This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2019.2893714, IEEE Transactions on Industrial Informatics

Deep Matrix Factorization with Implicit Feedback Embedding for Recommendation System

Baolin Yi, Xiaoxuan Shen, Hai Liu, *Member, IEEE,* Zhaoli Zhang, *Member, IEEE,* Wei Zhang, Sannyuya Liu, and Naixue Xiong

***Abstract*—Automatic recommendation has become an increasingly relevant problem to industries, which allows users to discover new items that match their tastes and enables the system to target items to the right users. In this article, we propose a deep learning (DL) based collaborative filtering (CF) framework, namely, deep matrix factorization (DMF), which can integrate any kind of side information effectively and handily. In DMF, two feature transforming functions are built to directly generate latent factors of users and items from various input information. As for the implicit feedback that is commonly used as input of recommendation algorithms, implicit feedback embedding (IFE) is proposed. IFE converts the high-dimensional and sparse implicit feedback information into a low-dimensional real-valued vector retaining primary features. Using IFE could reduce the scale of model parameters conspicuously and increase model training efficiency. Experimental results on five public databases indicate that the proposed method performs better than the state-of-the-art DL based recommendation algorithms on both accuracy and training efficiency in terms of quantitative assessments.**

***Index Terms*—Collaborative filtering (CF), deep learning, matrix factorization (MF), recommendation system, representation learning.**

I. Introduction

RECOMMENDATION systems (RS) have become extremely popular in recent years, which has been extensively used in the fields of movies [1], [2], music [3], news [4] , industrial big data [5]–[7] and industrial application [8]–[10] and so on. In the past decades,numerous recommendation algorithms [11]–[13] and platforms [14]–[18] have been developed. Collaborative Filtering (CF) [19] is one of the most distinguished approaches. CF estimates the unknown ratings based on known ones subject to globally high accuracy and other requirements. One of the most popular CF approaches is based on low-dimensional factor models. These models are called matrix factorization (MF) models or latent factor models.

To improve the recommendation effect, many kinds of side information are introduced to enhance recommendation performance. For instance, social recommendation [20]–[22] utilizes social relations or trust relations; content-based recommendation [1], [23] employs the content of items or users such as the text introduction, video content and so on. The most practical approach for recommendation algorithm utilizing side information can be summarized as follows. First, it builds a feature extraction model for side information (social relations, content information), then trains the MF model by using these features as the prior of user latent factors or item latent factors. Model performance could be improved by these reasonable priors. Much of the research focused on exploring a better feature extraction model [24], [25]. Meanwhile other researches tried to introduce many different kinds of side information to push the model performance to the utmost limits [1], [26].

With the considerable advancements in vision, speech and natural language processing tasks, deep learning (DL) has become a very significant research tool in many fields. With the DL algorithm, artificial intelligence has achieved substantial breakthroughs in many areas. In recommendation system, many DL based models have been proposed in the last several years, which can be classified into two groups, the deep learning prior estimate model (DLPE) and the single channel recommendation model (SCR). In the DLPEs, DL method is employed to estimate the prior of latent factors, then the latent factors of users and items are inferred by the observed ratings. The DL method includes the convolutional neural network (CNN) [25], [26] and the stacked denoising autoencoder (SDAE) [24], [27]. Although DLPEs have many advantages over traditional methods, unfortunately, it consumes prodigious time and computing resources in the inference process. As to SCRs, it learns the key patterns from users historical behaviour, then these key patterns are utilized to predict the unknown ratings. SCRs can be summarized as several primary models: restricted Boltzmann machine (RBM) [28], autoencoder (AE) [20], [29]–[31], neural autoregressive distribution estimator (NADE) [32], [33], recurrent neural network (RNN) [34] and so on. Compared with traditional recommendation frameworks, SCRs achieved extraordinary performance. Limited by the particular structure of SCRs, it is a challenge to merge side information of both users and items into the SCRs.

To build a high-efficiency recommendation framework and

This work was supported in part by the National Key R&D Program of China (2017YFB1401300, 2017YFB1401303), the Specific Funding for Education Science Research by Self-determined Research Funds of CCNU (No. CCNU18ZDPY10), and the Cultivating Excellent Doctoral Dissertations Program of CCNU (No. 2018YBZZ006). Paper no. TII-18-2022.R1. (*Corresponding author: Hai Liu.*)

H. Liu is with the National Engineering Research Center for E-Learning, Central China Normal University, Wuhan 430079, China (email: hailiu0204@mail.ccnu.edu.cn).

B. Yi, X. Shen, Z. Zhang and W. Zhang are with the National Engineering Research Center for E-Learning, Central China Normal University, Wuhan 430079, China.

S. Liu is with the National Engineering Research Center for E-Learning, the National Engineering Laboratory for Educational Big Data, Central China Normal University, Wuhan 430079, China.

N. Xiong is with the National Engineering Laboratory for Educational Big Data, Central China Normal University, Wuhan 430079, China.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2019.2893714, IEEE Transactions on Industrial Informatics

merge all kinds of side information into the framework easily, we propose the deep matrix factorization (DMF) model. First, implicit feedback embedding (IFE) is proposed to convert the high-dimensional and sparse implicit feedback information into a low-dimensional real-valued vector retaining primary features. Furthermore, DMF structures two feature transforming functions to produce latent factors from all input information of users and items directly. Finally, DMF can predict interests of users with constructed latent factors. The major contributions of our study are summarized as follows.

- A new recommendation framework DMF is proposed. DMF generates latent factors by the feature transforming function from users and items information directly instead of estimate it from the observed data. Furthermore, DMF adopts two channel structures, which can merge side information from both users and items.
- The idea of implicit feedback embedding (IFE) is developed to represent the implicit feedback for the first time. IFE maps each user and item in implicit feedback information graph to a low-dimensional real-valued vector persisting key patterns. By utilizing IFE, the scale of parameters in DMF is lessened and the training efficiency is raised vastly.
- Empirical study using real-world data has been set. We evaluate the proposed method DMF on five real-world data sets. The result indicates that DMF outperforms the state-of-the-art recommendation algorithms on both prediction accuracy and training efficiency.

The article is organized as follows. In the next section, we introduce the DMF model and representation learning for implicit feedback. The optimization method and parameter determination are presented in Section III. Experimental results on five public databases are provided in Section IV, and Section V concludes this article.

II. Proposed Method

A. Problem Formulation

In CF based recommendation system, historical rating behaviors of users are usually expressed as a user-item rating matrix. Given a user set U and an item set I, user-item rating matrix $\boldsymbol{Z}$ is a $|U| \times |I|$ matrix where each element r_{ij} is proportional to user i's preference on item j, which is shown in Fig. 1; $|U|$ and $|I|$ denote the numbers of user set and item set, respectively. In the user-item rating matrix $\boldsymbol{Z}$, exceedingly few elements have been observed; the observed dataset is named $\boldsymbol{R}$, shown in Fig. 1. Consequently, the problem of CF based RS is constructing an estimator $\hat{r}_{ij}$ to minimize the predictive error in $\boldsymbol{R}$ and generating the prediction for each unobserved element in $\boldsymbol{Z}$.

B. Data explanation

User historical rating behavior is the most indispensable source data in CF based recommendation algorithm. Moreover, it is also called explicit feedback. Relatively, implicit feedback is the most accessible data as it does not need users to express their tastes explicitly. Implicit feedback indicates the

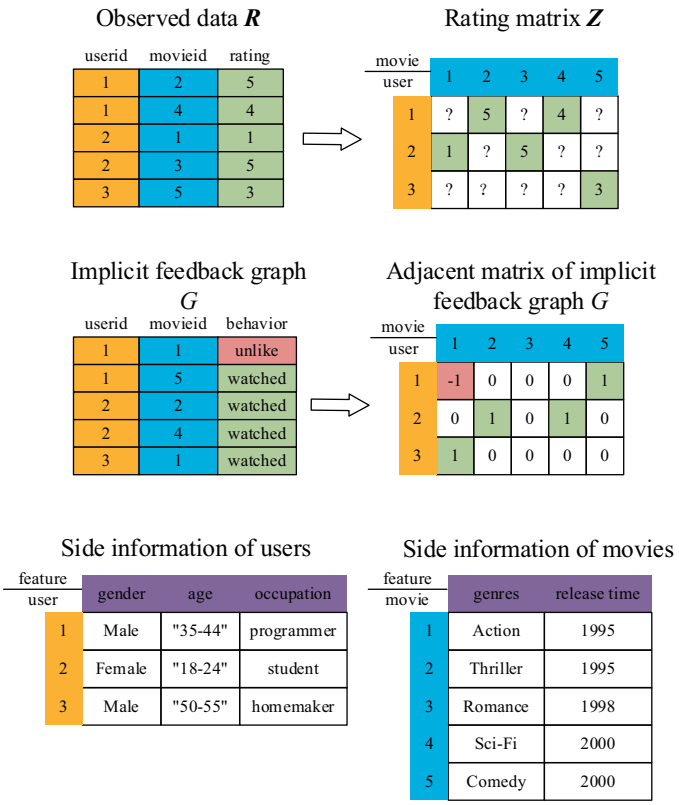


Fig. 1. Toy example of raw data excerpted from Movielens datasets.

users attitude to an item but not directly as rating behaviors. Both positive feedback (e.g., clicks, purchases) and negative feedback (e.g., blacklist, unlike) are considered as implicit feedback as shown in the illustrative example in Fig. 1. Along with the user historical behaviors, a wealth of information is conducive to predict the unobserved rating, such as user profile, item profile, and so on. Such information is called side information shown in Fig. 1.

