i} is the user set, including the users connected by v_i.

The social similarity based trust degree between two nodes measures the effect of local network topologies. When two nodes in a social network have two or more overlapped neighbours, they tend to have the community similarity which is a higher level of node similarity. The adjacent node sets of nodes v_i and v_j are defined as $N(v_i)$ and $N(v_j)$, respectively. The social similarity based trust degree is defined as

$$T^O_{v_i,v_j} = \frac{\sum_{t\in N(v_i),N(v_j)}\frac{1}{D(t)}}{\sqrt{\sum_{t\in N(v_i)}\frac{1}{D(t)}}\sqrt{\sum_{t\in N(v_j)}\frac{1}{D(t)}}} \tag{22}$$

where $D(t)$ represents the degree of node t.

Remark 2: It should be noted that (21) and (22) are only applicable to the scenario of two users who connect with each other directly. In terms of the scenario of two users who do not connect directly, the trust degree is calculated by using multiplication as the trust propagation operator. In addition, we only consider the shortest path if multiple trust propagation paths exist in a network.

We contribute to integrate the trust degree into Deep Semi-NMF model. In terms of the representation of Deep Semi-NMF approaches, the user's preference is the only factor which determines the estimated rating assigned to an item,

and it can be described as: $\hat{R}_{m,n} = P_m^T Q_n$. The recommendations from the target users' trusted friends should also be taken into consideration. Additionally, we distinguish the favors of user m's trust friends and user m, instead of using the favors of user m directly. Namely, after obtaining user m's favors on item n, i.e., $P_m^T Q_n$, we adjust it by the biases from his/her trusted friends' favors. As a consequence, the difference of the estimated ratings between he/she and his/her trust friends is expressed as

$$B_{m,n} = \sum_{v \in S_m} T_{m,v}(P_v^T Q_n + \Delta_{m,v} - P_m^T Q_n), \tag{23}$$

where $\Delta_{m,v}$ indicates the average bias on the rating between m and v.

For example, there are two users m and v, and they have different rating assignment behaviors. User m is generous and he/she usually assigns high scores to the items, while user v is critical and usually assigns low scores to the items. We assume the ratings for item n from these two users are 5 and 3, respectively. They have different preferences on item n at first glance. However, if the derivations on the ratings and their rating assignment behaviors are analyzed, the score 3 is almost the highest score allocated by v in his/her rating assignment history. From this perspective, we can conclude that both m and v have preferences for item n.

Additionally, we need to consider the biases of users on items. For instance, when we predict the rating assigned on the item n by the user m, if we know that the average rating over all items assigned by m is 3, and m assigns the rating 2.5 to item n, which is 0.5 lower than the average rating. Moreover, user m is critical, and he/she usually rate 0.4 lower than the mean. Finally, the estimated rating of n from m would be 2.1 (2.5-0.4). Therefore, we extend (23) with the biases of users and items as follows

$$\begin{aligned} B_{m,n} = &\sum_{v \in S_m} T_{m,v}(P_v^T Q_n + \Delta_{m,v} - P_m^T Q_n) \\ &+ \text{avg} + \text{bias}_m + \text{bias}_n, \end{aligned} \tag{24}$$

where the parameters bias_m and bias_n indicate the biases of user m and item n, respectively. Thus based on (24), the cost function for LRDMF integrated trust degree (LRDMF-TD) can be rewritten as

$$\begin{aligned} L(P,Q) = &\frac{1}{2}\min_{P,Q} \sum_{m=1}^{M}\sum_{n=1}^{N} I_{mn}(R_{m,n} - R_{m,n}^{LR}) \\ &+ \sum_{v \in S_m} T_{m,v}(P_v^T Q_n + \Delta_{m,v} - P_m^T Q_n) \\ &+ \frac{\lambda_P}{2}\|P\|_F^2 + \frac{\lambda_Q}{2}\|Q\|_F^2, \end{aligned} \tag{25}$$

R_{mn}^{LR} denotes the estimated rating on item n assigned by user m via LRDMF-TD.

In order to prevent over-fitting and reduce the complexity of the model, $\lambda_P = \lambda_Q$ is set in our experiments. It can be seen that all users are considered in the model for minimizing the whole difference when the social trust relationship are considered. The global minimum of L cannot be achieved because of its inherent inner structure [66] of the matrix factorization model. Fortunately, based on the gradient descent on P_m and Q_n for user m and item n, the local minimum of the cost function can be expressed as

$$\frac{\partial L}{\partial P_m} = \sum_{n=1}^{N} I_{mn}(R_{m,n} - R_{m,n}^{LR}) - \sum_{v \in S_m} T_{m,v} Q_n + \lambda_P P_m, \tag{26}$$

$$\frac{\partial L}{\partial Q_n} = \sum_{m=1}^{M} I_{mn}(R_{m,n} - R_{m,n}^{LR}) + \sum_{v \in S_m} T_{m,v}(P_v^T - P_m^T) + \lambda_Q Q_n. \tag{27}$$

The cost function for NLRDMF integrated trust degree (NLRDMF-TD) can be rewritten as

$$\begin{aligned} L(P,Q) = &\frac{1}{2}\min_{P,Q} \sum_{m=1}^{M}\sum_{n=1}^{N} I_{mn}(R_{m,n} - R_{m,n}^{NLR}) \\ &+ \sum_{v \in S_m} T_{m,v}(P_v^T Q_n + \Delta_{m,v} - P_m^T Q_n), \end{aligned} \tag{28}$$

where R_{mn}^{NLR} denotes the estimated rating on item n assigned by user m by NLRDMF-TD. The derivation of gradient descent on P_m and Q_n for NLRDMF-TD is similar as (26) and (27), respectively

$$\frac{\partial L}{\partial P_m} = \sum_{n=1}^{N} I_{mn}(R_{m,n} - R_{m,n}^{NLR}) - \sum_{v \in S_m} T_{m,v} Q_n + \lambda_P P_m, \tag{29}$$

$$\begin{aligned} \frac{\partial L}{\partial Q_n} = &\sum_{m=1}^{M} I_{mn}(R_{m,n} - R_{m,n}^{NLR}) + \sum_{v \in S_m} T_{m,v}(P_v^T - P_m^T) \\ &+ \lambda_Q Q_n. \end{aligned} \tag{30}$$

B. Autoencoder Ensemble

In this subsection, we exploit a variant of autoencoder, denoising autoencoder (DAE), to extract the latent representation in the hidden layer of this network. The latent representation of the social relationship can be enhanced from the trust relationship matrix $T \in \mathbb{R}^{M \times M}$ based on DAE. Then the latent representation of the users can be approximated as much as possible by the learned latent representation based on DAE.

1) Marginalized Denoising Autoencoders: Autoencoder is one kind of neural network, and it attempts to copy its input to its output after training the autoencoder which has a hidden layer in the interior of itself. A basic autoencoder consists of two components. An encoder is expressed as an activation function $d(\cdot)$ mapping an input data T into a hidden layer, and the representation of this hidden layer is $f(T)$. A decoder is expressed as a deactivation function $f(\cdot)$ mapping the hidden representation back into the output, and the representation of this reconstructed version of T is $f(d(T))$. In order to learn

the most significant features of data distribution from the input trust relationship matrix T, the activation and deactivation functions, such as sigmoid and identify functions, should be selected properly. In this section, the identify and the sigmoid functions are selected as the activation and deactivation functions, respectively.

