Joint Matrix Factorization: A Novel Approach for Recommender System

SHAOLUN SUN\textsuperscript{1,2}, YUETONG XIAO\textsuperscript{3}, YUE HUANG\textsuperscript{1,2}, SEN ZHANG\textsuperscript{1,2}, HENG ZHENG\textsuperscript{1,2}, WENDONG XIAO\textsuperscript{1,2}, (Senior Member, IEEE), AND XIAOLI SU\textsuperscript{1,2}

\textsuperscript{1}School of Automation and Electrical Engineering, University of Science and Technology at Beijing, Beijing 100083, China
\textsuperscript{2}Key Laboratory of Knowledge Automation for Industrial Processes of Ministry of Education, School of Automation and Electrical Engineering, University of Science and Technology at Beijing, Beijing 100083, China
\textsuperscript{3}School of Geophysics and Information Technology, China University of Geosciences at Beijing, Beijing 100083, China

Corresponding author: Sen Zhang (zhangsen@ustb.edu.cn)

This work was supported in part by the National Key Research and Development Program of China under Grant 2017YFB1401203, in part by the Beijing Natural Science Foundation under Grant 4182039, and in part by the National Natural Science Foundation of China under Grant 61673056 and Grant 61673055.

\textbf{ABSTRACT} Collaborative filtering (CF) is the most classical method for recommender system, but it is usually suffered from limited performance by the sparseness of user-to-item rating data. Recently, due to the powerful learning feature representation ability, deep learning components are used to leverage auxiliary information to assist recommendation. However, most existing models based on deep learning are incomplete so that merely extracting the item latent representation and ignoring the user parts. Besides, different data are not chosen from current models. This paper proposes a novel probability framework, named as joint matrix factorization (JMF). There are three components in JMF. Firstly, the modified multilayer crossing version of the factorization machine (MFM) is designed to extract the user latent factors based on user behavior information. Moreover, MFM is a general method which can be used to accomplish many tasks in terms of machine learning. Secondly, a modification of Long Short-Term Memory (LSTM), named as bidirectional LSTM (BLSTM), is used to extract the item latent factors of a document sequence from both front and back directions. Finally, we tightly integrate BLSTM and MFM into probabilistic matrix factorization (PMF) to form JMF. Compared with the classical matrix factorization and other integration models, JMF extracts document data as well as user behavioral data as item vectors and user vectors. Extensive experiments on five real-world datasets show the proposed model has better performance compared with the state-of-the-art recommendation methods.

\textbf{INDEX TERMS} Bidirectional Long Short-Term Memory, deep learning, factorization machine, auxiliary information, recommender system.

\section{I. INTRODUCTION}

With the rapid development of the internet, we are surrounded by all kinds of information. In order to help people, make effective decisions in the case of information overload, there are two main options: Search Engine and Recommender System. However, previous search engines merely returned information with high similarity according to input query. Current search engines also incorporate the idea of recommendation, i.e., combining user’s various preferences for personalization search. By analyzing the user’s preferences and the item’s attributes, the recommender system can actively offer personalized suggestions for users. The classical recommendation algorithm [1] is roughly divided into three categories: content-based method, collaborative filtering-based method (CF) [2], [3] and hybrid method. Among them, CF has been widely used and includes neighborhood-based CF and model-based CF. Matrix factorization (MF), as a representative of model-based CF, represents users and items by automatically learning latent factors. MF has attained outstanding performances and derived multiple versions in different recommendation tasks [3]–[5]. However, it is tough for MF to explicitly incorporate auxiliary information. To solve this problem, some neural network models or special components, such as factorization machine (FM) are used to integrate auxiliary information into MF, where auxiliary

\textsuperscript{224596} This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/
The contributions are summarized as follows:

- A novel multi-layer crossing version of the FM, named as MFM is proposed, which integrates intra-field and extra-field feature interaction based on multi-layer structure. Therefore, MFM can extract user latent vectors from auxiliary information and capture the similarity between users effectively.

- We regard the document as a sequence to model the item representation by BLSTM. Knowledge that may be missed by the one-way circulating neural network can be captured in both directions. Therefore, BLSTM effectively filters out all the important information in the whole sequence.

- In order to make full use of the rating and item document data, this paper tightly integrates BLSTM and MFM into PMF from the perspective of probability i.e., JMF. Combining alternating least squares (ALS) [28] and coordinate descent (CD) can jointly train the three components

- Extensive experiments on five real-world datasets show that JMF significantly outperforms baseline methods for rating prediction accuracy.

II. RELATED WORK

A. NEURAL NETWORK MODEL

The neural network (NN) model transforms the recommendation problem into a regression or classification problem. A hybrid structure is introduced by xDeepFM [12] as an improved version of Wide & deep [13] and deepFM [14]. At the vector level, the compressed interaction network (CIN) intersects the explicit features, while at the bit level, the plain DNNs part learns complex and selective feature interactions for implicit features. Recently, graph neural networks have been widely used in recommendation systems, the multi-task multi-view graph representation learning framework (M2GRL) learns with multi-view data through Combining graph representation, it constructs one graph for each single-view data, learns multiple separate representations from multiple graphs, and performs alignment to model cross-view relations. However, the NN model which only relies on neural networks are limited without the help of traditional recommendation models. Firstly, pretraining is usually require by complex network models to achieve the desired results, and the pre-training [15] of these models costs much time in most of the occasions. Secondly, The NN model excessively relies on the automatic learning feature ability. It is easy to fall into local minimum, which leads to the over-fitting about this pure network model. For example, NeuMF [16], which is initialized by the pre-trained models of GMF and MLP can achieve better performance. But more iterations may make NeuMF have overfitting problem. To avoid overfitting, flexible sampling ratio must be employed for negative instances. Both pre-training and flexible sampling waste lots of time.