C. Outline of DMF

Deep matrix factorization can be summed as two processes, namely, data preparation process and the DMF model (Fig. 2). The DMF model can work well with the accuracy of users and items information, which is provided by data preparation process.

In the first process (data preparation process), the implicit feedback is encoded by IFE and side information is encoded by one-hot encoding. Combining these two information, users and items information pools could be generated.

In the DMF model, predicted rating could be estimated by three steps. First, target user and item information are extracted from users and items information pools. Second, latent factors are generated from user and item feature transforming functions. Third, predicted rating could be estimated by latent factors.

D. Implicit Feedback Embedding and Data Preparation

We first define implicit feedback and IFE.

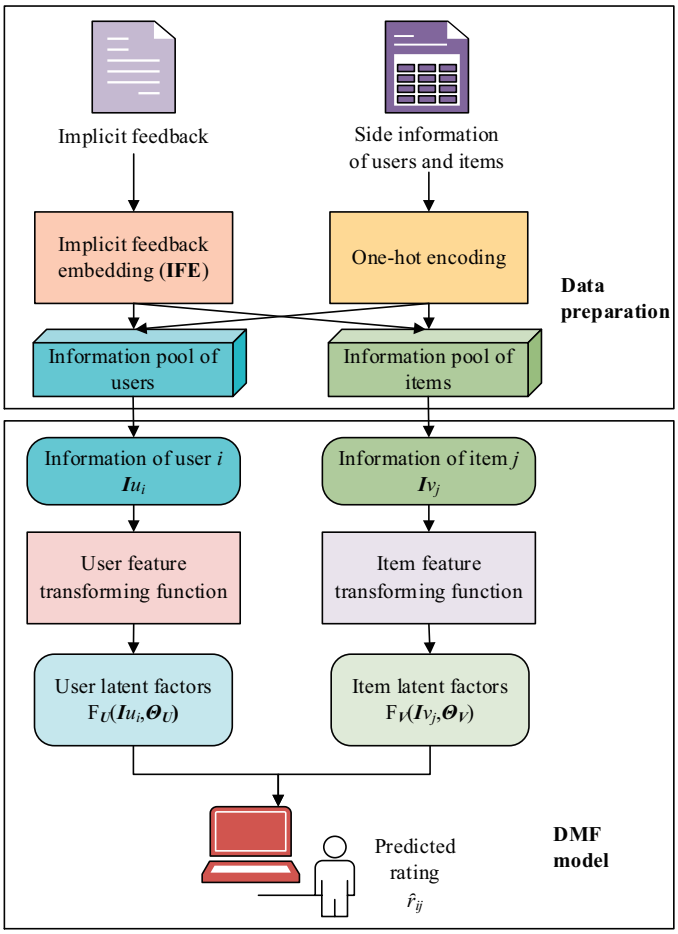


Fig. 2. Outline of DMF framework.

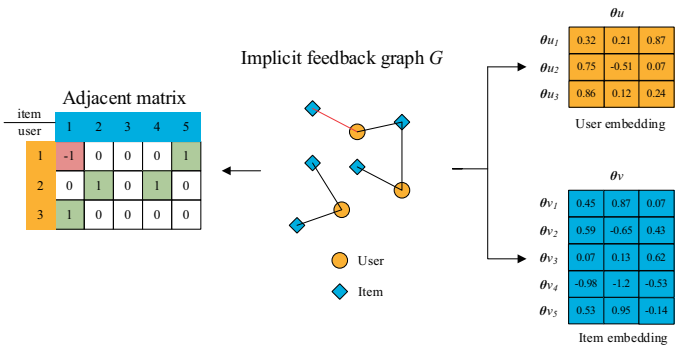


Fig. 3. Illustrative example for implicit feedback and implicit feedback embedding. The value "-1" is used to make explanation which is not exist in the most datasets.

Definition 1: Implicit feedback. Implicit feedback means the historical behaviour between users and items. Implicit feedback could be stated as an undirected graph G= (U, I, E), where U and I denote the user set and item set, respectively. E is the set of edges and E only connects between nodes in U and I which are elucidated in Fig. 3.

In implicit feedback graph, whether edges could be weighted or not depend on the type of historical behaviour. In this article, we consider the implicit feedback graph as an unweighted and undirected graph. Only positive feedback is included in our model and rating behaviour is treated as a positive implicit feedback even if the rating is relatively low.

Definition 2: Implicit feedback embedding. Given an implicit feedback graph G= (U, I, E), IFE aims to learn a function f: u and $v \rightarrow \mathbb{R}^k$, $u \in U$ and $v \in I$ that projects each user and item into a vector in k-dimensional space shown in Fig. 3, where $k \ll |U|$ and $k \ll |I|$.

Usually, implicit feedback information is encoded in its adjacent matrix, which is an extremely high-dimensional and sparse matrix. By employing the IFE method, implicit feedback could be represented as a low-dimensional real-valued vector and preserves the primary features. In addition, it could vastly reduce the scale of model parameters and increase training efficiency.

The vectors $\boldsymbol{\theta}_{ui}$ and $\boldsymbol{\theta}_{vj}$ are introduced to denote the embedding of user i and item j in IFE where $\boldsymbol{\theta}_{ui} \in \mathbb{R}^k$ and $\boldsymbol{\theta}_{vj} \in \mathbb{R}^k$. The symbol k is the dimensionality of user and item embedding. IFE models the probability of incident whether user i has positive feedback about item j which is given by,

$$p\left(E_{ui,vj} \in \boldsymbol{G}\right) = h\left(\boldsymbol{\theta}_{ui} \cdot \boldsymbol{\theta}_{vj}^{\mathrm{T}}\right) = \frac{1}{1+\exp\left(-\boldsymbol{\theta}_{ui} \cdot \boldsymbol{\theta}_{vj}^{\mathrm{T}}\right)} \tag{1}$$

where $\boldsymbol{G}$ is the implicit feedback graph, $E_{ui,vj}$ means the edge between user i and item j.

Negative feedback is an essential part of estimating users and items embedding. Because of the difficulty of negative feedback collecting, negative sampling [35], [36] is introduced to generate negative feedback for IFE. The probability of negative feedback is given as follows

$$p\left(E_{ui,vj} \in \boldsymbol{S}_{neg}\right) = 1 - h\left(\boldsymbol{\theta}_{ui} \cdot \boldsymbol{\theta}_{vj}^{\mathrm{T}}\right) \tag{2}$$

where $\boldsymbol{S}_{neg}$ means the negative feedback graph, which is generated by negative sampling. The probability of the user and item sampled in negative sampling is proportional to its frequency in training sets. The likelihood function for training sets is proposed as

$$p\left(\boldsymbol{G}, \boldsymbol{S}_{neg} | \boldsymbol{\theta}_u, \boldsymbol{\theta}_v\right) = \prod_{E_{ui,vj} \in \boldsymbol{G}} h\left(\boldsymbol{\theta}_{ui} \cdot \boldsymbol{\theta}_{vj}^{\mathrm{T}}\right) \cdot \prod_{E_{ui,vj} \in \boldsymbol{S}_{neg}} 1 - h\left(\boldsymbol{\theta}_{ui} \cdot \boldsymbol{\theta}_{vj}^{\mathrm{T}}\right) \tag{3}$$

where $\boldsymbol{\theta}_u$ and $\boldsymbol{\theta}_v$ indicate the embedding of all users and items, the logarithmic form of (3) is given

$$\mathrm{E}_{\mathrm{IFE}} = \log p\left(\boldsymbol{G}, \boldsymbol{S}_{neg} | \boldsymbol{\theta}_u, \boldsymbol{\theta}_v\right) = \sum_{E_{ui,vj} \in \boldsymbol{G}} \log h\left(\boldsymbol{\theta}_{ui} \cdot \boldsymbol{\theta}_{vj}^{\mathrm{T}}\right) + \sum_{E_{ui,vj} \in \boldsymbol{S}_{neg}} \log\left(1 - h\left(\boldsymbol{\theta}_{ui} \cdot \boldsymbol{\theta}_{vj}^{\mathrm{T}}\right)\right) \tag{4}$$

According to the maximum likelihood estimation rule, embeddings of users and items could be estimated by maximizing $\mathrm{E}_{\mathrm{IFE}}$, which is illustrated in (4). The application considered here is focused on implicit feedback graph containing only positive feedback. For the real-valued or discrete-valued implicit feedback graph, linear regression or softmax regression can be introduced to address the problem in the similar way respectively.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2019.2893714, IEEE Transactions on Industrial Informatics

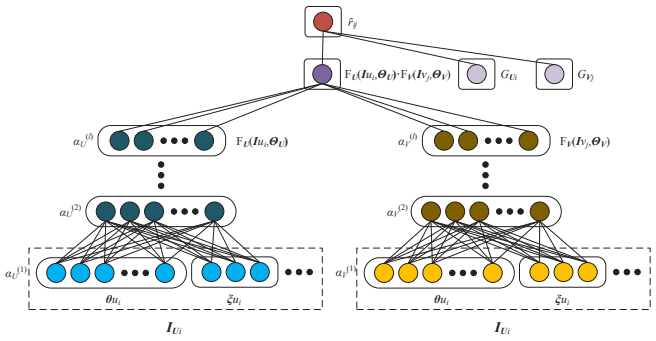


Fig. 4. Architecture of deep matrix factorization.