A DAE is a variant of autoencoder whose input is the corrupted data, and the output is the original, uncorrupted data based on the training of DAE. The randomly generated artificial noise, such as binary masking noise or Gaussian noise, can be injected into the input data with a probability p. From a deep learning perspective, multiple DAEs can be stacked sequentially [26]. The input of the lth DAE is the output of the $l-1$th DAE performing as a hidden representation. Therefore, the DAEs contained in the stacked DAEs (SDAE) have to be trained in an iterative way. This layer by layer operation has high computational burden due to the learning of the model parameter in each layer training [56]. In order to overcome the defects of SDAE, a modified version, SDAE with marginalized corruption, is performed in [26]. In contrary to the two-level encoder and decoder in SDAE, we utilize a weight matrix W to map the corrupted input T^i into the reconstructed output by minimizing the squared loss function as follows

$$\frac{1}{2I}\sum_{i=1}^{I}\left\|T^i - W\tilde{T}^i\right\|_F^2, \tag{31}$$

where I is the number of samples in the input data, $W \in \mathbb{R}^{M\times M}$ is the mapping consisting of the reconstructed weights, and $\tilde{T}^i$ is the corrupted version of the original, uncorrupted T^i. c passing with different corruptions have been taken into the input data to reduce the variance. Then the c-times version of T can be represented as $\bar{T} = [T, T, \ldots, T] \in \mathbb{R}^{M\times cM}$. The corrupted version of $\bar{T}$ can be defined as $\tilde{T} \in \mathbb{R}^{M\times cM}$. We can rewrite (31) as

$$\min_W \left\|\bar{T} - W\tilde{T}\right\|_F^2. \tag{32}$$

When c approximates the positive infinity, there are infinity copies of corrupted input data. The mapping weight matrix W has the following closed-form solution

$$W = E[U]E[V]^{-1}, \tag{33}$$

where $U = \tilde{T}\tilde{T}^T$, $V = \bar{T}\tilde{T}^T$. Thus we do not have to solve the highly non-convex optimization problem in each layer when we train each layer based on iterative procedure to learn the model parameters. In the procedure of obtaining the weight matrix W, the computational complexity for training a marginalized DAE is reduced significantly. From a deep learning perspective, multiple marginalized DAEs can be stacked sequentially [26]. The input of the lth marginalized DAE is the output of the $l-1$th marginalized DAE, which performs as a hidden representation.

2) Latent Representation of Trust Relationship Considering User Preferences: In order to integrate social relationship with the user preferences, we propose a framework that integrating the Deep Semi-NMF with marginalized SDAE to further improve the recommendation accuracy. The framework is shown in Fig. 3. The trust friends of user m can be represented as $T_m \in \mathbb{R}^M$, an m-dimensional binary vector, corresponding to the mth row of T. In order to compute the corrupted version $\tilde{T} \in R^{M\times cM}$ of the trust relationship matrix $\bar{T} \in R^{M\times cM}$, the generated binary masking noise is injected with a probability p based on the deep learning strategy of DAE [56]. Generally, T is sparse, and only the non-zero values of $\bar{T}$ representing the trust relationship are corrupted by the artificial noise.

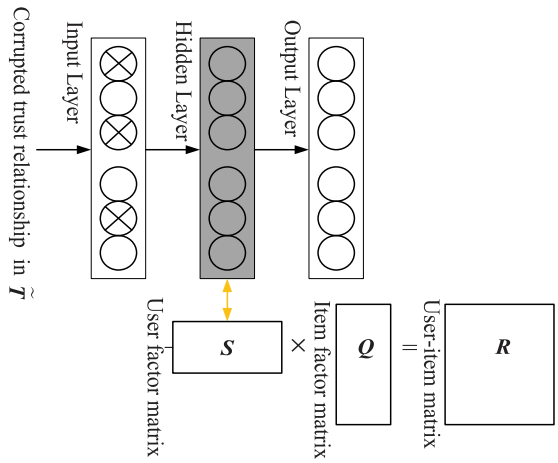


Fig. 3. Framework of mixed model. The orange double arrow indicates the coupling of the trust relationship represented by $\boldsymbol{W}$ and the user preference represented by $\boldsymbol{S}$.

We take both the weight matrix W and the user factor matrix S into consideration for improving the recommendation accuracy. The latent representation of the trust relationship, which should be as close as possible to the user factor matrix $S \in \mathbb{R}^{M\times K}$, can be enhanced from the trust relationship matrix $T \in \mathbb{R}^{M\times M}$ based on the DAE. The orange double arrow indicates the coupling of the trust relationship represented by W and the user preference represented by S.

3) Latent Representation of Trust Relationship: We map an m-dimensional binary vector $T_m \in \mathbb{R}^M$ into a latent space represented as $T'_m = S^T T_m \in \mathbb{R}^K$, $m = 1, 2, \ldots, M$, where $S \in \mathbb{R}^{M\times K}$ is the user factor matrix, and K is the number of latent factors. Different from the similarity measurement in (21), the user similarity can be characterized by the inner product. Thus the similarity in the latent space can be expressed as the inner product of the user factor matrices of two users in the users' latent space

$$\begin{aligned}\mathrm{sim}(T_m T_v) &= T'_m T'_v = (S^T T_m)^T S^T T_v \\ &= T_m^T S S^T T_v^T.\end{aligned} \tag{34}$$

In order to integrate the weight matrix W and the user factor matrix S, (34) can be formulated in a matrix form as

$$\min_{W,S}\left\|T^T S S^T T - WT\right\|_F^2, \tag{35}$$

where $W \in \mathbb{R}^{M\times M}$ is the weight matrix mapping the trust relationship matrix T into the hidden layer, and the latent

representation of the trust relationship matrix T in the hidden layer corresponds to the product $WT \in \mathbb{R}^{M\times M}$. In order to learn the weight matrix W based on the corrupted version $\tilde{T} \in R^{M\times cM}$, we minimize the objective function as

$$\min_{W,S} \left\|T^T SS^T T - WT\right\|_F^2 + \left\|\bar{T} - W\tilde{T}\right\|_F^2, \tag{36}$$

where c-times version of T is expressed as $\bar{T} \in \mathbb{R}^{M\times cM}$, and the corrupted version of $\bar{T}$ is expressed as $\tilde{T} \in \mathbb{R}^{M\times cM}$.

4) Community Regularization: Intuitively, community refers to some dense groups in a network. The nodes within each community are closely connected, but the connections among various communities are sparse. In social networks, the users who share the same opinions or interests tend to form a community [67]. It means that the opinion of one user can be affected by the opinions of other users in the identical community. Thus we need to integrate the community effect into the objective function to improve the recommendation performance. We need to introduce some trust based parameters used in the community detection algorithm.

Trust Potential. As we know, the trust degree of two users in a social network decrease as the distance of two users increases. Thus we need to define a parameter to measure the trust degree of inter-node objectively. Given a social network $G(V, E)$, user $v_i \in E$ is randomly selected from this network. We adopt $U(v_i) = v_1, v_2, \ldots, v_n$ to denote the users closely connected with v_i in this network. The trust potential of user v_i at user v_j is defined by the Gaussian potential-function as [68]

$$p_{v_i,v_j} = \exp\left(\frac{T^2_{v_i,v_j}}{2\sigma^2}\right), \tag{37}$$

where the user interaction range is control by the parameter σ, and σ is determined by the network details. The trust potential for user v_i is expressed as

$$p_{v_i} = \sum_{v_j \in U(v_i)} \exp\left(\frac{T^2_{v_i,v_j}}{2\sigma^2}\right). \tag{38}$$

Local High-Potential User. A community usually consists of a cluster center and its neighbours. In order to detect a community in a social network, we need to identify the high-potential users in the local network as the initial cluster centers. Given the adjacent users of user v as $N(v) = u_1, u_2, \ldots, u_n$. User v is a local high-potential user if it satisfies

$$p(v) \leq \max p(v, u_1), p(v, u_2), \ldots, p(v, u_n). \tag{39}$$

Trust Condensation. We define trust condensation to identify if a cluster has a good structure. Given a cluster C_i and its center node $u_i \in C$, the trust condensation of C is defined as [68]

$$CT(C_i, u_i) = \begin{cases} w_{low}C_{low} + w_{up}C_{up} & \text{if } \underline{C_i}, \overline{C_i} - \underline{C_i} \neq \phi \\ C_{low} & \text{if } \underline{C_i} \neq \phi, \overline{C_i} - \underline{C_i} = \phi \\ C_{up} & \text{if } \underline{C_i} = \phi, \overline{C_i} - \underline{C_i} \neq \phi \end{cases} \tag{40}$$

where $C_{low} = \sum_{v_i \in \underline{C_i}} p(u_i, v_i)$ and $C_{up} = \sum_{v_i \in \overline{C_i} - \underline{C_i}} p(u_i, v_i)$. $\overline{C_i}$ and $\underline{C_i}$ stand for the lower approximation set and the upper approximation set of clustering C_i, respectively. The weight for the lower approximation set and the upper approximation set of clustering C_i is defined as w_{low} and w_{up}, respectively and $w_{low} + w_{up} = 1$. The trust potential of center user u_i on user v_i is denoted as $p(u_i, v_i)$.