B. INTERGRATION MODEL

Because of the above problems for NN, a task-specific component should be integrated to provide guidance for NN. According to [17], integration model is a good choice for harnessing the perception ability from NN and the inference ability (causal and logical) from traditional recommendation methods. Specifically, stacked denoising autoencoder (SDAE) is used by collaborative deep learning (CDL) [6], [18] to extract the comment information as the item vector and integrates the item vector into the PMF [3] in a tightly coupled way. However, SDAE is based on the bag of words model whose semantic content of the text is ignored. Hence, many researches adopt convolutional neural network (CNNs)
to extract meaningful features for modeling items. Kim et al. proposed convMF [7] that uses CNN to capture critical information similar to n-gram through different filters. In the same way DeepCoNN [19] also uses CNN to learn latent factors, then feeds them into factorization machine (FM) [20]. Nevertheless, standard CNNs are only capable of extracting local and position-invariant features [21], which ignores the distant semantic. To alleviate this problem, some improved CNNs such as D-Attn [22] and SentiRec [23] have been proposed. Dual local and global attention mechanism were adopted by D-Attn on the CNN. What makes it more interesting is that SentiRec uses explicit feedbacks (ratings) as overall sentiment of reviews to guide CNN learning latent representations. However, these CNNs are so sensitive to hyperparameters which costs a large amount of time to find the optimal hyperparameters. Besides, incompleteness is the most obvious flaw of the existing integration model. The user is not effectively represented.

III. PRELIMINARY KNOWLEDGE

Before introducing the proposed multi-layer crossing FM (MFM) and BLSTM, it is necessary to briefly review the two original components: FM and LSTM.

A. FACTORIZATION MACHINE

Each feature is independent in terms of traditional linear regression (LR). Combination features can represent non-linear relationships. In order to craft the features automatically, we extend regression (LR), Combination features can represent non-linear factors, then feeds them into factorization machine (FM) [20]. FM intersects all features, resulting in inefficiency. Besides, factor dimension size. But one of major downside is that FM introduces some useless feature interactions may introduce noises and degrade the performance.

B. LONG SHORT-TERM MEMORY

LSTM [29] is a special category of recurrent neural network that settles the vanishing or exploding gradient and long-term dependence issues. LSTM contains a set of memory blocks, which include one or more core memory cells. Moreover, LSTM contains three control units: the input, output and forget gates. LSTM will maintain useful memory information in training process because the three control gates can keep the state of cells. Recently, LSTM has performed well in tasks such as text generation, sentiment analysis and answer systems.

IV. THE REDESIGNED COMPONENTS

In this section, the FM is redesigned and an improved version of LSTM is introduced. They are named as MFM and BLSTM, respectively. Firstly, we give the user representation component MFM, describing its principle and structure detailly. Then we introduce the items representation component BLSTM and explain how the model is applied to document processing.

A. MULTI-LAYER CROSSING FM

FM are mainly used in the case of sparse data. However, from the discussion in Section 2.1, the FM have the disadvantages of low efficiency and is easy to introduce noise. Therefore, this paper modifies the FM heavily, term as multi-layer crossing FM (shortly, MFM). Fig. 1 shows the structure of MFM. Supposing that Field 1 contains all the features, the unit in dashed-red rectangle is the classical FM.

The input to the MFM is one-hot encoding vector of category feature in this paper. Fig. 1 shows that MFM has two feature interaction layers. In the first layer, we split each field into multiple sub-fields, and there is only one active unit in each sub-field. Every sub-field has a probability output unit that acts as an intra-field crossover result. The first layer is divided into two parts: linear and interaction part. The linear part can be written as

$$\hat{y}_L = b_L + W_L X = b_L + \sum_{f=1}^{F} w_f x_f$$  (3)

where $X = [x_1, x_2, x_3, \ldots, x_F]^T$, $x_F$ is an input vector of the $F-th$ sub-field. $W_L = [w_1^T, w_2^T, w_3^T, \ldots, w_F^T]$ and $b_L \in R$ are weight vector and bias. For intersecting part, multiple sub-fields conduct intra-field crossing respectively, while $\hat{y}'$ is the result of the $f-th$ sub-field in first layer.

$$\hat{y}' = \text{sigmoid} \left( \sum_{i=1}^{n} \sum_{j=i+1}^{n} \sum_{f=1}^{k} V_{ij}^f x_i^f x_j^f \right)$$  (4)

where sigmoid ($x$) = $1 / (1 + e^{-x})$ is an activation effect, $V_{ij}^f \in R^{n \times k}$ is the $f-th$ field crossing weight matrix. $n$ is the number of features in the $f-th$ field, and $k$ is the embedding dimension. The linear and the crossing part can be combined to get a vector $P = [\hat{y}_L, \hat{y}_1', \hat{y}_2', \hat{y}_3', \ldots, \hat{y}_F']$, now we use $P$ as the input of the second layer for extra field crossing. Noting that pairwise interactions of FM can
FIGURE 1. The structure of the proposed MFM, where the unit in dashed-red rectangle is the simple classical FM.

be reformulated as (5)

\[
\sum_{i=1}^{n} \sum_{j=i+1}^{n} V V_{ij} x_i x_j = \frac{1}{2} \sum_{k=1}^{K} \left( \left( \sum_{i=1}^{n} v_{i,k} x_i \right)^2 - \sum_{i=1}^{n} v_{i,k}^2 x_i^2 \right)
\]

where \(VV^2\) is the second layer crossing weight matrix. Obviously, the result of the outside bracket is a \(K\)-dimensional vector, which is the user latent vector that we defined. Supposing \(u\) is used to represent the user latent vector \((u = (\sum_{i=1}^{n} v_{i,k} x_i)^2 - \sum_{i=1}^{n} v_{i,k}^2 x_i^2, u R^k)\). Essentially, this vector is correspondingly multiplied by the embedding vectors.

The MFM shown in Fig. 1 only has two interaction layers. By replicating the first layer, the multi-layer crossing FM (MFM) can be obtained easily. Unlike classical FM’s full feature interaction, inter-field and extra-field feature interaction can be performed respectively by MFM based on multi-layer structure. So, this method reduces noise and improve computational efficiency, which is similar to other FMs. MFM can also be trained by using backpropagation method.

Supposing a recommender system similar to YouTube recommendation [30], with browsing history, search tags, ratings, and other types of features. MFM performs intra-field feature interaction based on each type of feature separately. Then, in order to interact different sub-field features, the results of first layer are used to extra-field feature interaction. Globally useful feature interaction will be preserved with the backpropagation optimization algorithm, many useless feature interactions can be reduced, and computational efficiency can be improved. Therefore, MFM will obtain better recommendation quality than traditional FM.

So, In this paper, the MFM is mainly used as a component to extract user representations. Furthermore, MFM can also accomplish recommendation independently. Extra experiments are added to evaluate MFM.