The side information of users and items can be represented by one-hot encoding easily. Further, $\boldsymbol{I}_{ui}$ and $\boldsymbol{I}_{vj}$, the information of user i and item j, can be built by combining implicit feedback, side information and other additional information. $\boldsymbol{I}_{ui}$ and $\boldsymbol{I}_{vj}$ are produced as follows

$$\begin{cases} \boldsymbol{I}_{ui} = \boldsymbol{\theta}_{ui} \oplus \boldsymbol{\xi}_{ui} \oplus ... \\ \boldsymbol{I}_{vj} = \boldsymbol{\theta}_{vj} \oplus \boldsymbol{\xi}_{vj} \oplus ... \end{cases} \tag{5}$$

where $\oplus$ is the concatenation operator, $\boldsymbol{\theta}_{ui}$ and $\boldsymbol{\theta}_{vj}$ denote the IFE about user i and item j, respectively, $\boldsymbol{\xi}_{ui}$ and $\boldsymbol{\xi}_{vj}$ are the side information of user i and item j. Furthermore, $\boldsymbol{I}_{ui}$ and $\boldsymbol{I}_{vj}$ can be flexible and can be incorporate with any useful information such as content information of items, social information of users and so on. Correspondingly DMF has capability to convey diversified information for recommendation model.

E. Deep Matrix Factorization and Feature Transforming Function

The rating prediction formula of DMF is proposed as

$$\hat{r}_{ij} = \mathrm{F}_u\left(\boldsymbol{I}_{ui}, \boldsymbol{\Theta}_u\right) \cdot \mathrm{F}_v\left(\boldsymbol{I}_{vj}, \boldsymbol{\Theta}_v\right)^{\mathrm{T}} + G_{ui} + G_{vj} \tag{6}$$

where $\mathrm{F}_u\left(\boldsymbol{I}_{ui}, \boldsymbol{\Theta}_u\right)$ and $\mathrm{F}_v\left(\boldsymbol{I}_{vj}, \boldsymbol{\Theta}_v\right)$ are the feature transforming functions for users and items, respectively, $\boldsymbol{I}_{ui}$ and $\boldsymbol{I}_{vj}$ are the input information of user i and item j, $\boldsymbol{\Theta}_u$ and $\boldsymbol{\Theta}_v$ denote the parameters in F_u and F_v. G_{ui} and G_{vj} are introduced to present the global effects of user i and item j.

We assume the observed noise in DMF fit the Gaussian distribution, $\varepsilon \sim \mathcal{N}(0, \delta^2)$. Then, the conditional distribution of r_{ij} can be presented as

$$p\left(r_{ij}|\boldsymbol{\Theta}_u, \boldsymbol{\Theta}_v, G_{ui}, G_{vj}, \delta\right) = \mathcal{N}\left(\hat{r}_{ij}|r_{ij}, \delta^2\right) \tag{7}$$

where $\mathcal{N}(x|\mu, \delta^2)$ is the probability density function of the Gaussian distribution with the mean μ and variance δ^2. The likelihood function of observed ratings over training data can be proposed as

$$p\left(\boldsymbol{R}|\boldsymbol{\Theta}_u, \boldsymbol{\Theta}_v, G_u, G_v, \delta\right) = \prod_{i=1}^{|U|}\prod_{j=1}^{|I|}\left[\mathcal{N}\left(\hat{r}_{ij}|r_{ij}, \delta^2\right)\right]^{1_{ij}} \tag{8}$$

where 1_{ij} is the indicator matrix, in which 1_{ij} is equal to 1 if r_{ij} is observed and equal to 0 otherwise. To simplify the calculation process, we use the log-likelihood function instead of the original likelihood function, which is given by

$$\begin{aligned} \mathcal{L} &= \log p\left(\boldsymbol{R}|\boldsymbol{\Theta}_u, \boldsymbol{\Theta}_v, G_u, G_v, \delta\right) \\ &= -\frac{1}{2\delta^2}\sum_{i=1}^{|U|}\sum_{j=1}^{|I|} 1_{ij}(r_{ij} - \mathrm{F}_u\left(\boldsymbol{I}_{ui}, \boldsymbol{\Theta}_u\right) \cdot \mathrm{F}_v(\boldsymbol{I}_{vj}, \boldsymbol{\Theta}_v)^{\mathrm{T}} \\ &\quad - G_{ui} - G_{vj})^2 - \frac{1}{2}\left(\left(\sum_{i=1}^{|U|}\sum_{j=1}^{|I|} 1(ij)\right)\ln \pi\delta^2\right) + C \end{aligned} \tag{9}$$

thus,

$$\begin{aligned} \mathrm{E}_{\mathrm{DMF}} &= -\frac{1}{2}\sum_{i=1}^{|U|}\sum_{j=1}^{|I|} 1_{ij}(r_{ij} - \mathrm{F}_u\left(\boldsymbol{I}_{ui}, \boldsymbol{\Theta}_u\right) \cdot \mathrm{F}_v(\boldsymbol{I}_{vj}, \boldsymbol{\Theta}_v)^{\mathrm{T}} \\ &\quad - G_{ui} - G_{vj})^2 \end{aligned} \tag{10}$$

where C is a constant that does not depend on the . According to maximum likelihood estimation (MLE), parameters $\boldsymbol{\Theta}_u$, $\boldsymbol{\Theta}_v$, G_{ui} and G_{vj} can be estimated by maximizing (9) which is equivalent to minimizing the sum-of-squared-errors objective function (10).

The multi-layer perceptron (MLP) is introduced as a feature transforming function in DMF, shown in Fig.4. MLP is an effective model to transform the input features into any distribution. Furthermore, the dropout technique [37] is adopted to enhance the generalization ability of MLP in DMF. The functions of $\mathrm{F}_u\left(\boldsymbol{I}_{ui}, \boldsymbol{\Theta}_u\right)$ and $\mathrm{F}_v\left(\boldsymbol{I}_{vj}, \boldsymbol{\Theta}_v\right)$ depend on the hidden structure of MLP. Therefore, we introduce the recurrence equations of $\mathrm{F}_u\left(\boldsymbol{I}_{ui}, \boldsymbol{\Theta}_u\right)$ and $\mathrm{F}_v\left(\boldsymbol{I}_{vj}, \boldsymbol{\Theta}_v\right)$ in this section. The recurrence equations of $\mathrm{F}_u\left(\boldsymbol{I}_{ui}, \boldsymbol{\Theta}_u\right)$ are as follows

$$\begin{cases} \boldsymbol{a}_u^{(1)} = \boldsymbol{I}_{ui} \\ \boldsymbol{m}_u^{\zeta} \sim Bernoulli(\varphi) \\ \tilde{\boldsymbol{a}}_u^{(\zeta)} = \boldsymbol{a}_u^{(\zeta)} \circ \boldsymbol{m}_u^{\zeta} \\ \boldsymbol{a}_u^{(\zeta+1)} = \sigma\left(\tilde{\boldsymbol{a}}_u^{(\zeta)} \cdot \boldsymbol{W}_u^{(\zeta+1)} + \boldsymbol{b}_u^{(\zeta+1)}\right), \zeta \in \{1, 2, .., l-1\} \\ \mathrm{F}_u\left(\boldsymbol{I}_{ui}, \boldsymbol{\Theta}_u\right) = \boldsymbol{a}_u^{(l)} \end{cases} \tag{11}$$

We define F_u as a l-layers MLP, $\boldsymbol{a}_u^{(\zeta)}$ means the activations of ζ-th layer in F_u, we also use $\boldsymbol{a}_u^{(1)}$ to denote the values from the input layer and $\boldsymbol{a}_u^{(l)}$ denotes the output of F_u. $\boldsymbol{b}_u^{(\zeta+1)}$ and $\boldsymbol{b}_u^{(\zeta)}$ are the weight and bias in layer ζ. $\boldsymbol{m}_u^{(\zeta)}$ represents the dropout mask in ζ-th layer and φ is the keep rate in dropout. The symbol $\circ$ means the Hadamard product between two matrices. The parameters in F_u can be summarized as $\boldsymbol{\Theta}_u = \left\{\boldsymbol{W}_u^{(1)}, \boldsymbol{W}_u^{(1)}, ..., \boldsymbol{W}_u^{(l)}, \boldsymbol{b}_u^{(1)}, \boldsymbol{b}_u^{(2)}, ..., \boldsymbol{b}_u^{(l)}\right\}$, $\sigma(\bullet)$ indicates the active function in MLP. In DMF, rectified linear unit (ReLU) [38] is introduced as the active function. Moreover, it can be expressed as ReLU(x)=max(0, x).

In the same manner, the recurrence equations of $\mathrm{F}_v\left(\boldsymbol{I}_{vj}, \boldsymbol{\Theta}_v\right)$ is given by

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2019.2893714, IEEE Transactions on Industrial Informatics

$$\begin{cases} \boldsymbol{a}_v^{(1)} = \boldsymbol{I}_{vj} \\ \boldsymbol{m}_v^{\zeta} \sim Bernoulli(\varphi) \\ \tilde{\boldsymbol{a}}_v^{(\zeta)} = \boldsymbol{a}_v^{(\zeta)} \circ \boldsymbol{m}_v^{\zeta} \\ \boldsymbol{a}_v^{(\zeta+1)} = \sigma\left(\tilde{\boldsymbol{a}}_v^{(\zeta)} \cdot \boldsymbol{W}_v^{(\zeta+1)} + \boldsymbol{b}_v^{(\zeta+1)}\right), \zeta \in \{1, 2, .., l-1\} \\ \mathrm{F}_v\left(\boldsymbol{I}_{vj}, \boldsymbol{\Theta}_v\right) = \boldsymbol{a}_v^{(l)} \end{cases} \tag{12}$$

In DMF, the input $\boldsymbol{I}_{ui}$ and $\boldsymbol{I}_{vj}$ may have different dimensions, but the outputs of $\mathrm{F}_u\left(\boldsymbol{I}_{ui}, \boldsymbol{\Theta}_u\right)$ and $\mathrm{F}_v\left(\boldsymbol{I}_{vj}, \boldsymbol{\Theta}_v\right)$ must have the same dimensions. Accordingly, we set the same hidden structures for both $\mathrm{F}_u\left(\boldsymbol{I}_{ui}, \boldsymbol{\Theta}_u\right)$ and $\mathrm{F}_v\left(\boldsymbol{I}_{vj}, \boldsymbol{\Theta}_v\right)$ in this article.