For all $v_i \in V$, the potential difference σ in C_i and C_l is defined as $\sigma = p(v_i, C_l) - p(v_i, C_j)$. If $\sigma \leq \beta$, i.e., the potentials of v_i in two clusters are similar, we assign v_i to the upper approximation set of the intersection of C_i and C_l; otherwise, to the lower approximation set of C_l. Based on the definition of trust condensation, we can update the cluster as

$$u_i = u | u \in C_i \wedge CT(C_i, u) = \max_{x \in C_i} CT(C_i, x). \tag{41}$$

Overlapping Clusters. Given two clusters C_i and C_j, their overlapping clustering degree is defined as [68]

$$\text{Over}(C_i, C_j) = \frac{|C_i \bigcap C_j|}{\min(|C_i|, |C_j|)}, \tag{42}$$

where $\min(|C_i|, |C_j|)$ gives the size of the smaller cluster of C_i and C_j. The range of $\text{over}(C_i, C_j)$ falls into [0,1].

The overlapping community detection algorithm considering trust-based characteristic can be summarized as follows.

1) The trust degree between different users and the trust potential of each user in a social network are computed via (20) and (38), respectively.

2) The high trust potentials of the users in a social network can be identified via (39).

3) According to the trust potentials, the users in the networks can be classified and placed into clustering upper approximation and the clustering lower approximation sets by exploiting K-medoids clustering, respectively. The clustering center can be updated based on (41) after computing clustering upper approximation and the clustering lower approximation sets. The classification can be terminated until the clustering centers reach a stable state.

4) Different clusters can be merged if most of the users in different clusters are overlapping.

As we know, the users in the same community tend to share similar preferences on items with their trusted friends, who are usually regarded as the neighbors of the target user. We can utilize the meaningful information of these neighbours to improve the prediction accuracy. The neighbors of user u can be defined as

$$N(u) = \{v | v \in C \wedge u \in C, u \neq v\}, \tag{43}$$

where C is the community that contains user u. It should be noted that multiple communities can contain user u, and thus

all the communities containing u should be taken into consideration.

The behavior of the given user u would be affected by his/her neighbors $N(u)$ because of the community effect. It means that the behavior difference between the given user and his/her neighbours should be minor. This phenomena can be expressed in a mathematical form by minimizing the following formulation:

$$\left\| P_u - \frac{1}{|N(u)|} \sum_{v \in N(u)} T_{u,v} P_v \right\|_F^2. \tag{44}$$

The above equation can be utilized to minimize the preference between a user and his/her neighborhood to an average level. It means that a user's preference should be similar to the general preferences of all neighbours $N(u)$.

5) Parameters Training: By integrating the matrix factorization technique and the community effect with (36), we have the joint objective function as follows

$$\begin{aligned} \min_{W,S,Q} \mathcal{L} = & \|R - SQ\|_F^2 + \left\|T^T S S^T T - WT\right\|_F^2 \\ & + \left\|\bar{T} - W\tilde{T}\right\|_F^2 + \lambda(\|S\|_F^2 + \|Q\|_F^2) \\ & + \frac{\mu}{2} \sum_{u=1}^{m} \left\| S_u^T - \frac{1}{|N(u)|} \sum_{v \in N(u)} T_{u,v} S_v^T \right\|_F^2, \end{aligned} \tag{45}$$

where μ is regularization parameter, and the coefficient λ of Frobenius norm of S and Q are utilized for controlling the overfitting problem of model parameters. The third term couples the trust relationship with user preferences as mentioned in the former subsection.

Since the optimization function in (45) is a non-convex problem involving the matrices W, S and Q, we utilize a sub-optimal strategy to solve this problem. In each iteration, we fix two variables and update one variable as an alterative sub-optimal strategy.

By discarding the irrelevant term with respect to W in (45), we can reformulate the objective function by only considering W and fixing S and Q by minimizing the following optimal problem as

$$\min_W \left\|T^T S S^T T - WT\right\|_F^2 + \left\|\bar{T} - W\tilde{T}\right\|_F^2. \tag{46}$$

According to (33), W has a closed solution $W = E[U]E[V]^{-1}$, where $U \in \mathbb{R}^{M \times M}$, and $V \in \mathbb{R}^{M \times M}$. They can be updated using the equations as follows

$$U = \tilde{T}\tilde{T}^T + TSS^T TT^T, V = \bar{T}\tilde{T}^T + TT^T. \tag{47}$$

Then, by discarding the irrelevant term with respect to matrices S and Q in (45), the objective function (45) can be rewritten as

$$\begin{aligned} \mathcal{L}(S,Q) = & \operatorname{tr}[(R - SQ)(R - SQ)]^T \\ & + \operatorname{tr}[(T^T S S^T T - WT)(T^T S S^T T - WT)^T] \\ & + \frac{\mu}{2} \sum_{u=1}^{m} \left\| S_u^T - \frac{1}{|N(u)|} \sum_{v \in N(u)} T_{u,v} S_v^T \right\|_F^2 \\ & + \lambda \operatorname{tr}[SS^T + Q^T Q]. \end{aligned} \tag{48}$$

By taking the partial derivations of (48) with the matrices S and Q, we have

$$\begin{aligned} \frac{\partial \mathcal{L}(S,Q)}{\partial S} = & 2(-RQ^T + SQQ^T) \\ & + 2TT^T SS^T (TT^T S + TT^T SS^T) \\ & - 2T(T^T W^T T^T S + WTT^T S) + 2\lambda S \\ & + \mu \left\| S_u^T - \frac{1}{|N(u)|} \sum_{v \in N(u)} T_{u,v} S_v^T \right\|_F^2, \\ \frac{\partial \mathcal{L}(S,Q)}{\partial Q} = & 2(-R^T S + QS^T S + \lambda Q^T). \end{aligned} \tag{49}$$

Then, we can update the model parameter S and Q based on the classical gradient descent method, which are expressed as

$$\begin{aligned} S^{(t+1)} &= S^{(t)} - \eta_1 \frac{\partial \mathcal{L}^t}{\partial S^{(t)}}, \\ Q^{(t+1)} &= Q^{(t)} - \eta_2 \frac{\partial \mathcal{L}^t}{\partial Q^{(t)}}, \end{aligned} \tag{50}$$

where t stands for the tth iteration, and η_1 and η_2 stand for the learning rates. The maximum number of iterations is fix at 1000. The terminal rule is that the difference between two adjacent iterations satisfies $\mathcal{L}^{(t+1)} - \mathcal{L}^{(t)}/\mathcal{L}^{(t)} 1e-05$, where $\mathcal{L}^{(t)}$ is the value of (48) in the tth iteration.

