A Deep Graph Neural Network-based Mechanism for Social Recommendations

Zhiwei Guo, and Heng Wang*

***Abstract*—Nowadays, information overload issue is gradually exposed in the Internet of things (IoT), calling for researches on recommender system in advance for industrial IoT scenarios. With the ever-increasing prevalence of various social networks, social recommendations (SoR) will certainly become an integral application that provide more feasibly personalized information service for future IoT users. However, almost all of existing researches managed to explore and quantify correlations between user preferences and social relationships, while neglecting the correlations among item features which could further influence topologies of some social groups. To tackle with this challenge, in this paper, a deep Graph Neural Network-based Social Recommendation framework (GNN-SoR) is proposed for future IoT. First, user feature space and item feature space are abstracted as two graph networks and respectively encoded via graph neural network method. Next, two encoded spaces are embedded into two latent factors of matrix factorization to complete missing rating values in user-item rating matrix. Finally, a large amount of experiments are conducted on three real-world datasets to verify efficiency and stability of the proposed GNN-SoR.**

***Index Terms*—Internet of Things, recommender system, graph neural network, deep learning**

I. Introduction

FOR a long time, people have been committed to creating a universally connected world where human beings and computers are deeply interacted. In recent years, the rapid development of Internet of Things (IoT) technology contributes a lot to realization of this vision. Predictably, IoT will become another cyberspace closely related to daily life of people in the future, where colorful social, recreational and working activities are able to be carried out. In contrast to prevalence of future IoT, explosive increase of information amount will become inevitable circumstances in it, possibly making IoT users sunk into the ocean of information and cannot find the information they really require. To avoid information overload problem, it is of great significance to investigate recommender system (RS) in advance for industrial IoT scenarios. RS suggests suitable items for users according to their preference feedback, and is certainly a promising application in future IoT. In consideration of ever-increasing popularity of various social networks, integration of social network into RS will provide IoT users more suitably personalized information service. Thus, this paper focuses on research of social recommendations (SoR) for future IoT users. Social networks are essentially a kind of graph network with two core components: users and the relationships among users. Thus, one universal scenario of SoR can be expressed as the inferring unknown preference feedback through information of social relationships. However, generating recommendations in social networks is never an easy task, because representation and quantification of various relationship features remain challenging.

To solve this problem, a considerable number of researchers have engaged in the investigation of SoR for a long period, producing a considerable number of related techniques [1]-[23]. Almost all of them managed to explore and quantify correlations between user preferences and social relationships. Obviously, users are more possible to be affected by those who have stronger social relationships. Nevertheless, existing solutions just concentrated on modeling of user feature space, while neglecting item feature space. In fact, correlations also exist among item features, which will eventually impact the real preference features of social network users and internal topologies in social networks. Fig. 1 gives a typical example to demonstrate this view. As for the movie *Saving Private Ryan*, it has many attributes such as director, 1st actor, 2nd actor, etc. It can be easily found that there are relationships among all the attributes (e.g., Steven Spielberg ever cooperated with Tom Hanks). For many social groups, what really arouses their interest is not Steven Spielberg or Tom Hanks, but the combination of them two. Especially for RS in the scene of IoT, relationships among item features are expected to act as a more critical element.

To tackle with the challenge, intuitively, feature space of SoR is supposed to be regarded as the synthesis of user feature and item feature (shown as Fig. 2). User feature is the concatenation of inherent preference and his social influence, which is universally studied by previous works. While item feature leverages concatenation of item attributes and their relationships, which is what this research distinguishes itself

Manuscript received January 31, 2020; revised March 3 and March 23, 2020; accepted March 30, 2020. This work was supported in part by the Chongqing Natural Science Foundation of China under Grant cstc2019jcyj-msxmX0747, in part by the State Language Commission Research Program of China under Grant YB135-121, in part by the National Key Research & Development Program of China under Grant 2016YFE0205600, and in part by the Key Research Project of Chongqing Technology and Business University under Grant ZDPTTD201917 and Grant KFJJ2018060.

Zhiwei Guo is with Chongqing Engineering Laboratory for Detection, Control and Integrated System, Chongqing Technology and Business University, Chongqing 400067, China. (e-mail: zwguo@ctbu.edu.cn).

*(Corresponding Author) Heng Wang is with the College of Mechanical and Electrical Engineering, Henan Agricultural University, Zhengzhou 450002, China. (e-mail: dawn_wangh@163.com).

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.2020.2986316, IEEE Transactions on Industrial Informatics

from existing ones. Specially, the user feature and item feature are fundamentally two graph networks including entities and relations which can be well represented by the novel technique graph neural network. Thus, in this paper, a deep **G**raph **N**eural **N**etwork-based **So**cial **R**ecommendation framework (GNN-SoR) is proposed for future IoT. First, user feature space and item feature space are abstracted as two graph networks and respectively encoded through graph neural network method. Next, the encoded two spaces are viewed as two latent factors in the process of matrix factorization which is formulated to predict unknown preference ratings. Finally, a large amount of experiments are conducted on three real-world datasets to evaluate efficiency and stability of the proposed GNN-SoR.