III. Optimization And Parameter Determination

A. Optimize the IFE

To address the large-scale representation learning problem, we adopt mini-batch gradient ascent (MBGA) algorithm to optimize the objective function in (4). According to the MBGA strategy, we update $\boldsymbol{\theta}_u$, $\boldsymbol{\theta}_v$ respectively by the update rule in (13).

$$\begin{aligned} &\arg\max_{\boldsymbol{\theta}_u, \boldsymbol{\theta}_v} \ \mathrm{E}_{\mathrm{IFE}} \\ \Rightarrow &\begin{cases} \boldsymbol{\theta}_u \leftarrow \boldsymbol{\theta}_u + \alpha \cdot \dfrac{\partial \mathrm{E}_{\mathrm{IFE}}}{\partial \boldsymbol{\theta}_u} \\ \quad = \boldsymbol{\theta}_u + \alpha \cdot \displaystyle\sum_{E_{ui,vj} \in \boldsymbol{G}} \left(1 - h\left(\boldsymbol{\theta}_{ui} \cdot \boldsymbol{\theta}_{vj}^{\mathrm{T}}\right)\right) \cdot \boldsymbol{\theta}_{vj} \\ \qquad + \alpha \cdot \displaystyle\sum_{E_{ui,vj} \in \boldsymbol{S}_{neg}} -h\left(\boldsymbol{\theta}_{ui} \cdot \boldsymbol{\theta}_{vj}^{\mathrm{T}}\right) \cdot \boldsymbol{\theta}_{vj} \\ \boldsymbol{\theta}_v \leftarrow \boldsymbol{\theta}_v + \alpha \cdot \dfrac{\partial \mathrm{E}_{\mathrm{IFE}}}{\partial \boldsymbol{\theta}_v} \\ \quad = \boldsymbol{\theta}_v + \alpha \cdot \displaystyle\sum_{E_{ui,vj} \in \boldsymbol{G}} \left(1 - h\left(\boldsymbol{\theta}_{ui} \cdot \boldsymbol{\theta}_{vj}^{\mathrm{T}}\right)\right) \cdot \boldsymbol{\theta}_{ui} \\ \qquad + \alpha \cdot \displaystyle\sum_{E_{ui,vj} \in \boldsymbol{S}_{neg}} -h\left(\boldsymbol{\theta}_{ui} \cdot \boldsymbol{\theta}_{vj}^{\mathrm{T}}\right) \cdot \boldsymbol{\theta}_{ui} \end{cases} \end{aligned} \tag{13}$$

where α is the learning rate in gradient-based optimization algorithm, $\{ui, vj\}$ means a mini-batch specimen sampled from $\boldsymbol{G}$ or $\boldsymbol{S}_{neg}$. And a simple implement method of negative sampling is presented as follows.

Algorithm 1. Negative sampling

Input: implicit feedback graph $\boldsymbol{G}$
Set: batch size b
1:Sample two edges $E_{ua,va}$ and $E_{ub,vb}$ randomly from implicit feedback graph $\boldsymbol{G}$
2:**while** $|\boldsymbol{S}_{neg}| < b$ **do**:
  if $E_{ua,vb} \notin \boldsymbol{G}$
    add $E_{ua,vb}$ to $\boldsymbol{S}_{neg}$,
  end while
Output: $\boldsymbol{S}_{neg}$

Then, the implicit feedback embedding algorithm can be proposed as,

Algorithm 2. Implicit feedback embedding

Input: implicit feedback graph $\boldsymbol{G}$
Set: batch size b, learning rate α, dimensionality k
1:Initialize $\boldsymbol{\theta}_u$ and $\boldsymbol{\theta}_v$ randomly
2:**while not** $\mathrm{E}_{\mathrm{IFE}}$ is converged **do**:
  sample a mini batch samples from $\boldsymbol{G}$ in size b,
  generate a mini batch samples by Algorithm 1,
  update $\boldsymbol{\theta}_u$ and $\boldsymbol{\theta}_v$ via (13) with the mini batch
 end while
Output: $\boldsymbol{\theta}_u$ and $\boldsymbol{\theta}_v$

B. Optimize the DMF

After solving the IFE, $\boldsymbol{I}_{ui}$ and $\boldsymbol{I}_{vj}$ as the input of DMF can be built by (5). Then, we can optimize the objective function of DMF, in (10), by mini-batch gradient descent (MBGD) algorithm efficiently. The update rule for all parameters in DMF is given by (14),

$$\begin{aligned} &\arg\min_{\boldsymbol{\Theta}_u, \boldsymbol{\Theta}_v, G_u, G_v} \ \mathrm{E}_{\mathrm{DMF}} \\ \Rightarrow &\begin{cases} G_{ui} \leftarrow G_{ui} - \alpha \cdot \dfrac{\partial \mathrm{E}_{\mathrm{DMF}}}{\partial G_{ui}} \\ \quad = G_{ui} - \alpha \cdot \displaystyle\sum_{i=1}^{|U|} 1_{ij}(\hat{r}_{ij} - r_{ij}) \\ G_{vj} \leftarrow G_{vj} - \alpha \cdot \dfrac{\partial \mathrm{E}_{\mathrm{DMF}}}{\partial G_{vj}} \\ \quad = G_{vj} - \alpha \cdot \displaystyle\sum_{j=1}^{|I|} 1_{ij}(\hat{r}_{ij} - r_{ij}) \\ \boldsymbol{\Theta}_u \leftarrow \boldsymbol{\Theta}_u - \alpha \cdot \dfrac{\partial \mathrm{E}_{\mathrm{DMF}}}{\partial \boldsymbol{\Theta}_u} \\ \boldsymbol{\Theta}_v \leftarrow \boldsymbol{\Theta}_v - \alpha \cdot \dfrac{\partial \mathrm{E}_{\mathrm{DMF}}}{\partial \boldsymbol{\Theta}_v} \end{cases} \end{aligned} \tag{14}$$

where α denotes the learning rate in MBGD. The partial derivatives of $\mathrm{E}_{\mathrm{DMF}}$ with respect to $\boldsymbol{\Theta}_u$ and $\boldsymbol{\Theta}_v$ can be computed by the chain rule from the parameters in layer l to layer 1, and the recursion partial derivative equation for $\boldsymbol{\Theta}_u$ is as follows

$$\begin{cases} \dfrac{\partial \mathrm{E}_{\mathrm{DMF}}}{\partial \boldsymbol{b}_u^{(\zeta)}} = \dfrac{\partial \mathrm{E}_{\mathrm{DMF}}}{\partial \boldsymbol{a}_u^{(\zeta)}} \cdot \dfrac{\partial \boldsymbol{a}_u^{(\zeta)}}{\partial \boldsymbol{b}_u^{(\zeta)}} \\ \qquad = \dfrac{\partial \mathrm{E}_{\mathrm{DMF}}}{\partial \boldsymbol{a}_u^{(\zeta)}} \circ \sigma'\left(\tilde{\boldsymbol{a}}_u^{(\zeta-1)} \cdot \boldsymbol{W}_u^{(\zeta)} + \boldsymbol{b}_u^{(\zeta)}\right) \\ \dfrac{\partial \mathrm{E}_{\mathrm{DMF}}}{\partial \boldsymbol{W}_u^{(\zeta)}} = \dfrac{\partial \mathrm{E}_{\mathrm{DMF}}}{\partial \boldsymbol{a}_u^{(\zeta)}} \cdot \dfrac{\partial \boldsymbol{a}_u^{(\zeta)}}{\partial \boldsymbol{W}_u^{(\zeta)}} \\ \qquad = \tilde{\boldsymbol{a}}_u^{(\zeta-1)^{\mathrm{T}}} \cdot \left(\dfrac{\partial \mathrm{E}_{\mathrm{DMF}}}{\partial \boldsymbol{a}_u^{(\zeta)}} \circ \sigma'\left(\tilde{\boldsymbol{a}}_u^{(\zeta-1)} \cdot \boldsymbol{W}_u^{(\zeta)} + \boldsymbol{b}_u^{(\zeta)}\right)\right) \\ \dfrac{\partial \mathrm{E}_{\mathrm{DMF}}}{\partial \boldsymbol{a}_u^{(\zeta-1)}} = \dfrac{\partial \mathrm{E}_{\mathrm{DMF}}}{\partial \boldsymbol{a}_u^{(\zeta)}} \cdot \dfrac{\partial \boldsymbol{a}_u^{(\zeta)}}{\partial \boldsymbol{a}_u^{(\zeta-1)}} \\ \qquad = \dfrac{\partial \mathrm{E}_{\mathrm{DMF}}}{\partial \boldsymbol{a}_u^{(\zeta)}} \circ \sigma'\left(\tilde{\boldsymbol{a}}_u^{(\zeta-1)} \cdot \boldsymbol{W}_u^{(\zeta)} + \boldsymbol{b}_u^{(\zeta)}\right) \cdot \boldsymbol{W}_u^{(\zeta)^{\mathrm{T}}} \circ \boldsymbol{m}_u^{(\zeta-1)} \end{cases} \tag{15}$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2019.2893714, IEEE Transactions on Industrial Informatics

similarly, the recursion partial derivative equation for $\boldsymbol{\Theta}_v$ is given as,