6) DMF-Based Initialization for Marginalized DAEs: From a deep learning perspective, multiple marginalized DAEs as shown in Fig. 3 can be stacked sequentially, which is called Deep-MDAEs. The input of the lth marginalized DAE, which performs as a hidden representation, is the output of the $(l-1)$th marginalized DAE. If there are L layers in the Deep-MDAEs, the deepest layer will be the $(L+1)/2$th layer. In different hidden layers, we have different latent representations for trust relationships. Thus the latent representations of the deepest layer in the Deep-MDAEs should be as close as the user factor matrix S. As shown in (33), the close-form expression of the weight matrix $W = E[U]E[V]^{-1}$, which has been updated based on (47). However, in this paper, we update U by using $U = \tilde{T}\tilde{T}^T$ until we reach the $(L+1)/2$th layer, i.e., the deepest layer. In the $(L+1)/2$th layer, we update U by using $U = \tilde{T}\tilde{T}^T + TSS^T TT^T$, where S is update based on the LRDMF and NLRDMF methods in (10) and (18), respectively. Then, the model parameter S and Q can be updated based on (50). The final recommendation matrix can be calculated by $\hat{R} = \hat{S}\hat{Q}$. The Deep-MDAE initialized by LRDMF and NLRDMF without community effect are called LRDMF-DMDAE and NLRDMF-DMDAE, respectively. The Deep-

MDAE initialized by NLRDMF with community effect is called NLRDMF-DMDAECE.

The steps of DMF-DMDAECE are summarized as follows:

1) We perform the deep matrix factorization for the known part of the user-item rating matrix R to obtain better initializations of latent user and item feature matrices;

2) We calculate the trust degree based on (20) including the similarity and the social based trust degrees, respectively;

3) The DMF-TD algorithms can be achieved by optimizing (25) and (28);

4) We perform the overlapped community detection algorithm to detect the community in a trust relationship network;

5) We formulate a new joint objective function (45) by taking trust information and community effect into consideration;

6) We construct the marginalized DAEs to optimize the objective function (45);

7) We utilize the results of deep matrix factorization to initialize the marginalized DAEs;

8) We obtain the final result by training the marginalized DAEs.

VI. Experiments

In the field of trust-aware recommendation, the publicly available and suitable dataset is rare, we mainly adopt the following two datasets:

1) Epinions: Epinions is available freely [42], which is composed of 49 290 users and 139 738 items. The number of ratings and trust relationships contained in Epinions are 664 824 and 487 181, respectively. The scale of rating is from 1 to 5. We build a social trust network by using these records. Each user in Epinions keeps a trust relationship with others. In addition, the density of the user-item rating matrix is less than 0.01%.

2) Flixster: This is a social network allowing users to assign scores for movies [39]. It consists of 1 049 445 users who have rated 492 359 different items. The total number of ratings is 8 238 597. The total number of trust relationships is 26 771 123. The density of the rating matrix is lower than 0.0016%.

The rating matrixes extracted from Epinions and Flixster are both sparse. The density of Movielens, which consists of 6040 users, 3900 movies, and 1 000 209 ratings, is 4.25%, and the density of Eachmovie, which consists of 74 424 users, 1648 movies, and 2 811 983 ratings, is 2.29% [38]. Therefore, Epinions and Flixster are both ideal sources for make our trust-aware recommendations.

We use three metrics, the Root Mean Square Error (RMSE), the precision and F-Measure, to measure the performance of our proposed methods, i.e., LRDMF-TD, NLRDMF-TD, LRDMF-DMDAE, NLRDMF-DMDAE and NLRDMF-DMDAECE comparing with other the state-of-the-art recommendation methods. The metrics RMSE for measuring the error in recommendation is defined as

$$\mathrm{RMSE} = \sqrt{\frac{\sum_{m,n}(R_{m,n} - \bar{R}_{m,n})^2}{T_{\mathrm{test}}}}, \tag{51}$$

where $R_{m,n}$ denotes the rating assigned to item n by user m, $\bar{R}_{m,n}$ denotes the predicted rating assigned to item n by user m via a method, and T_{test} denotes the number of tested ratings. Meanwhile, most recommendation approaches cannot deal with the task that predicting all the ratings in the test data under the scenario of high sparse data. Therefore, the metric coverage rate can be adopted to measure the proportional of $\langle \mathrm{user}, \mathrm{item} \rangle$ pairs, and the values can be predicted as

$$\mathrm{coverage} = \frac{S}{T_{\mathrm{test}}}, \tag{52}$$

where S represents the number of ratings being predicted, and T_{test} represents the number of ratings being tested. Moreover, we integrate RMSE and coverage to form a full metric following the F-Measure's example. Therefore, the RMSE has to be converted into the metric of precision, whose value is distributed in the range of [0, 1]. We formulate the precision as: $\mathrm{precision} = 1 - \frac{\mathrm{RMSE}}{4}$. It can be inferred from this equation that the maximum possible error is 4, since all rating values are between 1 and 5. The definition of F-Measure is given as

$$F_{\mathrm{Measure}} = \frac{2 \times \mathrm{precision} \times \mathrm{coverage}}{\mathrm{precision} + \mathrm{coverage}}. \tag{53}$$

The initializations of most existing matrix factorization methods are straightforward, i.e., P and Q are initialized as dense matrices consisting of random numbers. We propose specific initialization methods in this paper, and compare it with random initialization, K-means initialization, normalized-cut (Ncut) initialization [69] and SVD-based initialization [70], autoencoder initialization [44] in order to verify the superiority of initialization with LRDMF and NLRDMF based approaches. Moreover, we remove the community detection algorithm in [44] for fair comparison, and then the method in [44] is called Auto-TD.

The dimension of the feature matrices is set to be $K = 80$. The DMF model consists of two representation layers (1260-625), and the scaled hyperbolic tangent $stanh(x) = \alpha tanh(\beta x)$ with $\alpha = 0.7159$ and $\beta = \frac{2}{3}$ is used as the non-linearity function. For DMDAE based methods, the regularization parameter λ is 0.1, and the number of the stacked MDAEs is 10. For community detection, we use 2-trust-cliques. For Epinions and Flixster dataset, the regularization parameter are set as $\mu = 10$ and $\mu = 5$, respectively. We set the parameter σ to 1.886, the clustering overlapping threshold to 0.75, W_{up} is set to 0.1 and the weight parameter $\beta = 0.6$. The percentage of the input data corrupted by the binary masking noise is 50%.

It can be seen from Fig. 4 that the RMSEs of LRDMF and NLRDMF are much smaller than the RMSEs of other approaches. In particular, the RMSEs of DMF based approaches are smaller than those of the Auto-TD approach proposed in [44]. This is because that LRDMF and NLRDMF extract more abstract features from the original space compared with other approaches, and the initialized latent feature matrices learned by LRDMF and NLRDMF make (25) and (28) more closer to the global minimum. The RMSE of NLRDMF is smaller than that of LRDMF because of the non-linearity learned by NLRDMF. The RMSEs of trust DAE based methods are smaller than those of the trust degree based methods. This is

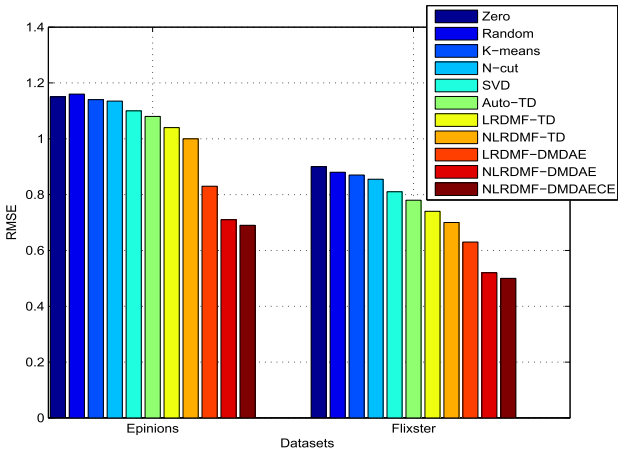


Fig. 4. Effect of different initialization for training model.