B. BIDIRECTIONAL LONG SHORT-TERM MEMORY

Key information can be effectively extracted by LSTM from overall sequence. Whereas LSTM cannot encode information from back to front. Assuming the data is document, the one-way LSTM may ignore word interaction information about the opposite direction. Fig. 2 shows our BLSTM architecture that consists of three layers: embedding layer, recurrent layer, and concatenation layer.

FIGURE 2. The architecture of the proposed BLSTM. The BLSTM contains two function units of LSTMs to extract the document features from two directions.

1) Embedding Layer

The embedding layer converts a raw document into a dense numeric matrix. In addition, words are represented by dense vectors. Essentially, the embedding layer is a weight matrix that can be learned along with a task.

When it comes to document information, to improve computational efficiency and reduce noise, this paper extracts words which contains more information. In NLP, there are several word filter methods, such as frequency and tf-idf [31].

2) Recurrent Layer

The objective of BLSTM is to study item latent vectors from documents of item. BLSTM is simply regarded as being
synthesized by two LSTMs. The left LSTM unit extracts key information forward and the state of the cell unit is represented as

$$C_t = \sigma(W_t[h_{t-1}, x_t] + b_t) \tanh(W_c[h_{t-1}, x_t] + b_c) + C_{s-1} \sigma(W_f[h_{t-1}, x_t] + b_f)$$

(6)

where $\sigma$ is sigmoid function, $W_f$, $b_f$, $W_i$, $b_i$, $W_c$, $b_c$ are the forget gate, input gate, input weight matrices and bias vectors, $x_t$ is the current input vector, the output of the previous cycle is $h_{s-1}$. From (6), it is found that the cell state is obtained from the previous state and the current input under controlling of two gates. The current output vector is obtained according to the cell state

$$h_t = \sigma(W_h[h_{t-1}, x_t] + b_h) \tanh(C_s)$$

(7)

where $W_h$, $b_h$ are the output gate weight matrix and bias. Similarly, the right LSTM unit extracts information from reverse direction, resulting in an output $h'_s$.

3) Concatenation layer

As for the concatenation layer, it combines the vectors generated by the two LSTMs in the previous layer. The output of BLSTM is $[h_s, h'_s]$. Finally, a full connection layer is used and the output item vectors is $v = W_F[h_s, h'_s]$, $v \in \mathbb{R}^K$ where $W_F$ is the weight matrix of full connection layer. For simplicity, $W_v$ is used to represent all weight matrices and biases in the left LSTM and $W'_v$ is used to represent all weight matrices and biases in the right LSTM.

V. MAIN RESULT: JOINT MATRIX FACTORIZATION

In this section, the probabilistic model of JMF is introduced in detail. First and foremost, this paper provides information on how to integrate MFM and BLSTM into the PMF from a probabilistic perspective. Then this paper explains how to train the proposed JMF and analyze the time complexity of JMF.

A. PROBABILISTIC MODEL OF JMF

Supposing there are $N$ users, $M$ items, so the rating matrix is $R \in \mathbb{R}^{N \times M}$. We aim to generate user and item latent vectors $(U^T \in \mathbb{R}^{N \times K}$ and $V \in \mathbb{R}^{M \times K})$ with the redesigned components, which can reconstruct the rating matrix. Fig. 3 shows the overall structure of JMF. The hypothesis space of the model is attained from the perspective of probability:

- All the weight matrices and bias vectors of MFM are subject to a Gaussian distribution.
- The weight matrices and bias vectors of BLSTM memory block are subject to a Gaussian distribution.
- The observed and the component integration noise (the error between observation rating matrix $R$ and the approximate rating matrix $\tilde{R}$) are subject to the Gaussian distribution.

As for the MFM and BLSTM, they are used as components. And the generative process of JMF is defined as follows:

- Extract user latent vectors with MFM,

1) For the first layer linear part of the weight $W_l$ and the bias $b_l$, draw $W_l \sim \mathcal{N}(0, \sigma^2_{\omega_W})$ and $b_l \sim \mathcal{N}(0, \sigma^2_{\omega_b})$.
2) For the $f$-th field of first layer crossing part of the weight matrix $VV^1_f$, for each column $n$, draw $VV^1_{n,f} \sim \mathcal{N}(0, \sigma^2_{\omega_{V^1_f}})$.
3) For the second layer of the weight matrix $VV^2$, for each column $n$, draw $VV^2_{n} \sim \mathcal{N}(0, \sigma^2_{\omega_{V^2}})$.
4) For each user $i$, draw a latent user offset vector $\epsilon_i \sim \mathcal{N}(0, \sigma^2_{\omega_{\epsilon}})$.
5) Regarding user category information $X_u$ as input, draw the user latent vector.

$$u_i \sim \mathcal{N}(MFM \left(VV^1_i, VV^2, W_l, b_l, X_u \right), \sigma^2_{\omega_u})$$

- Extract item latent vectors with BLSTM

1) For each column $n$ of the left LSTM parameter matrix $W_s$, draw $W_{s,n} \sim \mathcal{N}(0, \sigma^2_{W_s})$.
2) For each column $n$ of the right LSTM parameter matrix $W'_s$, $W'_{s,n} \sim \mathcal{N}(0, \sigma^2_{W'_s})$.
3) For each column $n$ of the fully connected layer parameter $W_F$, $W_{F,n} \sim \mathcal{N}(0, \sigma^2_{W_F})$.
4) For each item $j$, draw a latent item offset vector $\delta_j \sim \mathcal{N}(0, \sigma^2_{\delta})$.
5) Regarding item document information $X_v$ as input, draw the item latent vector

$$v_j \sim \mathcal{N}(BLSTM \left(W_s, W'_s, W_F, X_v \right), \sigma^2_{\omega_v})$$

- Rating distribution is obtained by user latent vector $u_i$ and item latent vector $v_j$.

1) For each user-item pair $(i, j)$, draw a rating

$$R_{i,j} \sim \mathcal{N}(u_i^T v_j, \sigma^2_{I_{i,j}})$$

where $\sigma^2_{I_{i,j}}, \sigma^2_{V^1_{i,j}}, \sigma^2_{V^2}, \sigma^2_{W_s}, \sigma^2_{W'_s}, \sigma^2_{\omega_u}, \sigma^2_{\omega_v}$ are hyperparameters and $I_{i,j}$ is indication function that it is equal to 1 if user $i$ rated item $j$ and 0 otherwise. Critically, the same dimension outputs of the two represented components serve as a bridge between the ratings and auxiliary information. The “dual channel” approach not only allows JMF to capture the similarity...
between users (items) effectively, but also allows JMF to get excellent performance when data is sparse.