$$\begin{cases} \dfrac{\partial \mathrm{E}_{\mathrm{DMF}}}{\partial \boldsymbol{b}_v^{(\zeta)}} = \dfrac{\partial \mathrm{E}_{\mathrm{DMF}}}{\partial \boldsymbol{a}_v^{(\zeta)}} \cdot \dfrac{\partial \boldsymbol{a}_v^{(\zeta)}}{\partial \boldsymbol{b}_v^{(\zeta)}} \\ \qquad = \dfrac{\partial \mathrm{E}_{\mathrm{DMF}}}{\partial \boldsymbol{a}_v^{(\zeta)}} \circ \sigma'\left(\tilde{\boldsymbol{a}}_v^{(\zeta-1)} \cdot \boldsymbol{W}_v^{(\zeta)} + \boldsymbol{b}_v^{(\zeta)}\right) \\ \dfrac{\partial \mathrm{E}_{\mathrm{DMF}}}{\partial \boldsymbol{W}_v^{(\zeta)}} = \dfrac{\partial \mathrm{E}_{\mathrm{DMF}}}{\partial \boldsymbol{a}_v^{(\zeta)}} \cdot \dfrac{\partial \boldsymbol{a}_v^{(\zeta)}}{\partial \boldsymbol{W}_v^{(\zeta)}} \\ \qquad = \tilde{\boldsymbol{a}}_v^{(\zeta-1)^{\mathrm{T}}} \cdot \left(\dfrac{\partial \mathrm{E}_{\mathrm{DMF}}}{\partial \boldsymbol{a}_v^{(\zeta)}} \circ \sigma'\left(\tilde{\boldsymbol{a}}_v^{(\zeta-1)} \cdot \boldsymbol{W}_v^{(\zeta)} + \boldsymbol{b}_v^{(\zeta)}\right)\right) \\ \dfrac{\partial \mathrm{E}_{\mathrm{DMF}}}{\partial \boldsymbol{a}_v^{(\zeta-1)}} = \dfrac{\partial \mathrm{E}_{\mathrm{DMF}}}{\partial \boldsymbol{a}_v^{(\zeta)}} \cdot \dfrac{\partial \boldsymbol{a}_v^{(\zeta)}}{\partial \boldsymbol{a}_v^{(\zeta-1)}} \\ \qquad = \dfrac{\partial \mathrm{E}_{\mathrm{DMF}}}{\partial \boldsymbol{a}_v^{(\zeta)}} \circ \sigma'\left(\tilde{\boldsymbol{a}}_v^{(\zeta-1)} \cdot \boldsymbol{W}_v^{(\zeta)} + \boldsymbol{b}_v^{(\zeta)}\right) \cdot \boldsymbol{W}_v^{(\zeta)^{\mathrm{T}}} \circ \boldsymbol{m}_v^{(\zeta-1)} \end{cases} \tag{16}$$

where $\circ$ denotes the Hadamard product between two matrices, $\sigma'(\bullet)$ means derived function of the active function in MLP. The partial derivatives with respect to output of F_u and F_v are

$$\begin{cases} \dfrac{\partial \mathrm{E}_{\mathrm{DMF}}}{\partial \boldsymbol{a}_u^{(l)}} = \dfrac{\partial \mathrm{E}_{\mathrm{DMF}}}{\partial \mathrm{F}_u\left(\boldsymbol{I}_{ui}, \boldsymbol{\Theta}_u\right)} \\ \qquad = \displaystyle\sum_{i=1}^{|U|} \sum_{j=1}^{|I|} 1_{ij}(\hat{r}_{ij} - r_{ij}) \cdot \mathrm{F}_v\left(\boldsymbol{I}_{vj}, \boldsymbol{\Theta}_v\right) \\ \dfrac{\partial \mathrm{E}_{\mathrm{DMF}}}{\partial \boldsymbol{a}_v^{(l)}} = \dfrac{\partial \mathrm{E}_{\mathrm{DMF}}}{\partial \mathrm{F}_v\left(\boldsymbol{I}_{vj}, \boldsymbol{\Theta}_v\right)} \\ \qquad = \displaystyle\sum_{i=1}^{|U|} \sum_{j=1}^{|I|} 1_{ij}(\hat{r}_{ij} - r_{ij}) \cdot \mathrm{F}_u\left(\boldsymbol{I}_{ui}, \boldsymbol{\Theta}_u\right) \end{cases} \tag{17}$$

Algorithm 3. Deep Matrix Factorization

Input: implicit feedback set $\boldsymbol{G}$, rating set $\boldsymbol{R}$, side information $\boldsymbol{S}$
Set: batch size b, learning rate α, dimensionality k
1:Initialize $\boldsymbol{\Theta}_u$, $\boldsymbol{\Theta}_v$ and G_u, G_v randomly
2:Compute the embeddings of users and items, θ_u and θ_v, by Algorithm 2
3:Built information pool $\boldsymbol{I}_u$ and $\boldsymbol{I}_v$ with $\boldsymbol{S}$, θ_u and θ_v
4:**while not** $\mathrm{E}_{\mathrm{DMF}}$ is converged **do**:
    sample a mini batch from $\boldsymbol{R}$ in size b,
    update G_{ui} and G_{vj} via (14) with mini batch,
    update $\boldsymbol{\Theta}_u$ and $\boldsymbol{\Theta}_v$ via (14)-(17) with mini batch,
end while
Output: DMF model

C. Parameter determination

In IFE, k denotes the dimensionality of user and item embedding. It controls the capacity of the representation model. The hyper-parameter φ, in DMF, is the parameter in dropout. The value of φ affects the generalization of DMF. The structure and number of neurons in MLP controls the capacity of DMF. The deeper or wider the structure, the bigger the capacity DMF will have [39], [40].

Some approaches have been developed to determine these parameters automatically, such as the discrepancy principle [41], generalized cross-validation [42], and the L-curve method [43]. In this article, the parameters are determined heuristically. We validate the large range of the parameters with the method given in [42]. For the different scale of the size levels of datasets, there are small changes for the optimal parameters. We find that promising performance can be achieved with the parameters $\varphi = 0.5$ and with three layers MLP structure [k+, a×k, 0.5×a×k], where k+ means the dimensionality of input with k-dimensional embedding and one-hot encoding side information, a∈[2,4].

IV. Experiments And Discussion

A. General setting

1) Evaluation Metrics: There are numerous aspects to evaluate a recommendation algorithm, such as the prediction accuracy, coverage, serendipity, and so on. In this article, we mainly consider the error between predictions and the actual ratings, because it can directly demonstrate whether the model is in a position to capture the essential features of training data or not. In our experiments, two popular error metrics are used to measure the prediction accuracy: the mean absolute error (MAE) and the root mean square error (RMSE). The smaller the MAE or RMSE value becomes, the higher the accuracy. MAE and RMSE are defined as

$$RMSE = \sqrt{\frac{1}{|R_{test}|} \sum_{r_{ij} \in R_{test}} \left(r_{ij} - \hat{r}_{ij}\right)^2} \tag{18}$$

and

$$MAE = \frac{1}{|R_{test}|} \sum_{r_{ij} \in R_{test}} |r_{ij} - \hat{r}_{ij}|_{abs} \tag{19}$$

where $|R_{test}|$ denotes the cardinality of the test set, and $|\bullet|_{abs}$ means the absolute value.

2) Tested Models: Five models are included in our experiment as follows.

a) *Mean*: Each rating is predicted by the average of the ratings on the training set.
b) *PMF*: Probabilistic matrix factorization is a baseline matrix factorization model proposed by Salakhutdinov, et al. in [13]. PMF is the most widely used recommendation model.
c) *AutoRec*: AutoRec is an autoencoder-based recommendation framework, designed by Sedhai, et al. in [31]. In this paper, we use I-AutoRec as the test model.
d) *NADE*: Neural autoregressive distribution estimation is submitted by Uria B, et al. in [32]. And NADE is used to address CF problem by Zheng and Tang et al. in [33].
e) *DLTSR*: Deep learning for long-tail web service recommendations is proposed by Bai B, et al. in [29].
f) *ReDa*: Representation learning via dual-autoencoder for recommendation is a dual-autoencoder based recommendation algorithm which is designed by Zhuang F, et al. in [30].
g) *DMF*: Deep matrix factorization is the model proposed in this article. To obtain objective results, we only use rating

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2019.2893714, IEEE Transactions on Industrial Informatics

data to train the model, such as the above mentioned models.

h) *DMF+*: The model proposed in this article. DMF+ adopts side information and extra implicit feedback to enhance the model performance.

3) Datasets: Five datasets are employed in our experiments as listed below

a) *MovieLens-100k* dataset: MovieLens datasets were collected under the GroupLens Research Project at the University of Minnesota. MovieLens-100k contains 100,000 anonymous ratings for 1,682 movies by 943 users. The sparsity on this dataset is 93.70%. It also has side information such as age, gender, occupation for users and release date, and genres for movies.

b) *MovieLens-1M* dataset: MovieLens-1M is also from MovieLens datasets. MovieLens-1M has 1,000,209 ratings from 3,900 users for 6,040 movies. The sparsity on it is 95.75%. MovieLens-1M has the same side information as MovieLens-100k.

c) *Douban-Book* dataset is a subset of Douban-50000 dataset, which is shared by Zhong [44] from Douban. Douban is one of the Chinese online social network site providing reviews and recommendations services for books, movies, and music. Douban-Book has 543,432 anonymous ratings from 9,671 users on 8,330 books. Douban-Books sparsity is 99.32%. It contains 422,783 implicit feedback.

d) *Douban-Movie* dataset is also the subset of Douban-50000. Douban-Movie consists of 2,530,679 ratings and 1,116,269 implicit feedback from 13,363 users for 13,530 movies. The data sparsity is 98.60% in Douban-Movie.

e) *Douban-Music* dataset is the subset of Douban-50000 on the music area. It has 809,000 ratings and 333,673 implicit feedback on 11,073 music marked by 8,334 users. The sparsity of Douban-Music is 99.12%.