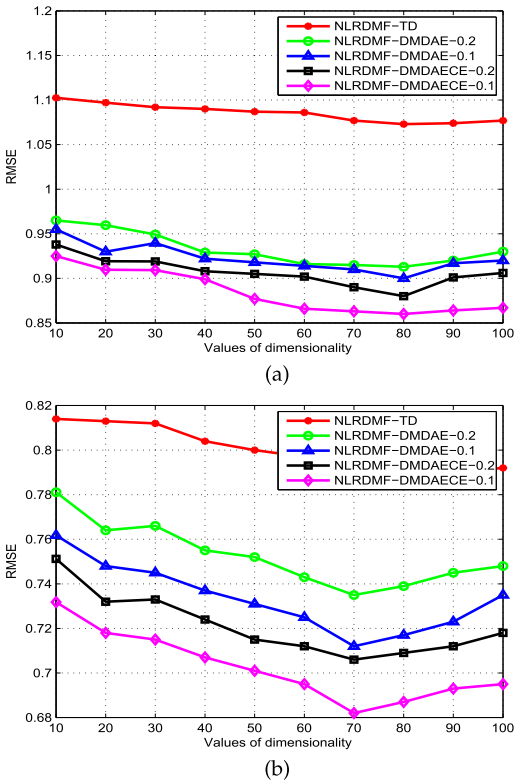


Fig. 5. Effect of different values with dimension K. (a) Epinions. (b) Flixster.

caused by the reason that the deep structure of the DAE based methods enforces the latent represent of the trust relationship as much as the user factor matrix S. NLRDMF-DMDAECE performs best of all since we take the community effect into consideration.

In order to find the best dimension K, the RMSEs versus various dimensions for NLRDMF-TD and NLRDMF-DMDAE are depicted in Fig. 5. λ_P and λ_Q are fixed at 0.1. It can be seen that there is a turning point around 80 for Epinions, and there is a turning point around 70 for Flixster. The main reason is that a relative larger dimension can improve the prediction accuracy. However, when the number of dimension is too large, the

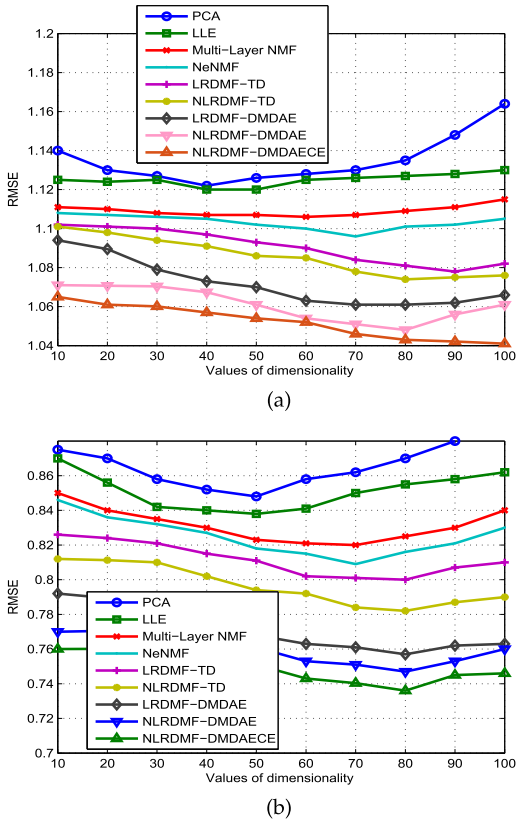


Fig. 6. RMSE versus values of dimension with different approaches. (a) Epinions. (b) Flixster.

overfitting may exist which leads to the degradation of the prediction accuracy.

In order to have an intuitive expression about the relationship between the value of $\lambda_P = \lambda_Q$ and the best dimension K, we draw different values of $\lambda_P = \lambda_Q = 0.2, 0.1$, respectively, with varying the best dimension K. It can be seen from Fig. 5 that when $\lambda_P = \lambda_Q = 0.1$, our proposed framework NLRDMF-DMDAECE performs best of all.

In our model, the LRDMF and NLRDMF approaches are used to reduce dimension and extract features from the user-item rating matrix. We compare these two approaches with not only classical algorithms such as principal component analysis (PCA) [71] and locally linear embedding (LLE) [72] but also with other NMF variants such as multi-layer NMF [73] and NeNMF [74] for pretraining in order to validate the effectiveness of feature extraction. It can be seen from Fig. 6 that compared with the classical approaches, the four NMF variants approaches can improve the prediction accuracy greatly. The RMSEs of LRDMF-TD and NLRDMF-TD approaches are smaller than those of multi-layer NMF and NeNMF approaches. This is because that LRDMF-TD and NLRDMF-TD approaches extract better features compare with Multi-Layer NMF and NeNMF approaches. The RMSEs of LRDMF-DMDAE and NLRDMF-DMDAE methods are smaller than those of LRDMF-TD and NLRDMF-TD methods. This is caused by the reason that the deep structure of the LRDMF-DMDAE and NLRDMF-DMDAE methods enforces the latent represent of the trust relationship as much as the

TABLE I
COMPARISON RESULTS WITH ALL USERS

	Epinions			Flixster		
Methods	RMSE	Coverage	F-Measure	RMSE	Coverage	F-Measure
UserCF	1.3436	18.11%	0.2846	0.8552	64.64%	0.7095
ItemCF	1.3621	20.55%	0.3134	0.8432	68.21%	0.7318
TidalTrust	1.2571	37.21%	0.4824	0.8359	75.81%	0.7742
MoleTrust	1.2824	37.36%	0.4821	0.8243	76.11%	0.7772
BMF	1.2176	69.88%	0.6972	0.8160	87.66%	0.8344
STE	1.1733	78.45%	0.7429	0.8056	90.60%	0.8489
SocialMF	1.1236	82.37%	0.7682	0.7950	90.32%	0.8490
CDL	1.0821	86.44%	0.7912	0.7940	92.55%	0.8590
NLRDMF	1.0713	88.66%	0.8003	0.7898	94.21%	0.8661
TD	1.3826	16.66%	0.2621	0.8677	60.23%	0.6821
NLRDMF-TD	1.0633	89.66%	0.8073	0.7821	96.31%	0.8767
DMDAE	1.0733	88.36%	0.8153	0.7835	94.31%	0.8634
NLRDMF-DMDAE	1.0547	91.69%	0.8256	0.7965	98.28%	0.8854
NLRDMF-DMDAECE	1.0454	92.76%	0.8343	0.8054	98.88%	0.8986

user factor matrix S. More latent representations in the hidden layers have been learned compared with the trust degree. The RMSE of NLRDMF-DMDAECE is the smallest of all because it takes both trust information and community effect into consideration.

We compare our proposed approach with the following baseline methods to show the superiority of our proposed methods. It should be noted that NLRDMF approach is used for the initialization of DMFTrust since it has a better prediction accuracy compared with LRDMF.

1) *UserCF:* This is a typical user-based collaborative filtering method, and it utilizes the users' similarity for predicting missing values.

2) *ItemCF:* This is a typical item-based collaborative filtering method, and it capture the items' similarity for predicting missing values.

3) *TidalTrust:* A trust inference algorithm is used for recommendation [41].

4) *MoleTrust:* This algorithm can promote trust in social networks, and the trust weight corresponds to the similarity weight [42].

5) *BMF:* This is the basic matrix factorization method proposed in [17], and the trust social network is not considered.

6) *STE:* This method combines users' preferences with their trusted friends' favors [38].

7) *SocialMF:* This method fuses the trust propagation in recommendation systems [39]. The parameter λ is set to be 5, which provides the best recommendation performance in this experiment.