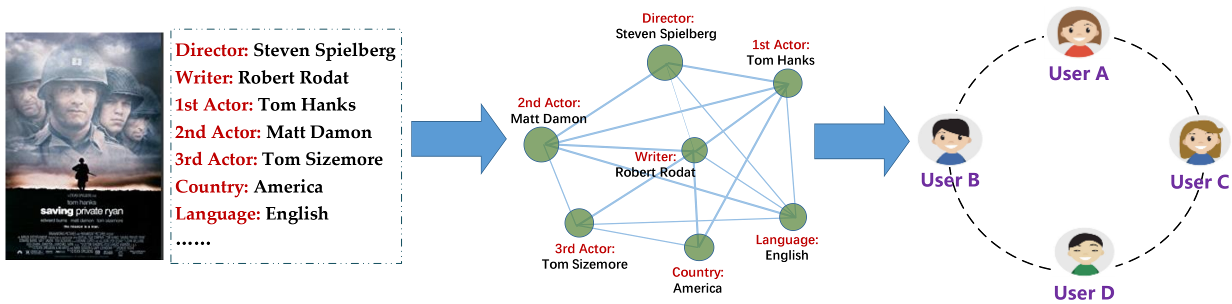


Fig. 1. A Typical Example to Illustrate Relationships Among Item Attributes.

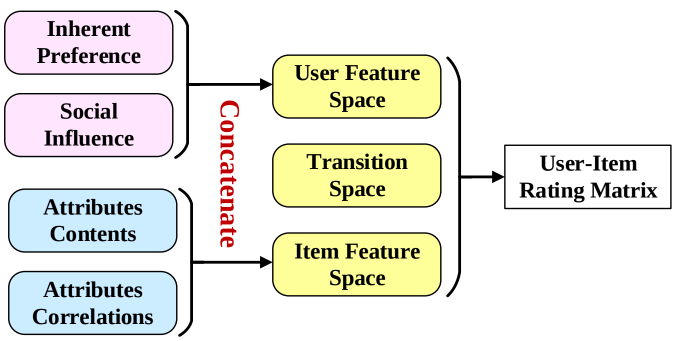


Fig. 2. Flowchart for Solution Thoughts of This Research.

II. Problem Statement

Let $u_i (i = 1,2,\cdots,n)$ be the set of n users and $v_j (j = 1,2,\cdots,m)$ be the set of m items. It is assumed that user u_i may interact with item v_j through releasing preference rating r_{ij}, and that $\boldsymbol{R} = \{r_{ij}\} \epsilon \mathbb{R}^{n\times m}$ denotes the user-item rating matrix. As it is unlikely that each user has interaction records with all the items, some elements of the initial rating matrix are likely to be empty and can be represented as $r_{ij} = 0$.

For user u_i, he can create or delete social relations with another user $u_k (k = 1,2,\cdots,n; k \neq i)$ in the user set. Note that this paper just considers whether the social relations exist or not, regardless of strength degree. Thus, adjacency matrix of user-user relations is represented by $\mathcal{N}_{ik} (i,k = 1,2,\cdots,n; i \neq k)$, in which $\mathcal{N}_{ik}$ is binary and has two possible values 0 or 1. Among, $\mathcal{N}_{ik} = 1$ denotes that user u_i has social relations with user u_k, and $\mathcal{N}_{ik} = 0$ denotes no social relations.

For item v_j, its attributes are denoted as $a_{j\alpha} (\alpha = 1,2,\cdots,z)$. It is assumed that adjacency matrix of attribute-attribute correlations of item v_j is represented as $\boldsymbol{A} = \boldsymbol{\mathcal{A}}_j(\alpha,\beta)$ where $\alpha,\beta = 1,2,\cdots,z$ and $\beta \neq \alpha$. Clearly, α and β are the indexes of attribute pairs. Similarly, elements of $\boldsymbol{A}$ are binary of 0 or 1, where $\boldsymbol{\mathcal{A}}_j(\alpha,\beta) = 1$ denotes existence of correlations and $\boldsymbol{\mathcal{A}}_j(\alpha,\beta) = 0$ otherwise.

Given above, main goal of the SoR task is to predict unknown preference ratings in the rating matrix $\boldsymbol{R}$. The roadmap of the proposed GNN-SoR is illustrated in Fig. 3.