Rating scale on all five datasets is [1, 5]. For gaining objective and unbiased results, we have employed the 80%-20% train-test settings and 5-fold cross-validation technique.

B. Experimental Implementation

The involved five models are trained and compared on all five datasets. For PMF, we set λ=0.005 in all datasets, and k=50 in MovieLens-100k, k=200 in the rest of the datasets. As to AutoRec, we choose λ=1 in all datasets, and 200 hidden neurons for MovieLens-100k, 500 hidden neurons for the other datasets. We set the hidden size equal to 200 in MovieLens-100k and 500 for others in NADE. In DLTSR, we set a = 100, λ_n = 1, λ_v=10, λ_w=0.0001, and c_H=0.1 in all datasets as the author recommended, then hidden structure is built as [200, 50, 200] in MovieLens-100k, [500, 200, 500] in the rest of the datasets. For ReDa, α and β are set at 0.5, γ is set as 1, k=50 in all datasets. In DMF, φ is set at 0.5 in all datasets, the hidden structure is [50, 200, 100] in MovieLens-100k dataset, [300, 600, 300] in MovieLens-1M and [200, 400, 200] in three Douban datasets. The first value in the hidden structure represents the representation dimensionality in IFE, namely k. DMF+ has the same parameters setting as DMF. All seven models are optimized by mini-batch gradient descent algorithm with learning rate α=0.02 and batch size 128.

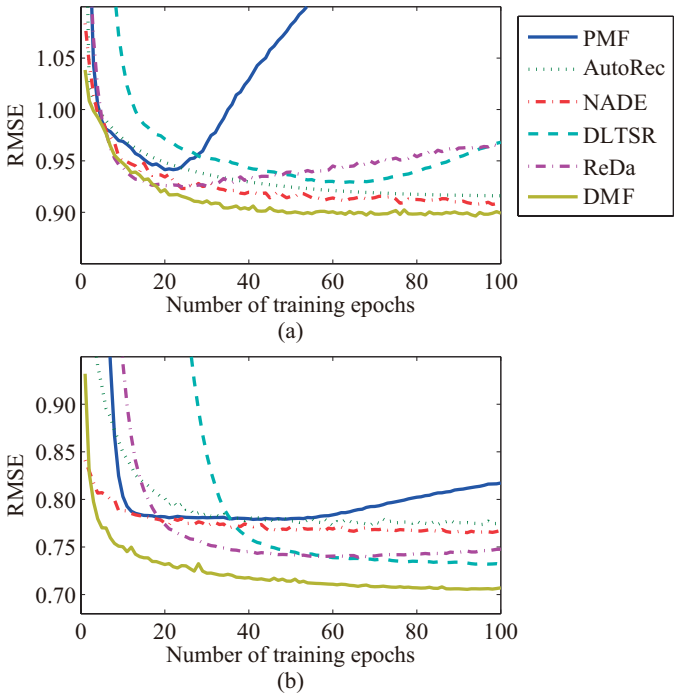


Fig. 5. Training processes of all models compared with RMSE measuring the generalized error on (a) MovieLens-100k and (b) Douban-Book.

The experiment was performed on a PC Server equipped with an Intel(R) Core(TM) i7-7700K CPU@4.20GHz, NVIDIA GeForce GTX 1080 Ti GPU, and 32 GB RAM. We implemented the DMF with software library TensorFlow [45].

C. Results and Discussion

1) Accuracy analysis of DMF: For accuracy comparison of Mean, PMF, AutoRec, NADE, DLTSR, ReDa, and DMF on all five datasets, the lowest RMSE and MAE of each model are summarized in Table I. The consumed time for each training epoch is also included in Tables I. The RMSE at training process is depicted in Fig. 5. In a general manner, DL based methods have a better performance than the traditional methods, as manifested in Table I and Fig. 5. AutoRec, NADE, DLTSR, ReDa, and DMF have prodigious improvement relative to the baseline method PMF and Mean. Compared with PMF, DMF has 5.9%$\sim$9.2% improvement in RMSE and 12.3%$\sim$19.7% in MAE. Moreover DMF achieves the best performance in both MAE and RMSE metrics. During the training process, DMF demonstrates high efficiency which is clarified in Fig. 5. Moreover, DMF+ has better result than DMF as it utilizes side information and extra implicit feedback. Results are improved 0.3%$\sim$0.4% by using side information in MovieLens-100k and MovieLens-1M, 2%$\sim$3% by using extra implicit feedback in three Douban datasets, respectively. It proves that the DMF+ is a high-performance recommendation framework and it has the capability to enhance model effectiveness by extracting the features from additional input information as well.

2) Training efficiency analysis of DMF: AutoRec, NADE, DLTSR, and ReDa introduce user rating behavior as input

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2019.2893714, IEEE Transactions on Industrial Informatics

TABLE I
PERFORMANCE COMPARISON IN TERMS OF RMSE, MAE, CONSUMED TIME FOR EACH EPOCH ON MOVIELENS-100K, MOVIELENS-1M, DOUBAN-BOOK, DOUBAN-MOVIE AND DOUBAN-MUSIC.

MovieLens-100k			
Methods	RMSE	MAE	Consumed time for each epoch (ms)
Mean	1.1537	0.9680	-
PMF [13]	0.9701	0.7823	136
AutoRec [31]	0.9019	0.6771	243
NADE [33]	0.8984	0.6578	563
DLTSR [29]	0.9304	0.7375	2503
ReDa [30]	0.9190	0.7203	2412
DMF	0.8918	0.6568	176
DMF+	**0.8889**	**0.6550**	180

MovieLens-1M			
Methods	RMSE	MAE	Consumed time for each epoch (ms)
Mean	1.1169	0.9335	-
PMF [13]	0.8891	0.6971	1239
AutoRec [31]	0.8401	0.6214	2173
NADE [33]	0.8457	0.6122	4028
DLTSR [29]	0.8637	0.6709	8912
ReDa [30]	0.8485	0.6646	8873
DMF	0.8358	0.6107	1816
DMF+	**0.8321**	**0.6082**	1854

Douban-Book			
Methods	RMSE	MAE	Consumed time for each epoch (ms)
Mean	0.8477	0.6516	-
PMF [13]	0.7791	0.6131	502
AutoRec [31]	0.7832	0.5788	1217
NADE [33]	0.7656	0.5603	2098
DLTSR [29]	0.7304	0.5267	4481
ReDa [30]	0.7390	0.5386	4509
DMF	0.7221	0.5195	923
DMF+	**0.7088**	**0.5070**	929

Douban-Movie			
Methods	RMSE	MAE	Consumed time for each epoch (ms)
Mean	0.9368	0.7570	-
PMF [13]	0.7909	0.6233	1287
AutoRec [31]	0.8036	0.5900	2568
NADE [33]	0.7458	0.5381	4564
DLTSR [29]	0.7327	0.5160	11037
ReDa [30]	0.7362	0.5277	11532
DMF	0.7229	0.5189	2317
DMF+	**0.7105**	**0.5076**	2395

Douban-Music			
Methods	RMSE	MAE	Consumed time for each epoch (ms)
Mean	0.7874	0.6455	-
PMF [13]	0.7192	0.5663	609
AutoRec [31]	0.7466	0.5616	1305
NADE [33]	0.6776	0.4790	2164
DLTSR [29]	0.6609	0.4605	5124
ReDa [30]	0.6699	0.4712	4557
DMF	0.6529	0.4545	980
DMF+	**0.6415**	**0.4361**	993

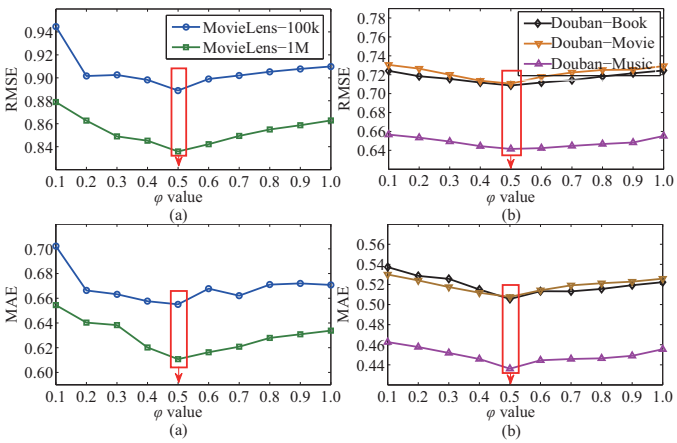


Fig. 6. Performance of DMF+ measured by RMSE and MAE with increasing from 0.1 to 1 on (a) MovieLens-100k, MovieLens-1M; (b) Douban-Book, Douban-Movie, Douban-Music.

information. However, they organize behavior information graph in adjacent matrix, which is high-dimensional and sparse. Therefore, that scale of model parameters of the above mentioned model is extremely large. Unlike these methods, DMF and DMF+ adopt IFE as model input, which reduces the scale of model parameters prominently and increases model training efficiency. The time costs for each epoch with these approaches are given in the last column of Table I. The result proves that DMF has superiority on training efficiency when compared to AutoRec, NADE, DLTSR or ReDa.