Besides, for cold start, JMF uses auxiliary information to recommend interaction information that without user-item. For newly joining users or arriving items, JMF models latent representations using basic information of users (age, gender, etc.) and item attribute information to recommend.

B. MODEL TRAINING

For the sake of optimizing the parameters of three components (MFM, BLSTM, PMF), this paper minimizes the structural risk, i.e., the maximum posterior probability (MAP), which is equivalent to the minimum negative likelihood function [32]. Therefore, the loss function is written as.

\[
\mathcal{L} = \frac{\lambda_{W_1}}{2} \sum_{j} \|W_*\|^2 + \frac{\lambda_{W_2}}{2} \sum_{j} \|W_2\|^2 + \frac{\lambda_F}{2} \sum_{j} \|W_F\|^2
\]

\[
+ \frac{\lambda_U}{2} \sum_{j} \|U_j - MFM (VV^1 \ast, VV^2, W_L, b_L, X_i)\|^2
\]

\[
+ \frac{\lambda_V}{2} \sum_{j} \|V_j - BLSTM (W_* , W_*', W_F, X_{vj})\|^2
\]

\[
+ \frac{\lambda_{VV^1 \ast}}{2} \sum_{j} \|VV^1 \ast\|^2 + \frac{\lambda_{VV^2}}{2} \sum_{j} \|VV^2\|^2
\]

where \(\lambda_{W_1}\) is \(\sigma^2/\sigma_{W_1}^2\), \(\lambda_{W_2}\) is \(\sigma^2/\sigma_{W_2}^2\), \(\lambda_F\) is \(\sigma^2/\sigma_{F}^2\), \(\lambda_U\) is \(\sigma^2/\sigma_{U}^2\), \(\lambda_V\) is \(\sigma^2/\sigma_{V}^2\), \(\lambda_{VV^1 \ast}\) is \(\sigma^2/\sigma_{VV^1 \ast}^2\), \(\lambda_{VV^2}\) is \(\sigma^2/\sigma_{VV^2}^2\), \(\lambda_U\) is \(\sigma^2/\sigma_{U}^2\), and \(\lambda_V\) is \(\sigma^2/\sigma_{V}^2\).

Since the user latent vector and the item latent vector are coupled together while other parameters exist in different components, it is very tough to directly optimize these parameters. Consequently, in the same way [6], an EM-style optimization algorithm is adopted [33]. Specifically, the method combines with the idea of alternating least squares (ALS) [29] and coordinate descent (CD). When a set of parameters are optimized, we fix all remaining parameters. Specifically, in one calculation cycle, we determine that the parameters of MFM and BLSTM are constant. With respect to \(U\) (or \(V\)), the loss function becomes a quadratic function. This paper computes the gradients of \(\mathcal{L}\) with respect to \(u_i\) and \(v_j\) as follows

\[
u_i \leftarrow \left(\frac{1}{2}V_i V^T + \lambda_U I_k\right)^{-1} \left(V R_i + \lambda_{U} MFM \left(VV^1 \ast, VV^2, W_L, b_L, X_i\right)\right)
\]

\[
v_j \leftarrow \left(\frac{1}{2}V_j V^T + \lambda_{V} I_k\right)^{-1} \left(U V_j + \lambda_{V} BLSTM \left(W_* , W_*', W_F, X_{vj}\right)\right)
\]

where \(I_k = \text{diag} (I_{1}, I_{2}, \ldots, I_{d})\), \(J\) is the number of items rated by the user \(i\). \(R_i = (R_{i1}, R_{i2}, \ldots, R_{id})^T\) is a column vector containing all the ratings of user \(i\). Through the above formulas, descriptions of user extracted by MFM is incorporated in the process of generating the user vector. Descriptions of the items extracted by BLSTM is incorporated into the item vector generation process. When it comes to the hyperparameter and \(\lambda_U\) and \(\lambda_V\), they play the role of balancing auxiliary information and rating matrix information.

In the same calculation cycle, given \(U\) and \(V\). In order to extract the auxiliary information these two vectors are used as labels to guide the two components. Thus, the above loss function (8) degenerates into the two loss functions of the two components. The two loss functions are showed as

\[
\mathcal{L}_U = \frac{\lambda_U}{2} \sum_{i} \left\| u_i - MFM \left(VV^1 \ast, VV^2, W_L, b_L, X_i\right)\right\|^2 + \frac{\lambda_{VV^1 \ast}}{2} \sum_{j} \left\|VV^1 \ast\right\|^2 + \frac{\lambda_{VV^2}}{2} \sum_{j} \left\|VV^2\right\|^2
\]

\[
\mathcal{L}_V = \frac{\lambda_V}{2} \sum_{j} \left\|v_j - BLSTM (W_* , W_*', W_F, X_{vj})\right\|^2 + \frac{\lambda_{VV^1 \ast}}{2} \sum_{i} \left\|VV^1 \ast\right\|^2 + \frac{\lambda_{VV^2}}{2} \sum_{j} \left\|VV^2\right\|^2
\]

We can employ a variety of optimization algorithms to train the two components, such as SGD, RMSprop, Adam, etc. By alternating the update of \(U\), \(V\) and other components, the several optimization processes are repeated until convergence. We approximately call the predicted rating as

\[
R_{ij} \approx \left(MFM \left(VV^1 \ast, VV^2, W_L, b_L, X_i\right) + \varepsilon_i\right) \left(BLSTM (W_* , W_*', W_F, X_{vj}) + \delta_j\right) = u_i^T v_j
\]

C. TIME COMPLEXITY ANALYSIS

JMF integrates three components: MFM, BLSTM and PMF. For MFM, similar to FM, the complexity of time is \(O(KM)\). For BLSTM, complexity of time for updating parameters is dominated by the computation of recurrent layer. All weight and bias of BLSTM are updated in \(O(2K^2n + K^3M + K^3N)\). Finally, the total time complexity per epoch is \(O(KM + 4(Kn + K) + K^2n + K^3M + K^3N)\), where \(K\) is latent vector dimension, \(n\) is word embedding dimension, and \(r\) is the number of ratings.