8) *CDL:* This is a deep learning-based method proposed in [29]. However, the content information and the trust network are not used.

9) *NLRDMF:* The Non-Linear Representation DMF part of NLRDMF-TD is considered, while Trust Degree (TD) is not considered.

10) *TD:* The Trust Degree (TD) part of NLRDMF-TD is considered, while the Non-Linear Representation DMF is not considered.

11) *NLRDMF-TD:* The proposed NLRDMF-TD method that considers both Non-Linear Representation DMF and the Trust Degree (TD).

12) *DMDAE:* The Deep MDAE part of NLRDMF-DMDAE is considered, while NLRDMF is not considered. The basic matrix factorization is used for initialization.

13) *NLRDMF-DMDAE:* The proposed NLRDMF-DMDAE method that considers both Non-Linear Representation DMF and Deep MDAE.

14) *NLRDMF-DMDAECE:* The proposed NLRDMF-DMDAECE method that considers community effect.

In this section, we mainly analyze the comparison results with different approaches and datasets. Specifically, we compare the above methods with our proposed methods by using Epinions and Flixster datasets, respectively, which have different data sparsity. In terms of the parameters, the dimension of latent feature matrix is fixed at $K = 80$ for Epinions dataset and $K = 70$ for Flixster dataset. As a result, we can see from Table I that the NLRDMF-TD and NLRDMF-DMDAE outperform other methods, and the STE and SocialMF methods outperform the BMF method, which only adopts the user-item rating matrix for recommendation. Besides, the TidalTrust and MoleTrust methods are superior to BMF method. It can be known that the performance of collaborative filtering-based approaches are not well enough. It relies on the trusted friends' comments, and thus it is not suitable for sparse data. It can be known that UserCF and ItemCF cannot work well at this situation. TD method performs worse than the traditional UserCF and ItemCF based methods. The NLRDMF-DMDAECE performs best of all.

Experimental has been implemented to analyze the independent contributions benefited from the two steps, i.e., DMF and trust degree, in Table I and Table II. It can be seen that the performance of the trust degree (TD) based method is worse than that of the traditional UserCF and ItemCF methods. The NLRDMF performs better than the recent developed deep learning based method, and outperforms other traditional matrix factorization based methods as well. The coverage of NLRDMF is five times of that of TD, and the F-measure of NLRDMF is three times of that of TD. Thus we can conclude that "deep" is more important and beneficial than trust.

We regard cold-start users as those who have rating assignment behaviors less than 5 times [42]. Thus how to recommend items to cold-start users is still a challenge for

TABLE II
COMPARISON RESULTS WITH COLD-START USERS

	Epinions			Flixster		
Methods	RMSE	Coverage	F-Measure	RMSE	Coverage	F-Measure
UserCF	1.4658	12.31%	0.2061	0.9578	54.96%	0.6381
ItemCF	1.5421	13.46%	0.2208	0.9367	57.82%	0.6589
TidalTrust	1.3652	24.84%	0.3609	0.9274	68.32%	0.7232
MoleTrust	1.3724	27.66%	0.3893	0.9177	69.89%	0.7330
BMF	1.3264	54.67%	0.6015	0.8994	78.91%	0.7821
STE	1.2660	67.49%	0.6792	0.8578	85.69%	0.8197
SocialMF	1.2176	75.35%	0.7234	0.8212	86.45%	0.8281
CDL	1.1725	80.22%	0.7912	0.8120	89.55%	0.8434
NLRDMF	1.1513	82.62%	0.7985	0.7976	91.43%	0.8576
TD	1.654	11.86%	0.1956	0.9788	50.35%	0.5631
NLRDMF-TD	1.1435	82.93%	0.7515	0.7899	94.36%	0.8673
DMDAE	1.1733	83.27%	0.8085	0.7936	92.72%	0.8664
NLRDMF-DMDAE	1.0445	86.63%	0.8236	0.7754	94.65%	0.8975
NLRDMF-DMDAECE	1.0367	86.98%	0.8323	0.7832	94.90%	0.9023

recommender systems. In the Epinions, most users belong to cold-start users. Therefore, it is meaningful for recommending cold-start users with high effectiveness. In order to validate the performance superiority of our proposed methods compared with other methods, the cold-start users are picked out from the two datasets for comparing the recommendation performance of these methods. The final comparison results are shown in Table II. Our proposed methods outperform other methods for cold-start users. This indices that our proposed methods are more suitable for dealing with cold-start users. We can also obtain from both Table I and Table II that the improvement on our proposed methods for cold-start users is higher than that for all users.

VII. CONCLUSION

In this paper, based on deep learning technique and trust relationship, a novel trust-based deep learning method is proposed for recommendation by integrating community effect. Since the matrix factorization-based approaches rely much on the initialization of the latent feature matrices, a novel deep architecture for matrix factorization named DMF is proposed. This method can extract better features from the original space to improve the initialization accuracy. Then the DMF is integrated into the Deep-MDAE to extract the trust relationship. By taking the community effect into consideration, the recommendation accuracy is further improved. Our experimental results verify that our proposed methods outperform other baseline methods. In the future work, we will integrate the DMF technique into collaborative filtering approaches to further improve the recommendation performance.

REFERENCES

[1] F. Tang, L. T. Yang, C. Tang, J. Li, and M. Guo, "A dynamical and load-balanced flow scheduling approach for big data centers in clouds," *IEEE Trans. Cloud Comput.*, vol. 6, no. 4, pp. 915–928, Oct.-Dec. 2018.

[2] B. Zhou *et al.*, "Online internet traffic monitoring system using spark streaming," Big Data Mining and Analytics, vol. 1, no. 1, pp. 47–56, 2018.

[3] R. Li, H. Asaeda, J. Li, "A distributed publisher-driven secure data sharing scheme for information-centric IoT," *IEEE Internet Things J.*, vol. 4, no. 3, pp. 791–803, Jun. 2017.

[4] Z. Wang, Y. Zhang, Y. Li, Q. Wang, and F. Xia, "Exploiting social influence for context-aware event recommendation in event-based social networks," in *IEEE Int. Conf. Comput. Commun.*, Atlanta, GA, USA, May. 2017, pp. 1–4.

[5] J. Li and H. Kameda,"Load balancing problems for Multi-class jobs load balancing problems for multi-class jobs in distributed/parallel computer systems," *IEEE Trans. Comput.*, vol. 47, no. 3, pp. 322–332, Mar. 1998.

[6] H. Liu, F. Xia, Z. Chen, N. Yaw Asabere, J. Ma, and R. Huang, "TruCom: Exploiting domain-specific trust networks for multi-category item recommendation," *IEEE Syst. J.* , vol. 1, no. 1, pp. 295–304, Mar. 2017.

[7] R. Li, J. Li, and H. Asaeda, "A hybrid trust management framework for wireless sensor and actuator networks in cyber-physical systems," *IEICE Trans. Inf. Syst.*, vol. E97-D, no. 10, pp. 2586–2596, Oct. 2014.

[8] 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.

[9] R. Li and J. Li, "Requirements and design for neutral trust management framework in unstructured networks," *J. Supercomput.*, vol. 64, no. 3, pp. 702–716, Jun. 2013.

[10] Y. Koren, "Factor in the neighbors: Scalable and accurate collaborative filtering," *ACM Trans. Knowl. Discov. Data*, vol. 4, no. 1, pp. 1–24, Jan. 2010.

[11] Z. Tang, A. Liu, Z. Li, Y. Choi, H. Sekiya, and J. Li, "A trust-based model for security cooperating in vehicular cloud computing," *Mobile Inf. Syst.*, vol. 2016, Article ID 9083608, 22 pages, 2016.