3) Sensitivity analysis of the parameter φ: To boost the generalization ability of DMF, the dropout technique is introduced. To analyze the effect of the keep rate named φ, we performed some experiments on all five datasets with the keep rate sampled from 0.1 to 1. We depicted the RMSE and MAE values of different sampled φ on all five datasets in Fig. 6. From Fig. 6, we can observe that dropout model ($\varphi \neq 1$) has achieved better performance compared with none dropout model (φ=1), and we can also find that the result in φ=0.1 is much worse than the result in φ=0.5. Consequently it is essential to choose the value of φ cautiously. In the contrast experiment, the model gets the best result when φ equals to 0.5 on all five datasets. We may conclude that dropout can improve model performance effectively. As to the value of φ, it can be set at 0.5 in most cases.

4) Sensitivity analysis of active function and the structure of MLP: In DL based model, the choice of active function or the structure of MLP effects model performance directly. To reveal the relevance between model performance and the parameters in MLP, we add some comparison experiments. Three widely used active functions: the rectified linear unit (ReLU), the Sigmoid function, and the Tanh function, and four MLP structures are chosen for the comparison experiment on MovieLens-100k and Douban-Book. We drew the RMSE and MAE value of different parameters on histograms which is illustrated in Fig. 7. For the active function, we find in Fig. 7 that ReLU achieves the smallest MAE and RMSE values with all MLP structures. ReLU makes statistically insignificant features to zero, which is analogous to the pruning technique.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2019.2893714, IEEE Transactions on Industrial Informatics

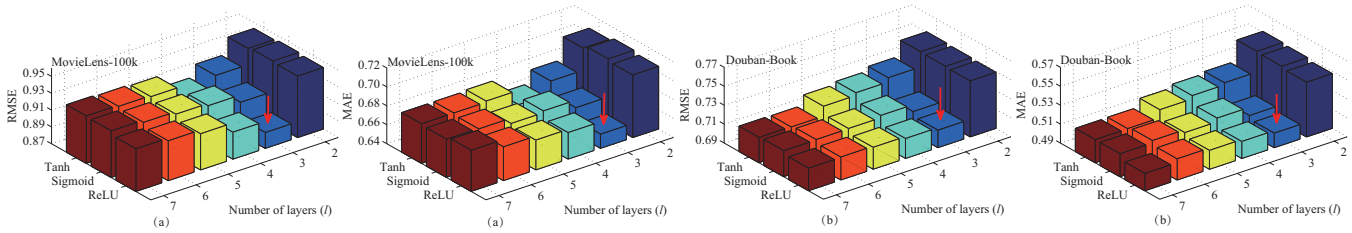


Fig. 7. Performance of DMF+ measured by RMSE and MAE with different active function and MLP structure on (a) MovieLens-100k and (b) Douban-Book.

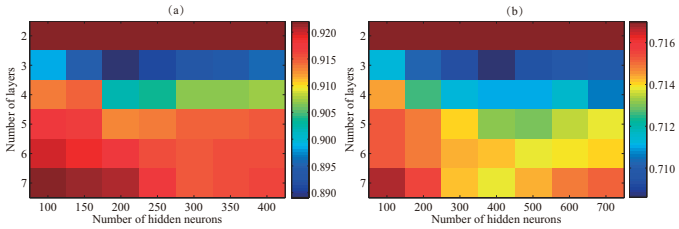


Fig. 8. Performance of DMF+ measured by RMSE and MAE with different number of hidden layers and hidden neurons in MLP structure on (a) MovieLens-100k, and (b) Douban-Book.

The model robustness and generalization capacity are boosted by adopting ReLU as active function. Because of the linear construction of ReLU, it alleviates the gradient vanishing problems of MLP and promotes model training efficiency. To analyze the effect of the MLP structure, we set the same input and output size in the comparison models. The input size is related to the dimensionality of user and item representation k and the extended information, and the output size is set at one or two times of k. In the experiment, we insert different number of layers (depth of MLP) and hidden neurons for each layer (width of MLP). The input size is related to the dimensionality of user and item embedding k and the extended information, and the output size is set at one or two times of k. In the experiment, we insert different number of hidden layers (depth of MLP) and hidden neurons for each layer (width of MLP). The experiment result is presented in Figs. 7 and 8. MLP structures have a direct correlation with model capacity. Further, model capacity should match the scale of training data to avoid the under-fitting or over-fitting problem. The results in Figs. 7 and 8 demonstrate that the model has no hidden neurons (l=2), has less accuracy than the other models, leading to the under-fitting problem distinctly. Thus, it is indispensable to set at least one hidden neurons between the input layer and output layer. Consequently, the result manifests that a three-layer structure achieves the best performance. In our experiments, the model performance is not increased by structuring deeper MLP. The vanishing gradient problem [46] is the major factor behind the phenomenon. Deep MLP structure increases the difficulty of gradient propagating process. It makes the model difficult to be trained smoothly. Therefore, a three-layer MLP model is the best choice to avoid this problem. Based on the above analysis and experimental results, we may conclude that the result is promising when the ReLU pair and three-layer MLP model is chosen.

V. Conclusion

In this article, we have proposed a DL based recommendation framework DMF, which can integrate any kinds of side information handily and efficiently. The main idea of DMF is to build two feature transforming functions to generate users and items latent factors instead of optimizing them directly. DMF architecture is more competent to generate logical latent factors from multiple kinds of features. As to the implicit feedback that is most commonly used in recommendation systems, a representation learning approach IFE is proposed. IFE converts the high-dimensional and sparse implicit feedback information into a low-dimensional real-valued vector and retains primary features. Based on these strategies, the DMF is proposed, which is especially designed for CF problems. Experimental results on five manifold, industrial datasets well demonstrate that DMF can obtain advantage in prediction accuracy even compared to the state-of-the-art models. Furthermore, DMF is proficient to enhance recommendation effect by merging side information. The tuning process of DL based model is time-consuming and tedious. Thus, automatic adjustment of hyper-parameters is an important issue in our future research. Moreover it is necessary to explore new DL models, such as attention model, and embedding model in our future work. Although the application considered here is focused on recommendation systems, DL based MF model could be the general solution for some multisource industrial applications. The next step of our research is to apply the proposed method on more industrial problems. Furthermore, the parallel computing of proposed algorithm is also worth exploring.

References

[1] H. Zhang, Y. Ji, J. Li, and Y. Ye, "A triple wing harmonium model for movie recommendation," *IEEE Transactions on Industrial Informatics*, vol. 12, no. 1, pp. 231–239, 2016.

[2] C. A. Gomez-Uribe and N. Hunt, "The netflix recommender system: Algorithms, business value, and innovation," *Acm Transactions on Management Information Systems*, vol. 6, no. 4, p. 13, 2016.

[3] S. Oramas, V. C. Ostuni, T. D. Noia, X. Serra, and E. D. Sciascio, "Sound and music recommendation with knowledge graphs," *ACM Trans. Intell. Syst. Technol.*, vol. 8, no. 2, pp. 21:1–21:21, 2016.

[4] B. B. Cambazoglu, B. B. Cambazoglu, F. Gullo, F. Silvestri, and F. Silvestri, "Exploiting search history of users for news personalization," *Information Sciences*, vol. 385, pp. 125–137, 2017.

[5] P. Basanta-Val, "An efficient industrial big-data engine," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 4, pp. 1361–1369, 2018.

[6] M. Congosto, P. Basanta-Val, and L. Sanchez-Fernandez, "T-hoarder: A framework to process twitter data streams," *Journal of Network & Computer Applications*, vol. 83, pp. 28–39, 2017.

[7] M. Zaharia, R. S. Xin, P. Wendell, T. Das, M. Armbrust, A. Dave, X. Meng, J. Rosen, S. Venkataraman, and M. J. Franklin, "Apache spark: a unified engine for big data processing," *Communications of the Acm*, vol. 59, no. 11, pp. 56–65, 2016.

[8] S. W. A. T. W. P. J. S.-S. R. L. M. I. J. I. S. Robert Nishihara, Philipp Moritz, "Real-time machine learning: The missing pieces," in *Proceedings of the 16th Workshop on Hot Topics in Operating Systems. ACM*, 2017, pp. 106–110.

[9] X. Kong, F. Xia, J. Wang, A. Rahim, and S. K. Das, "Time-location-relationship combined service recommendation based on taxi trajectory data," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 3, pp. 1202–1212, 2017.

[10] I. Stoica, "Trends and challenges in big data processing," *Proceedings of the Vldb Endowment*, vol. 9, no. 13, pp. 1619–1619, 2016.

[11] G. J. P. P. Rodrigues L R, "Spare parts list recommendations for multiple-component redundant systems using a modified pareto ant colony optimization approach," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 3, pp. 1107–1114, 2018.

[12] C. Volinsky, C. Volinsky, and C. Volinsky, *Matrix Factorization Techniques for Recommender Systems*. IEEE Computer Society Press, 2009.

[13] R. Salakhutdinov and A. Mnih, "Probabilistic matrix factorization," in *International Conference on Neural Information Processing Systems*, 2007, pp. 1257–1264.

[14] Z. Lv, H. Song, P. Basanta-Val, A. Steed, and M. Jo, "Next-generation big data analytics: State of the art, challenges, and future research topics," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 4, pp. 1891–1899, 2017.

[15] P. Basanta-Val, N. Fernandez-Garcia, L. Sanchez-Fernandez, and J. A. Fisteus, "Patterns for distributed real-time stream processing," *IEEE Transactions on Parallel & Distributed Systems*, vol. 28, no. 11, pp. 3243–3257, 2017.

[16] P. Basanta-Val, N. Fernndez-Garcła, A. J. Wellings, and N. C. Audsley, "Improving the predictability of distributed stream processors," *Future Generation Computer Systems*, vol. 52, no. C, pp. 22–36, 2015.