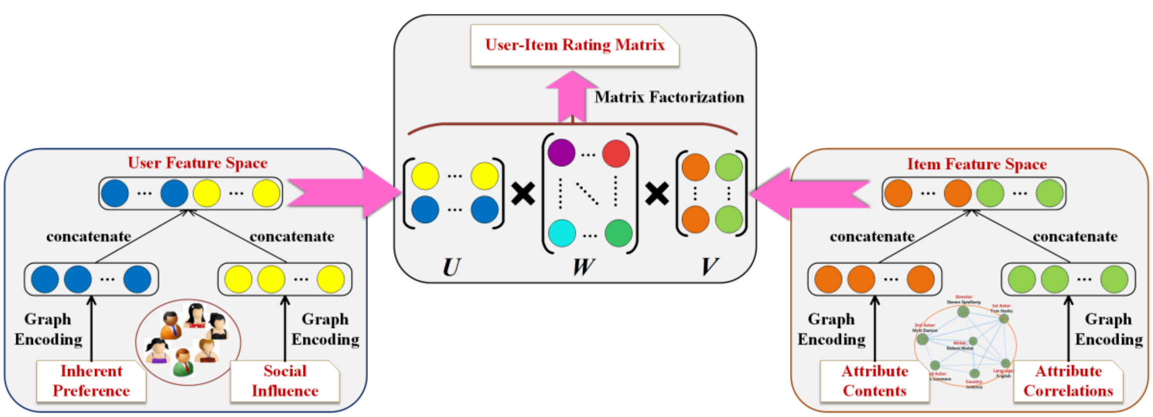


Fig. 3. Roadmap Architecture of the GNN-SoR.

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.2020.2986316, IEEE Transactions on Industrial Informatics

III. Methodology

This section describes detailed mathematical modeling process of the proposed GNN-SoR framework and contains three subsections: user feature modeling, item feature modeling and rating prediction.

A. User Feature Modeling

1) Inherent Preference

Inherent preference in this research refers to known preference features of users, and an m-dimensional inherent preference vector $p_i(i=1,2,\cdots,n)$ is supposed to be set up for each user u_i to represent his inherent preference. A typical case of p_i is illustrated as $\boldsymbol{p}_i=[r_{i1},r_{i2},\cdots,r_{im}]$, where m is the number of items. The purpose of inherent preference encoding is to transform the p_i into another encoded vector h_i^I called inherent preference factor, which is expressed as:

$$h_i^I=\sigma_1\{\boldsymbol{W}_1\cdot f_1(\boldsymbol{p}_i)+\boldsymbol{b}_1\} \tag{1}$$

where $\sigma_1(\cdot)$ is non-linear activation function ReLU [24], $f_1(\boldsymbol{p}_i)$ is a function that automatically extracts abstract feature expression of $\boldsymbol{p}_i$ by mapping it into a novel feature value. $\boldsymbol{p}_i$ can be transformed into another vector $\boldsymbol{p}_i'$ which comprises multiple concatenations of three elements of $\boldsymbol{p}_i$ in turn. The $\boldsymbol{p}_i'$ can be illustrated as the following vector format:

$$[(r_{i1},r_{i2},r_{i3}),(r_{i2},r_{i3},r_{i4}),\cdots,(r_{i\,m-2},r_{i\,m-1},r_{im}) \tag{2}$$

where each concatenation in $\boldsymbol{p}_i'$ is its one element. Let $\boldsymbol{p}_{i\xi}'(\xi=1,2,\cdots,\psi)$ denote all the elements in $\boldsymbol{p}_i'$. Weighted mean operator is utilized to formulate $f_1(\boldsymbol{p}_i)$ as:

$$f_1(\boldsymbol{p}_i)=\frac{1}{\psi}\sum_{\xi=1}^{\psi}a_i\cdot\left(\boldsymbol{p}_{i\xi}'\right)^{\mathrm{T}} \tag{3}$$

where $\boldsymbol{a}_i$ is three-dimensional vector. Eq. (1) can be rewritten as:

$$h_i^I=\sigma_1\left\{\boldsymbol{W}_1\left[\frac{1}{\psi}\sum_{\xi=1}^{\psi}\boldsymbol{a}_i\cdot\left(\boldsymbol{p}_{i\xi}'\right)^{\mathrm{T}}\right]+\boldsymbol{b}_1\right\} \tag{4}$$

where $\boldsymbol{W}_1$ and $\boldsymbol{b}_1$ are parameter vectors.