[12] R. Li and J. Li, "Towards neutral trust management framework in unstructured networks," *Proc. IEEE Int. Conf. Mobile Ad Hoc Sensor Syst. (IEEE MASS 2009) Symp. Trust, Secur. Privacy Pervasive Appl.*, Macau, 2009, pp. 771–776.

[13] M. D. Ekstrand, J. T. Riedl, and J. A. Konstan, "Collaborative filtering recommender systems," *Found. Trends Hum.-Comput. Interact.*, vol. 4, no. 2, pp. 81–173, 2011.

[14] Y. Ren, G. Li, J. Zhang, and W. Zhou, "Lazy collaborative filtering for data sets with missing values," *IEEE Trans. Cybern.*, vol. 43, no. 6, pp. 1822–1834, Dec. 2013.

[15] E. J. Candes and B. Recht, "Exact matrix completion via convex optimization," *Found. Comput. Math.*, vol. 9, no. 6, pp. 717–772, Dec. 2009.

[16] N. Guan, D. Tao, Z. Luo, and B. Yuan, "Online nonnegative matrix factorization with robust stochastic approximation," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 23, no. 7, pp. 1087–1099, Jul. 2012.

[17] R. Kannan, M. Ishteva, and H. Park, "Bounded matrix factorization for recommender system," *Knowl. Inf. Syst.*, vol. 39, no. 3, pp. 491–511, Jun. 2014.

[18] M. Li, T. Zhang, Y. Chen, and A. Smola, "Efficient mini-batch training for stochastic optimization," in *Proc. 20th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2014, pp. 661–670.

[19] V. P. Pauca, J. Piper, and R. J. Plemmons, "Nonnegative matrix factorization for spectral data analysis," *Lin. Alg. Appl.*, vol. 416, no. 1, pp. 29–47, 2006.

[20] D. Cai, X. He, J. Han, and T. S. Huang, "Graph regularized nonnegative matrix factorization for data representation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 8, pp. 1548–1560, Aug. 2011.

[21] S. Jiang, K. Li, and R. Y. Xu, "Relative pairwise relationship constrained non-negative matrix factorisation," *IEEE Trans. Knowl. Data Eng.*, vol. 31, no. 8, pp. 1595–1609, Aug. 2019, doi: 10.1109/TKDE.2018.2859223.

[22] V. Kumar, A. K. Pujari, S. K. Sahu, V. R. Kagita, and V. Padmanabhan, "Collaborative filtering using multiple binary maximum margin matrix factorizations," *Inf. Sci.*, vol. 380, pp. 1–11, 2017.

[23] S. Zhang, L. Yao, and A. Sun, "Deep learning based recommender system: A survey and new perspectives," 2017, *arXiv:1707.07435*.

[24] G. K. Dziugaite and D. M. Roy, "Neural network matrix factorization," 2015, *arXiv:1511.06443*.

[25] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, "Neural collaborative filtering," in *Proc. 26th Int. Conf. World Wide Web*, 2017, pp. 173–182.

[26] S. Sedhain, A. K. Menon, S. Sanner, and L. Xie, "Autorec: Autoencoders meet collaborative filtering," in *Proc. 24th Int. Conf. World Wide Web*. ACM, 2015, pp. 111–112.

[27] F. Strub and J. Mary, "Collaborative filtering with stacked denoising autoencoders and sparse inputs," in *NIPS Workshop Mach. Learn. eCommerce*, 2015.

[28] M. van Baalen, "Deep matrix factorization for recommendation," Ph.D. dissertation, Informat. Institute, Univ. of Amsterdam, Amsterdam, Netherlands, 2016.

[29] H. Wang, N. Wang, and D. Y. Yeung, "Collaborative deep learning for recommender systems," in *Proc. 21th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2015, pp. 1235–1244.

[30] H. Wang and D. Y. Yeung, "Towards bayesian deep learning: A framework and some existing methods," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 12, pp. 3395–3408, Dec. 2016.

[31] J. Tang and K. Wang, "Personalized top-n sequential recommendation via convolutional sequence embedding," in *Proc. 11th ACM Int. Conf. Web Search Data Mining*. ACM, 2018, pp. 565–573.

[32] C.-Y. Wu, A. Ahmed, A. Beutel, A. J. Smola, and H. Jing, "Recurrent recommender networks," in *Proc. 10th ACM Int. Conf. Web Search Data Mining*, 2017, pp. 495–503.

[33] G. Trigeorgis, K. Bousmalis, S. Zafeiriou, and B. W. Schuller, "A deep matrix factorization method for learning attribute representations," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 3, pp. 417–429, Mar. 2017.

[34] H.-J. Xue, X.-Y. Dai, J. Zhang, S. Huang, and J. Chen, "Deep matrix factorization models for recommender systems." in *Proc. Int. Joint Conf. Artif. Intell.*, 2017, pp. 3203–3209.

[35] F. Zhang, J. Song, and S. Peng, "Deep matrix factorization for recommender systems with missing data not at random," in *Proc. J. Phys.: Conf. Series*, vol. 1060, no. 1, 2018, Art. no. 012001.

[36] H.-F. Yu, C.-J. Hsieh, S. Si, and I. S. Dhillon, "Parallel matrix factorization for recommender systems," *Knowl. Inf. Syst.*, vol. 41, no. 3, pp. 793–819, Dec. 2014.

[37] J. Li, R. Li, and J. Kato, "Future trust management framework for mobile Ad Hoc networks," *IEEE Commun. Mag.*, vol. 46, no. 4, pp. 108–114, Apr. 2008.

[38] H. Ma, I. King, and M. R. Lyu, "Learning to recommend with explicit and implicit social relations," *ACM Trans. Intell. Syst. Technol.*, vol. 2, no. 3, pp. 29–48, Apr. 2011.

[39] M. Jamali and M. Ester, "A matrix factorization technique with trust propagation for recommendation in social networks," in *Proc. ACM Conf. Recommender Syst.*, 2010, pp. 135–142.

[40] F. Xia, A. Mohammed Ahmed, L. Tianruo Yang, J. Ma, and J. Rodrigues, "Exploiting social relationship to enable efficient replica allocation in Adhoc social networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 312, pp. 3167–3176, Dec. 2014.

[41] J. A. Golbeck, "Computing and applying trust in Web-based social networks," Ph.D. dissertation, Dept. Comput. Sci., Univ. Maryland, College Park, MD, USA, 2005.

[42] P. Massa and P. Avesani, "Trust-aware recommender systems," in *Proc. ACM Conf. Recommender Syst.*, 2007, pp. 17–24.

[43] D. Rafailidis and F. Crestani, "Recommendation with social relationships via deep learning," in *Proc. ACM SIGIR Int. Conf. Theory Inf. Retrieval*, 2017, pp. 151–158.

[44] S. Deng, L. Huang, G. Xu, X. Wu, and Z. Wu,"On deep learning for trust-aware recommendations in social networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 5, pp. 1164–1177, May 2017.

[45] M. Wang, Z. Wu, X. Sun, G. Feng, and B. Zhang, "Trust-aware collaborative filtering with a denoising autoencoder," *Neural Process. Lett.*, pp. 1–15, 2018.

[46] Z. Zhang, Y. Liu, Z. Jin, and R. Zhang, "Selecting influential and trustworthy neighbors for collaborative filtering recommender systems," in *Proc. IEEE 7th Annu. Comput. Commun. Workshop Conf.*, 2017, pp. 1–7.

[47] G. Pitsilis and L.F. Marshall, "Modeling trust for recommender systems using similarity metrics," in *Proc. IFIPTM*, Trondheim, Norway, 2008, 263, pp. 103–118.

[48] F. Xia, L. Liu, J. Li, A. Mohammed Ahmed, L. Tianruo Yang, and J. Ma, "BEEINFO: An interest-based forwarding scheme using artificial bee colony for socially-aware networking," *IEEE Trans. Veh. Technol.*, vol. 64, no. 3, pp. 1188–1200, Mar. 2015.