[17] P. Basantaval, N. C. Audsley, A. J. Wellings, I. Gray, and N. Fernandez-garcia, "Architecting time-critical big-data systems," *IEEE Transactions on Big Data*, vol. 2, no. 4, pp. 310–324, 2016.

[18] P. Basanta-Val and L. Snchez-Fernndez, "Big-boe: Fusing spanish official gazette with big data technology," *Big Data*, vol. 6, no. 2, pp. 124–138, 2018.

[19] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Item-based collaborative filtering recommendation algorithms," in *International Conference on World Wide Web*, 2001, pp. 285–295.

[20] S. Deng, L. Huang, G. Xu, X. Wu, and Z. Wu, "On deep learning for trust-aware recommendations in social networks." *IEEE Transactions on Neural Networks & Learning Systems*, vol. 28, no. 5, pp. 1164–1177, 2016.

[21] Z. Zhao, H. Lu, D. Cai, X. He, and Y. Zhuang, "User preference learning for online social recommendation," *IEEE Transactions on Knowledge & Data Engineering*, vol. 28, no. 9, pp. 2522–2534, 2016.

[22] G. Guo, J. Zhang, and N. Yorke-Smith, "A novel recommendation model regularized with user trust and item ratings," *IEEE Transactions on Knowledge & Data Engineering*, vol. 28, no. 7, pp. 1607–1620, 2016.

[23] E. Aslanian, M. Radmanesh, and M. Jalili, "Hybrid recommender systems based on content feature relationship," *IEEE Transactions on Industrial Informatics*, vol. DOI:10.1109/TII.2016.2631138, 2018.

[24] X. Li and J. She, "Collaborative variational autoencoder for recommender systems," in *The ACM SIGKDD International Conference*, 2017, pp. 305–314.

[25] D. Kim, C. Park, J. Oh, and H. Yu, "Deep hybrid recommender systems via exploiting document context and statistics of items," *Information Sciences*, vol. 417, pp. 72–87, 2017.

[26] F. Zhang, N. J. Yuan, D. Lian, X. Xie, and W. Y. Ma, "Collaborative knowledge base embedding for recommender systems," in *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016, pp. 353–362.

[27] H. Wang and D. Y. Yeung, "Towards bayesian deep learning: A framework and some existing methods," *IEEE Transactions on Knowledge & Data Engineering*, vol. 28, no. 12, pp. 3395–3408, 2016.

[28] K. Georgiev and P. Nakov, "A non-iid framework for collaborative filtering with restricted boltzmann machines," in *International Conference on International Conference on Machine Learning*, 2013, pp. 1148–1156.

[29] B. Bai, Y. Fan, W. Tan, and J. Zhang, "Dltsr: A deep learning framework for recommendation of long-tail web services," *IEEE Transactions on Services Computing*, vol. DOI:10.1109/TSC.2017.2681666, 2018.

[30] F. Zhuang, Z. Zhang, M. Qian, C. Shi, X. Xie, and Q. He, "Representation learning via dual-autoencoder for recommendation," *Neural Networks*, vol. 90, pp. 83–89, 2017.

[31] S. Sedhain, A. K. Menon, S. Sanner, and L. Xie, "Autorec:autoencoders meet collaborative filtering," in *International Conference on World Wide Web*, 2015, pp. 111–112.

[32] B. Uria, Marc-Alexandre, K. Gregor, I. Murray, and H. Larochelle, "Neural autoregressive distribution estimation," *Journal of Machine Learning Research*, vol. 17, no. 1, pp. 7184–7220, 2016.

[33] Y. Zheng, B. Tang, W. Ding, and H. Zhou, "A neural autoregressive approach to collaborative filtering," in *International Conference on Machine Learning*, 2016, pp. 764–773.

[34] M. Quadrana, D. Tikk, and D. Tikk, "Parallel recurrent neural network architectures for feature-rich session-based recommendations," in *ACM Conference on Recommender Systems*, 2016, pp. 241–248.

[35] M. U. Gutmann and H. A., "Noise-contrastive estimation of unnormalized statistical models, with applications to natural image statistics," *Journal of Machine Learning Research*, vol. 13, pp. 307–361, 2012.

[36] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

[37] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[38] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *International Conference on International Conference on Machine Learning*, 2010, pp. 807–814.

[39] Hornik and Kurt, "Approximation capabilities of multilayer feedforward networks," *Neural Networks*, vol. 4, no. 2, pp. 251–257, 1991.

[40] M. Raghu, B. Poole, J. Kleinberg, S. Ganguli, and J. Sohl-Dickstein, "On the Expressive Power of Deep Neural Networks," *ArXiv e-prints*, Jun. 2016.

[41] H. W. Engl, "Discrepancy principles for tikhonov regularization of ill-posed problems leading to optimal convergence rates," *Journal of Optimization Theory & Applications*, vol. 52, no. 2, pp. 209–215, 1987.

[42] G. Golub, MichaelHeath, and GraceWahba, "Generalized cross-validation as a method for choosing a good ridge parameter," *Technometrics*, vol. 21, no. 2, pp. 215–223, 1979.

[43] P. C. Hansen, "Analysis of discrete ill-posed problems by means of the l-curve," *Siam Review*, vol. 34, no. 4, pp. 561–580, 1992.

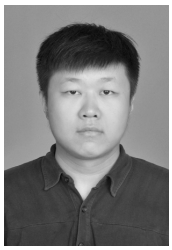
[44] E. Zhong, Y. Li, Y. Li, Y. Li, and Y. Li, "Comsoc: adaptive transfer of user behaviors over composite social network," in *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2012, pp. 696–704.

[45] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, and M. Isard, "Tensorflow: a system for large-scale machine learning," in *OSDI*, vol. 16, 2016, pp. 265–283.

[46] S. Hochreiter, "The vanishing gradient problem during learning recurrent neural nets and problem solutions," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 06, no. 02, pp. 107–116, 1998.

Baolin Yi received the BE and MS degree in mathematics from the Wuhan University, China, in 1992 and 1997, respectively, and the Ph.D. degree in computer science from the Huazhong University of Science and Technology in 2003. He is professor and Ph.D. supervisor of National Engineering Research Center for E-Learning in Central China Normal University since 2010. He is awarded as the expert in the field of Education Information by the Department of Education of Hubei province. His research interest includes database and data mining, education information technology, education cloud computing and education big data analysis.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2019.2893714, IEEE Transactions on Industrial Informatics



Xiaoxuan Shen is currently a Ph.D. candidate in the National Engineering Research Center for E-Learning, Central China Normal University. His research interests include deep learning, representation learning, and their applications in recommendation system, intelligent network learning environment and education big data analysis.

Hai Liu (S'12-M'14) received the M.S. degree in applied mathematics from Huazhong University of Science and Technology (HUST), Wuhan, China, in 2010, and the Ph.D. degree in pattern recognition and artificial intelligence from the same university, in 2014.

Since June 2017, he has been an Assistant Professor with the National Engineering Research Center for E-Learning, Central China Normal University, Wuhan. Currently, he is a "Hong Kong Scholar" postdoctoral fellow with the Department of Mechanical Engineering, City University of Hong Kong, Kowloon, Hong Kong, where he is hosted by the Porfessor Youfu Li; he will hold the position till March 2019. He has authored more than 30 peer-reviewed articles in international journals from multiple domains such as pattern recognition, image processing. His current research interests include big data processing, artificial intelligence, recommendation system, deep learning, signal processing and pattern recognition.

Dr. Liu has been frequently serving as a reviewer for more than six international journals including the *IEEE/ASME Transactions on Mechatronics*, *IEEE Translations on Industrial Informatics*, *IEEE Translations on Cybernetics*, *IEEE Translations on Instrumentation and Measurement*, *Digital Signal Processing*, *Measurement Science & Technology*, and *Applied Optics*. He is also a Communication Evaluation Expert for the National Natural Science Foundation of China.

Zhaoli Zhang (M'18) received the M.S. degree in Computer Science from Central China Normal University, Wuhan, China, in 2004, and the Ph.D. degree in Computer Science from Huazhong University of Science and Technology in 2008. He is currently a professor in the National Engineering Research Center for E-Learning, Central China Normal University. His research interests include signal processing, knowledge services and software engineering. He is a member of IEEE and CCF (China Computer Federation).

Wei Zhang is currently an associate professor in the National Engineering Research Center for E-Learning (http://nercel.ccnu.edu.cn/) and National Engineering Laboratory for Educational Big Data at the Central China Normal University. He holds a Ph.D. degree from Huazhong University of Science and Technology. His research interests include computer applications, big data analysis, data mining, and application of information technology in education. He published more than 40 papers in the academic journals, including 20 papers indexed by SSCI, SCI, EI, ISTP.

Sannyuya Liu received the B.E. and M.E. degrees in 1996 and 1999, and received the Ph.D. degree in 2003 from (HUST). He devoted himself to his postdoctoral research in Xiamen University from 2003 to 2005, and worked for the field of enterprise information, business intelligence, and distributed computing. Currently, he is a professor in NERCEL, CCNU. His research interests include artificial intelligence, computer application, and educational data mining.

Naixue Xiong received the Ph.D. degrees in sensor system engineering and in dependable sensor networks from Wuhan University and the Japan Advanced Institute of Science and Technology, respectively. Before he attended Tianjin University, he was with Northeastern State University, Georgia State University, the Wentworth Technology Institution, and Colorado Technical University (Full Professor for about five years) for about 10 years. He is currently a Professor with the College of Intelligence and Computing, Tianjin University, China. His research interests include cloud computing, security and dependability, parallel and distributed computing, networks, and optimization theory.