VI. EXPERIMENTS

The most intuitive evaluation indicator of the recommender system is rating prediction accuracy. To validate the proposed JMF, this paper uses 5 real-world open-source datasets to answer the following questions:

Q1 Whether using sparse or dense datasets or not, can our JMF achieve higher prediction accuracy compared to other competitors?
Q2 Can MFM enhance the performance of JMF by extracting user latent representations? Compared with other FMs, how effective is MFM (as a component) in joint optimization with other components?
Q3 How do the popularity of items and the number of fields affect the MFM and JMF in the long run?
Q4 How does the setting of parameters impact the performance of JMF?

A. DATASETS
To demonstrate the JFM prediction performance. Five real-world datasets were selected from different domains with explicit feedback rates, while three from MovieLens and two from Amazon. The user’s rating of the item is from 1 to 5, and the particulars of the datasets are as follows.

1) MovieLens
The movie dataset is collected by the GroupLens team and widely used to validate recommendation and information retrieval methods. The dataset is separated into multiple versions according to the number of interactions. MovieLens 100K, MovieLens 1M, MovieLens 10M are the three typical datasets. These three datasets have diverse density and can fully verify the model performance. But the original dataset does not contain document description data, which is similar to [6] this paper obtains the document description of the corresponding item from IMDB.

2) Amazon
E-commerce data are collected by Amazon for each topic and aggregates it into dataset. The dataset contains item textual comments and ratings, from which we selected datasets for two topics: Digital Music and Baby. Due to the large number of Amazon users and high fluidity, the original dataset is large but highly sparse. More than 80% of users in the Digital Music dataset only rate one item. It is quite tough for any collaborative filtering algorithms to discover the association between users (items). Therefore, this paper removes users with less than 3 ratings and items with less than 5 ratings.
Then, this paper shifts items that do not have their description documents in each dataset. Finally, the processed datasets are summarized in Table 1.

Similar to [32], the documents data are processed for all datasets as follows: 1) remove stop words, 2) calculate TFIDF value and select the first 10,000 words as dictionary according to the value, 3) filter out words that are not in dictionary, 4) set the maximum length of each sentence to 300 and delete extra words.
We should note that because of privacy, user’s base information is not included in Amazon dataset. So, we reuse the rating matrix and use user rating vector as category features. Therefore, the proposed JMF is consistent with other competitors on the premise of the same data information.

B. EVALUATION SCHEMA
The rating prediction is transformed into regression problem, so we choose the widely used evaluation protocols named root mean square error (RMSE), which can directly reflect the prediction accuracy.

\[
RMSE = \sqrt{\frac{\sum_{i,j}^{N,M} (r_{ij} - \hat{r}_{ij})^2}{ratings\_num}}
\] (12)

The training process is not fixed at each time for the reason that the parameter initialization obeys the Gauss distribution. We divide the data set five times randomly and take the average of five times as the final result.

C. BASELINES
The following baselines are used to compare with the proposed means.

- **BiasedSVD** [34]: The model introduces user and item biases into matrix factorization, which reflects user preferences. And BiasedSVD does not use item document data to assist recommendation.
- **BPMF** [4]: The model improves the PMF from the perspective of Bayesian estimation. BPMF’s hyperparameters will change with the training, which is trained by Markov chain Monte Carlo method.
- **NeuMF** [16]: The model includes two models: GMF and MLP, whose inputs are embedded latent factors. NeuMF possesses both GMF’s linear crossover ability and DNN’s non-linear crossover ability. In this paper, the embedding of user ID is used as user vector, while the item vector is concatenated with item ID embedding and document description vector, which are generated by BLSTM.
- **NeuMF∗**: The model is the NeuMF with pre-trained MLP and GMF.
- **CDL** [6]: Collaborative deep learning is the first tightly coupled model integrating deep learning and probability matrix factorization. It employs SDAE to extract text information to rich item representations.
- **deepFM** [14]: A Factorization-Machine based Neural Network, it integrates the architectures of FM and deep neural networks (DNN). It models low-order feature interactions like FM and models high-order feature interactions like DNN.
- **convMF** [7]: Convolution matrix factorization is also a tightly coupled model of probability generation. ConvMF uses convolution neural network to extract items document information. Convolution neural

| Dataset         | #users | #items | #ratings | Sparsity |
|-----------------|--------|--------|----------|----------|
| MovieLens 100K  | 943    | 1546   | 94,808   | 93.50%   |
| MovieLens 1M    | 6,040  | 3,544  | 993,482  | 95.36%   |
| MovieLens 10M   | 69,878 | 10,073 | 9,945,875| 98.59%   |
| Amazon_music    | 27125  | 3568   | 194,897  | 99.80%   |
| Amazon_baby     | 41914  | 7050   | 230,973  | 99.92%   |
network can grasp the correlation between adjacent words and improve the accuracy of rating prediction at the same time.

- **JMF**: Joint matrix factorization is the proposed method. The model leverages MFM and BLSTM to extracts latent factors in parallel. Ultimately, they are integrated into PMF in the form of probability.

- **JMF**: JMF is a weakened model which gets rid of MFM on the basis of JMF.

### D. IMPLEMENTATION DETAIL
All the models are implemented based on TensorFlow and Keras. All experiments are conducted on two Intel Xeon E5-2620v4 2.1GHz with 16 physical cores and 32 GB memory. We divide each data set into two parts randomly, including training set (80%) and test set (20%) separately. The specific structure and parameters of JMF are as follows:

1) The vector dimension of users and items is 50.
2) User rating vectors are used as user behavior data. We rank items based on popularity and separate them into three fields by default.
3) The maximum length of each item’s description sentence is 300.
4) The word embedding dimension is 200, and the embedding layer will be trained with the optimization process.
5) Dropout is used as a regularization method for BLSTM and the output dimension of BLSTM is 32*2=64 by default.
6) The above parameters are initialized with Gauss distribution with mean 0 and variance 0.01.

The original CDL and NeuMF use implicit data to complete the Top-N task. So, some modifications are made to these two models. For CDL, convMF’s convolutional neural network are replaced with SDAE and used it as CDL model. For NeuMF, this paper changes the activation function of the output layer to “Relu” and the loss function to mean square error.

As for the MovieLens 1M dataset, it is used as a representative for grid search, and the best parameters for all models are found. For NeuMF, MLP adopts four hidden layers and the size of predictive factors is 8. Similarly [16], NeuMF is initialized with pre-trained MLP and GMF. For CDL, which can be found that the model achieves the best performance when SDAE layer number \(L = 4\) and \(\lambda_U = \lambda_V = 10\). For deepFM, the model was originally designed for click-through rate prediction, here we use the optimal network structure and hyperparameters proposed by the original author for scoring prediction, the activation function is relu, the dropout is set to 0.9, the number of hidden layers is 3 and with 200 neurons per layer. For convMF, the model performs best (noting that we set up a convolutional moderate network structure according to the original paper) when \(\lambda_U = 10\) and \(\lambda_V = 100\). For our JMF, the model performs best when \(\lambda_U = 5\) and \(\lambda_V = 100\).