[49] X. Han *et al.*, "CSD: A multi-user similarity metric for community recommendation in online social networks," *Expert Sys. Appl.*, vol. 53, pp. 14–26, 2016.

[50] A. L. Barabasi, H. Jeong, Z. Neda, E. Ravasz, A. Schubert, and T. Vicsek. "Evolution of the social network of scientific collaborations," *Phys. A: Stat. Mech. Appl.*, vol. 311, no. (3-4), pp. 590–614, 2002.

[51] L. Lu, Ci-Hang Jin, and T. Zhou., "Similarity index based on local paths for link prediction of complex networks," *Phys. Rev. E* 80.4, 2009, Art. no. 046122.

[52] F. Xia, J. Liu, H. Nie, Y. Fu, L. Wan and X. Kong, "Random Walks: A review of algorithms and applications," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 4, no. 2, pp. 95–107, Apr. 2020.

[53] X. Han, A. Cuevas, N. Crespi, R. Cuevas and X. Huang, "On exploiting social relationship and personal background for content discovery," in *p2p Netw. Future Gener. Comput. Syst.*, vol. 40, pp. 17–29, 2014.

[54] A. Grover and J. Leskovec., "node2vec: Scalable feature learning for networks," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2016.

[55] L. Wan, Y. Yuan, F. Xia, and H. Liu, "To your surprise: Identifying serendipitous collaborators," *IEEE Trans. Big Data*, to be published, doi: 10.1109/TBDATA.2019.2921567.

[56] D. Rafailidis and F. Crestani, "Recommendation with social relationships via deep learning," in *Proc. Int. Conf. Theory Inf. Retrieval*, 2017, pp. 151–158.

[57] A. Mnih and R. Salakhutdinov, "Probabilistic matrix factorization," in *Proc. Adv. Neural Inf. Process. Syst.*, 2007, pp. 1257–1264.

[58] C. H. Ding, T. Li, 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.

[59] G. E.Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of datawith neural networks," *Science*, vol. 313, pp. 504–507, 2006.

[60] M. Riesenhuber and T. Poggio, "Hierarchicalmodels of object recognition in cortex," *Nat. Neurosci.*, vol. 2, pp. 1019–1025, 1999.

[61] J. Malo, I. Epifanio, R. Navarro, and E. P. Simoncelli, "Nonlinear image representation for efficient perceptual coding," *IEEE Trans. Image Process.*, vol. 15, no. 1, pp. 68–80, Jan. 2006.

[62] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Netw.*, vol. 2, pp. 359–366, 1989.

[63] Y. Nesterov, "Gradient methods for minimizing composite objective function," *Math. Program.*, vol. 140, no. 1, pp. 125–161, 2013.

[64] N. S. Keskar, D. Mudigere, J. Nocedal, M. Smelyanskiy, and P. T. P. Tang, "On large-batch training for deep learning: Generalization gap and sharp minima," Presented at the Int. Conf. Learn. Representations, Toulon, France, 2017.

[65] M. Li, T. Zhang, Y. Chen, and A. Smola, "Efficient mini-batch training for stochastic optimization," in *Proc. 20th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2014, pp. 661–670.

[66] M. Rezaei, R. Boostani, and M. Rezaei, "An efficient initialization method for nonnegative matrix factorization," *J. Appl. Sci.*, vol. 11, no. 2, pp. 354–359, 2011.

[67] F. Xia, A. M. Ahmed, L. T. Yang, and Z. Luo, "Community-based event dissemination with optimal load balancing," *IEEE Trans. Comput.*, vol. 64, no. 7, pp. 1857–1869, Jul. 2015.

[68] S. Ding, Z. Yue, S. Yang, F. Niu, and Y. Zhang, "An efficient initialization method for nonnegative matrix factorization," *IEEE Trans. Knowl. Data Eng.*, vol. 32, no. 11, pp. 2101–2114, Nov. 2020.

[69] H. Zhang, Z. Yang, and E. Oja, "Improving cluster analysis by co-initializations," *Pattern Recognit. Lett.*, vol. 45, no. 1, pp. 71–77, Aug. 2014.

[70] N. Gillis and A. Kumar, "Exact and heuristic algorithms for seminonnegative matrix factorization," *SIAM J. Matrix Anal. Appl.*, vol. 36, no. 4, pp. 1404–1424, 2015.

[71] H. Abdi and L. J. Williams, "Principal component analysis," *Wiley Interdiscipl. Rev., Comput. Statist.*, vol. 2, no. 4, pp. 433–459, Jul./Aug. 2010.

[72] S. T. Roweis and L. K. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *Science*, vol. 290, no. 5500, pp. 2323–2326, 2000.

[73] H. A. Song and S.-Y. Lee, "Hierarchical data representation model - multi-layer NMF," Presented at the Int. Conf. Learn. Representations, Scottsdale, AZ, USA, 2013.
[74] N. Guan, D. Tao, Z. Luo, and B. Yuan, "NeNMF: An optimal gradient method for nonnegative matrix factorization," *IEEE Trans. Signal Process.*, vol. 60, no. 6, pp. 2882–2898, Jun. 2012.

Liangtian Wan (Member, IEEE) received the B.S. and Ph.D. degrees from the College of Information and Communication Engineering, Harbin Engineering University, Harbin, China, in 2011 and 2015, respectively. From October 2015 to April 2017, he has been a Research Fellow with the School of Electrical and Electrical Engineering, Nanyang Technological University, Singapore. He is currently an Associate Professor with the School of Software, Dalian University of Technology, China. He has authored more than 30 papers published in related international conference proceedings and journals. His current research interests include graph learning, data science, and array signal processing.

Feng Xia (Senior Member, IEEE) received the B.Sc. and Ph.D degrees from Zhejiang University, Hangzhou, China. He is currently an Associate Professor and Discipline Leader with the School of Engineering, IT and Physical Sciences, Federation University Australia, Australia. He has authored or coauthored two books and more than 300 scientific papers in international journals and conferences. His research interests include data science, social computing, and systems engineering. He is a Senior Member of ACM.

Xiangjie Kong (Senior Member, IEEE) received the B.Sc. and Ph.D. degrees from Zhejiang University, Hangzhou, China. He is currently a Full Professor with the College of Computer Science and Technology, Zhejiang University of Technology. He has authored or coauthored more than 130 scientific papers in international journals and conferences. His research interests include network science, mobile computing, and computational social science. He is a Senior Member CCF and is a member of ACM.

Ching-Hsien Hsu (Senior Member, IEEE) is the Chair Professor and the Dean of the College of Information and Electrical Engineering, Asia University, Taiwan. His research interests include cloud computing, parallel and distributed systems, big data analytics, artificial intelligence, and smart medical. He was the recipient of seven Talent awards from the Ministry of Science and Technology, Ministry of Education, Taiwan. He is a Fellow of the Institution of Engineering and Technology and the Chair of IEEE Technical Committee on Cloud Computing (TCCLD).

Runhe Huang received the Ph.D degree in computer science and mathematics from the University of the West of England, Bristol, U.K., in 1993. He is currently a Full Professor with the Faculty of Computer and Information Sciences, Hosei University, Japan. Her research interests include multi-agent systems, computational intelligence computing, ubiquitous intelligence computing, and big data.

Jianhua Ma received the B.S. and M.S. degrees from the National University of Defense Technology, Changsha, China, in 1982 and 1985, respectively, and the Ph.D. degree from Xidian University, Xi'an, China, in 1990. He is a Professor with the Faculty of Computer and Information Sciences, Hosei University, Tokyo, Japan. He has authored or coauthored more than 200 papers and edited more than 20 books/proceedings and more than 20 journal special issues. His research interests include multimedia, networks, ubiquitous computing, social computing, and cyber intelligence.