In order to compare these models more rapidly, this paper merely select parameters for a most balanced MovieLens 1M dataset. Parameters do not change for other datasets.

### VII. EXPERIMENTAL RESULTS

#### A. PREDICTION ACCURACY COMPARISON (Q1)
Table 2 provides information on the results of comparing the prediction accuracy of Biased SVD, BPMF, CDL, ConvMF, JMF, JMF* based on five real-world datasets. We need to note that “Improve” indicates the relative improvement of “JMF” over the best baseline “ConvMF”. It can be clearly shown that JMF outperforms several baselines significantly.

For traditional CF methods, JMF improves significantly than BiasedSVD and BPMF, which indicates that the use of deep learning components to extract auxiliary information into matrix factorization can improve prediction accuracy effectively.

For the factorization machine model, deepFM has a good performance on five data sets, it also uses deep learning components. But JMF is better than its performance.

For integration model, convMF performs slightly better than CDL on the whole. But on MovieLens 1M data set, CDL is even better than convMF. As for movielens datasets,
TABLE 3. Comparison of JMF* equipped with FM, FFM and MFM.

| Evaluation Protocols | Root Mean Squared Error (RMSE) | Training time (in seconds) |
|----------------------|---------------------------------|----------------------------|
| Models               | MovieLens 100K                  | MovieLens 1M               |
| JMF*                 | 0.9178                          | 0.8492                     | 9.21 | 23.59 |
| JMF* with FM         | 0.9081                          | 0.8416                     | 9.31 | 24.76 |
| JMF* with FFM        | 0.9040                          | 0.8395                     | 48.29 | 127.88 |
| JFM* with MFM(JMF)   | 0.9022                          | 0.8406                     | 9.79 | 24.73 |

JMF is about 1%-2.5% better than the best competitor (convMF).

For Amazon data set, JMF has a more significant improvement compared with convMF, about 6%-11%. This implies that in the case of extremely sparse data, BLSTM plays a leading role in our model, and BLSTM has better document analysis capacity than convMF’s convolutional neural network.

For neutral collaborative filtering, NeuMF* is 1%-2.5% average better than NeuMF, which implies neural network pretraining can improve model’s performance. The performance gap between JMF and NeuMF* indicates that performance can be improved by combining traditional recommendation algorithms compared with pure neural network.

Besides, this paper generates nine additional train datasets of various sparseness by random sampling from Movielens 1M. As shown in Fig. 4, JMF always outperforms other models for any degree of sparseness Except for deepFM. This is because deepFM uses deep learning components like JMF, which can better extract auxiliary information to improve prediction accuracy. Specifically, JMF is also 1.3%-2.5% approximately better than the best competitor (convMF) when data is extremely sparse (10%-20% of entire data). It can be found that the performance gap between JMF and traditional CF methods (BiasedSVD, BPMF) increases with the decline of data density. It implies that JMF can use auxiliary information to enrich latent representation effectively, thus alleviating the problem of data sparsity.

B. EFFECTIVENESS OF PROPOSED MFM (AS A COMPONENT) (Q2)

To evaluate, as a component in joint optimization with other components, we replaced MFM with FM and FFM to form two additional models. Table 3 reports the comparison results of the four models over MovieLens 100K and MovieLens 1M. Noting that the time in table refers to the average training time of per epoch.

It can be observed that JMF* is the worst performer. It demonstrates that JMF* which incorporates user representation component makes meaningful improvements in accuracy. Besides, JMF* with MFM and JMF* with FFM attaining better performance than JMF* with FM in RMSE. However, because of the complexity of FFM, JMF* with FFM takes much more training time than JMF. There is no difference in training time between JMF and JMF* with FM. In other words, when it comes to MFM, less time is cost to extract accurate user representations. Hence, MFM is the best choice as the component of JMF.

This paper has proved that using MFM to extract user representations can improve the prediction accuracy effectively. We believe that using user category data from other sources, such as demographic information, social information, etc., to generate user latent vectors will help us attain better performance.

C. THE IMPACT OF ITEM POPULARITY AND THE NUMBER OF FIELDS (Q3)

Fig. 5 illustrates the influence brought by the number of fields based on the Movielens 1M dataset, which can be found that with the increase of the number of fields, the polyline presents concave shape as a whole. JMF reaches the best performance when the number of fields is about 3 to 6. When the number of category fields is too few, the performance is deteriorated due to the degeneration of MFM into FM. In contrast, when the number of categories fields is too many, the explicit feedback in each field is so less (even 0), which cannot capture enough user preferences. Hence, the most suitable number of fields needs to be selected for MFM.

In Table 4, “Yes” indicates that items are artificially sorted according to popularity then subdivided into multiple fields. “No” denotes dividing user rating features directly into
multiple fields. “No” denotes dividing user rating features directly into multiple fields. We need to note that “Improve” indicates the relative improvement of JMF with popularity over JMF without popularity, which can be found that in the process of extracting user latent vectors, artificially sorting items according to popularity will slightly improve the accuracy of JMF rating prediction, which shows that the proposed MFM method is very sensitive to data distribution.

D. PARAMETER ANALYSIS (Q4)

In this section, this paper explores the effect of number of latent dimensions on the accuracy prediction accuracy. Because each baseline regularization parameter is inconsistent, it not shown in limited space.

Fig. 6 illustrates the changes in training and testing errors of Biased SVD and JMF as the dimension of latent vectors increases. Obviously, the proposed JMF has always outperformed Biased SVD. With the raise of dimensions, the performance of JMF becomes better and more stable. Nevertheless, for Biased SVD Interestingly, when the dimension is over 90, the training error drops, but the test error increases. This may be caused by the over-fitting.

VIII. EXTRA EXPERIMENTS OF MFM ITSELF

The MFM is a novel advanced version of FM. We have proved its effectiveness as a component in the above experiments. In fact, MFM can also accomplish some machine learning tasks independently, such as recommendation.

In this section, we compare MFM with traditional FM and FFM in terms of RMSE and training time in order to evaluate the effectiveness of MFM itself. We first provide the details about the comparison experimental.

- **Datasets** Due to Amazon sets do not include user information, MovieLens 100K and MovieLens 1M are chosen as datasets. And they are randomly divided into two parts, including training set (80%) and test set (20%). For these two datasets, we mainly consider rating data (include UserID and ItemID) and basic information of user (Gender, Age, Occupation, Zip-code). A hashing trick is applied to generate sparse features. The statistics data are summarized in Table 6.

- **Field Numbers** For FM, all features belong to the same field. For FMM, all features are assigned to 6 fields. For the proposed MFM, this paper divides 2 fields according to rating data and user information.
TABLE 6. Statistics of two real-world datasets. Features indicate binary features after transforming categorical features.

| Dataset        | #instances | #features |
|----------------|------------|-----------|
| MovieLens 100K | 100,000    | 3412      |
| MovieLens 1M   | 1,000,000  | 13189     |

- **Optimization Method** Three methods are all implemented by the same SGD method.
- **The parameter** matrices of three methods are initialized with Gauss distribution with mean 0 and variance 0.01. Table 7 represents the best performing values of Learning rate ($\eta$) and embedding dimension ($k$) found by grid search.

TABLE 7. Hyperparameter settings of three methods. $\eta$ indicates learning rate and $k$ indicates embedding dimension.

| Model | Hyperparameters |
|-------|-----------------|
| FM    | $\eta = 0.03$, $k = 40$ |
| FFM   | $\eta = 0.1$, $k = 4$ |
| MFM   | $\eta = 0.03$, $k = 20$ |

The comparison results of three methods are provided in Table 5. We need to note that the time in this table refers to the average training time of per epoch. The comparison in terms of RMSE provide us with the following two observations: 1) Clearly, traditional FM performs the worst among all kinds of approaches, which implies that modeling all feature interactions without field-aware may introduce noises and degrade the performance. 2) For MovieLens 1M datasets, FFM performs far better than MFM. However, FFM is not significantly better on MovieLens 100k. so FFM has an advantage in dealing with extreme sparse data. And when datasets contain numerical features, the performance of MFM and FFM is almost the same.

On the other side of the coin, FFM costs more significant training time than MFM. And there is no difference in training time between MFM and FM. This paper provides two possible reasons: 1) FFM has more parameters to optimize. 2) The feature interaction part of MFM and FM will be simplified to reduce the complexity of time, but FFM cannot. Therefore, MFM is a good choice by weighing performance of RMSE and training time.

IX. CONCLUSION AND FUTURE WORK

The exiting model has huge disadvantage in using document information and incorporating user behavior information. Hence, a tight coupling framework named as JMF is proposed, which integrates two feature extraction components. We innovatively designed MFM and BLSTM in order to avoid the shortcomings of FM and LSTM, which can effectively extract auxiliary information to form latent vectors for PMF. And we seamlessly integrate these two components into PMF through probability angle to form JMF. Extensive experiments indicate that JMF outperforms other classical MF and integration models significantly. In addition, MFM is an overall method that can be applied to multiple tasks alone and get outstanding performance. In future, we will further improve MFM by incorporating the attention mechanism to learn variable weights of feature interactions. This will enhance the interpretability of the model. And we intend to combine other forms of structural information to understand user behavior, such as user-generated content in social networks [35].

In this paper, JMF is designed for rating prediction. However, it has been a while since the industry acknowledge that rating prediction is not a business issue but top-n recommendation. Therefore, in the long run, JMF will be ameliorated to fulfill top-n recommendation tasks.

Moreover, we will also concentrate on the impact of word pre-training on our model, hoping that the performance of JMF can be further improved by adding pre-training.

REFERENCES

[1] J. Bobadilla, F. Ortega, A. Hernando, and A. Gutiérrez, “Recsysender systems survey,” Knowl.-Based Syst., vol. 46, pp. 109–132, Jul. 2013.
[2] H. Liu, X. Kong, X. Bai, W. Wang, T. M. Bekele, and F. Xia, “Context-based collaborative filtering for recommendation,” IEEE Access, vol. 3, pp. 1695–1703, 2015.
[3] R. Salakhutdinov and A. Mnih, “Probabilistic matrix factorization,” in Proc. Int. Conf. Neural Inf. Process. Syst., 2007, pp. 1–8.
[4] R. Salakhutdinov and A. Mnih, “Bayesian probabilistic matrix factorization using Markov chain Monte Carlo,” in Proc. 25th Int. Conf. Mach. Learn. (ICML), 2008, pp. 880–887.
[5] Z. Zhang, Y. Liu, and Z. Zhang, “Field-aware matrix factorization for recommender systems,” IEEE Access, vol. 6, pp. 45690–45698, 2018.
[6] H. Wang, N. Wang, and D. Y. Yeung, “Collaborative deep learning for recommender systems,” in Proc. 21th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining, 2015, pp. 1235–1244.
[7] D. Kim, C. Park, and J. Oh, “Convolutional matrix factorization for document context-aware recommendation,” in Proc. 10th ACM Conf. Recommender Syst., 2016, pp. 233–240.
[8] F. Zhang, N. J. Yuan, D. Lian, X. Xie, and W.-Y. Ma, “Collaborative knowledge base embedding for recommender systems,” in Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining, Aug. 2016, pp. 353–362.
[9] Aaron, D O Van, S Dieleman, and B. Schrauwen, “Deep content-based music recommendation,” in Proc. Adv. Neural Inf. Process. Syst., 2013, pp. 2643–2651.
[10] L. Nie, X. Song, and T.-S. Chua, “Learning from multiple social networks,” Synth. Lectures Inf. Concepts, Retr., Services, vol. 8, no. 2, pp. 1–118, Apr. 2016.
[11] S. Zhang and L. N. Yao, “Deep learning based recommender system: A survey and new perspectives,” ACM Comput. Surv., vol. 52, no. 1, 5. 2017.
[12] J. Lian, X. Zhou, and F. Zhang, “xDeepFM: Combining explicit and implicit feature interactions for recommender systems,” in Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining, 2018, pp. 1754–1763.
[13] H. T. Cheng, L. Koe, and J. Harmens, “Wide & deep learning for recommender systems,” in Proc. 1st Workshop Deep Learn. Recommender Syst., 2016, pp. 7–10.
[14] H. Guo, R. Tang, Y. Ye, Z. Li, and X. He, “DeepFM: A factorization-machine based neural network for CTR prediction,” 2017, arXiv:1703.04247. [Online]. Available: http://arxiv.org/abs/1703.04247.
[15] D. Erhan, Y. Bengio, and A. Courville, “Why does unsupervised prep-training help deep learning?” J. Mach. Learn. Res., vol. 11, no. 3, pp. 625–660, 2010.
[16] X. He, L. Liao, and H. Zhang, “Neural collaborative filtering,” in 26th Int. Conf. World Wide Web, 2017, pp. 173–182.
[17] 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.
[18] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, “Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion,” J. Mach. Learn. Res., vol. 11, no. 12, pp. 3371–3408, Dec. 2010.

[19] L. Zheng, V. Noroozi, and P. S. Yu, “Joint deep modeling of users and items using reviews for recommendation,” in Proc. 10th ACM Int. Conf. Web Search Data Mining (WSDM), 2017, pp. 425–434.

[20] S. Rendle, “Factorization machines,” in Proc. IEEE Int. Conf. Data Mining, Dec. 2010, pp. 995–1000.

[21] Yin, Wenpeng, “Comparative study of CNN and RNN for natural language processing,” Tech. Rep., 2017.

[22] S. Seo, J. Huang, H. Yang, and Y. Liu, “Interpretable convolutional neural networks with dual local and global attention for review rating prediction,” in Proc. 14th ACM Conf. Recommender Syst., Aug. 2017, pp. 297–305.

[23] D. Hyun, C. Park, M. C. Yang, I. Song, and J. T. Lee, “Review sentiment-guided scalable deep recommender system,” in Proc. 41st Int. ACM SIGIR Conf. Res. Develop. Inf. Retr., 2018, pp. 965–968.

[24] A. Graves and J. Schmidhuber, “Framewise phoneme classification with bidirectional LSTM and other neural network architectures,” Neural Netw., vol. 18, nos. 5–6, pp. 602–610, 2005.

[25] Y. Juan, Y. Zhuang, W.-S. Chin, and C.-J. Lin, “Field-aware factorization machines for CTR prediction,” in Proc. 10th ACM Conf. Recommender Syst., Sep. 2016, pp. 43–50.

[26] Y. Qu, H. Cai, K. Ren, W. Zhang, Y. Yu, Y. Wen, and J. Wang, “Product-based neural networks for user response prediction,” in Proc. IEEE 16th Int. Conf. Data Mining (ICDM), Dec. 2016, pp. 1149–1154.

[27] J. Xiao, H. Ye, X. He, H. Zhang, F. Wu, and T.-S. Chua, “Attentional factorization machines: Learning the weight of feature interactions via attention networks,” 2017, arXiv:1708.04617. [Online]. Available: http://arxiv.org/abs/1708.04617

[28] B.-W. Chen, W. Ji, S. Rho, and Y. Gu, “Supervised collaborative filtering based on ridge alternating least squares and iterative projection pursuit,” IEEE Access, vol. 5, pp. 6600–6607, 2017.

[29] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” Neural Comput., vol. 9, no. 8, pp. 1735–1780, 1997.

[30] P. Covington, J. Adams, and E. Sargin, “Deep neural networks for YouTube recommendations,” in Proc. 10th ACM Conf. Recommender Syst., Sep. 2016, pp. 191–198.

[31] J. Ramos, “Using TF-IDF to determine word relevance in document queries,” in Proc. 1st Instructional Conf. Mach. Learn., 2003, pp. 133–142.

[32] H. Wang and W.-J. Li, “Relational collaborative topic regression for recommender systems,” IEEE Trans. Knowl. Data Eng., vol. 27, no. 5, pp. 1343–1355, May 2015.

[33] D. Kushary, “The EM algorithm and extensions,” Technometrics, vol. 40, no. 3, p. 260, 1998.

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

[35] X. Song, Z. Y. Ming, L. Nie, Y. L. Zhao, and T. S. Chua, “Volunteerism tendency prediction via harvesting multiple social networks,” ACM Trans. Inf. Syst., vol. 34, no. 2, p. 10, 2016.

SHAOLUN SUN received the M.S. degree from Tokyo Agricultural and Technical University, Japan, in 2018. He is currently pursuing the Ph.D. degree with the Department of Control Science and Engineering, University of Science and Technology at Beijing. His research interests include deep learning, machine learning, and complex system modeling.

YUE TONG XIAO is currently pursuing the degree with the School of Geophysics and Information Technology, China University of Geosciences at Beijing.

XIAOLI SU is currently pursuing the Ph.D. degree with the University of Science and Technology at Beijing, Beijing, China. Her current research interests include machine learning, complex system modeling, and intelligent control.

YUE HUANG is currently pursuing the M.S. degree with the Department of Instrument Science and Technology, University of Science and Technology at Beijing. His research interests include recommendation systems and deep learning for social recommendation.

SEN ZHANG received the Ph.D. degree in electrical from Nanyang Technological University, in 2005. She has been a Postdoctoral Research Fellow with the National University of Singapore and a Lecturer in Charge with Singapore Polytechnic. She is currently an Associate Professor with the School of Automation and Electrical Engineering, University of Science and Technology at Beijing. Her research interests include extreme learning machine, target tracking, and estimation theory.

HENG ZHENG is currently pursuing the M.S. degree with the Department of Instrument Science and Technology, University of Science and Technology at Beijing. His research interest includes deep learning and its application to recommender systems.

WENDONG XIAO (Senior Member, IEEE) received the Ph.D. degree from Northeastern University, China, in 1995. From 1996 to 1999, he was a Postdoctoral Research Fellow with the POSCO Technical Research Laboratories, South Korea, from 1999 to 2001, he was an Associate Professor with Northeastern University, from 2001 to 2004, he was a Research Fellow with Nanyang Technological University, Singapore, and from 2004 to 2012, he was a Scientist with the Institute for Infocomm Research, Agency for Science, Technology and Research, Singapore. He is currently a Professor with the School of Automation and Electrical Engineering, University of Science and Technology at Beijing, Beijing, China. He has authored about 150 papers in journals and conferences and has been participating in several research and industrial projects in the related areas. He has been actively participating in the organizations for more than 70 international conferences. His current research interests include wireless localization and tracking, energy harvesting based network resource management, wearable computing for healthcare, big data processing, wireless sensor networks, and the Internet of Things. He is a reviewer for many top international journals.