2) Social Influence

In this part, an n-1-dimensional vector $\boldsymbol{s}_i(i=1,2,\cdots,n)$ is set up for user u_i to represent his social influence feature, where n-1 is the number of resting users except u_i. A typical case of $\boldsymbol{s}_i$ is illustrated as $\boldsymbol{s}_i=[\mathcal{N}_{i1},\mathcal{N}_{i2},\cdots,\mathcal{N}_{i\,n-1}]$, where value of $\mathcal{N}_{ik}$ is 0 or 1. The goal of social influence encoding is to transform the $\boldsymbol{s}_i$ into one another encoded vector $\boldsymbol{h}_i^S$ called social influence factor, which is expressed as:

$$\boldsymbol{h}_i^S=\sigma_1[\boldsymbol{W}_2\cdot f_2(\boldsymbol{s}_i)+\boldsymbol{b}_2] \tag{5}$$

where $f_2(\boldsymbol{s}_i)$ is another function that automatically extracts abstract feature expression of $\boldsymbol{s}_i$. Mean operator is introduced to construct mapping function as:

$$f_2(\boldsymbol{s}_i)=\boldsymbol{c}_i\cdot(\boldsymbol{s}_i)^{\mathrm{T}} \tag{6}$$

where $\boldsymbol{c}_i$ is the weight vector for feature transition demonstrated as:

$$\boldsymbol{c}_i=\boldsymbol{\omega}_1^{\mathrm{T}}\cdot\,\sigma_1(\boldsymbol{\omega}_2^{\mathrm{T}}\cdot\boldsymbol{c}_i'+\tau_2)+\tau_1 \tag{7}$$

where $\boldsymbol{c}_i'$ is latent hidden variable of user u_i, and $\boldsymbol{\omega}_1^{\mathrm{T}}$, $\boldsymbol{\omega}_2^{\mathrm{T}}$, τ_1 and τ_2 are model parameters. Eq. (5) can be rewritten as:

$$\boldsymbol{h}_i^S=\sigma_1\{\boldsymbol{W}_2[\boldsymbol{\omega}_1^{\mathrm{T}}\cdot\,\sigma(\boldsymbol{\omega}_2^{\mathrm{T}}\cdot\boldsymbol{c}_i'+\tau_2)+\tau_1]\cdot(\boldsymbol{s}_i)^{\mathrm{T}}+\boldsymbol{b}_2\} \tag{8}$$

where $\boldsymbol{W}_2$ and $\boldsymbol{b}_2$ are parameter vectors. Note that introduction of two trade-off parameters is to improve generalization capability of the item feature subspace.

3) Concatenation

The combination process here is implemented with the aid of multi-layer perception, where the first hidden layer vector is concatenation of $\boldsymbol{h}_i^I$ and $\boldsymbol{h}_i^S$. The process is a ζ-round iterative process and illustrated as the following formulas:

$$\boldsymbol{U}_i^{(0)}=[\boldsymbol{h}_i^I\oplus\boldsymbol{h}_i^S] \tag{9}$$

$$\boldsymbol{U}_i^{(1)}=\sigma_1\left[\boldsymbol{W}_3\cdot\boldsymbol{U}_i^{(0)}+\boldsymbol{b}_3\right] \tag{10}$$

$$\cdots$$

$$\boldsymbol{U}_i^{(\zeta)}=\sigma_1\left[\boldsymbol{W}_3\cdot\boldsymbol{U}_i^{(\zeta-1)}+\boldsymbol{b}_3\right] \tag{11}$$

where $\boldsymbol{W}_3$ and $\boldsymbol{b}_3$ are parameter vectors, and contents in the upper right corner bracket of $\boldsymbol{U}_i$ is the number of iterative rounds.

B. Item Feature Modeling

1) Attribute Contents

As attribute contents of items are usually not feasible for vectorized computation or convolutional computation, especially some of them are textual data, it is expected to map them into vectorized numerical data. Attribute contents of items can be roughly divided into two types: structured attributes and unstructured attributes.

Taking the movie *Saving Private Ryan* in Fig. 1 as an example, listed attributes (Director, 1st Actor, Language, Country, etc.) in Fig. 1 are all structured attributes, because their contents are a fixed range of options (specific names of persons). Contents of categorical attributes are encoded into feature vector via one-hot encoding demonstrated as TABLE I.

TABLE I
An Example For Structured Attributes Encoding

Attributes	Encoded Formats
Director	$[1,0,\cdots,0]$
1st Actor	$[0,1,\cdots,0]$
Language	$[0,0,\cdots,1]$
Country	$[0,1,\cdots,0]$
…	…

As for other possible attributes such as textual reviews, the Twitter-LDA algorithm [25] is utilized to extract topics. Twitter-LDA is generally applied to short text classification, and is able to assign each paragraph of text a latent topic from a set of latent topics. The total number of latent topics can be determined manually or adaptively. All the textual contents associated with one item are regarded as a whole for text classification, regardless of different length. Mapping result of textual data is classification results whose are also format of one-hot encoding.

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.2020.2986316, IEEE Transactions on Industrial Informatics

Due to the possibility that dimensions of different attributes are not same, attribute with the most dimensions will be selected as uniformity. Therefore, zeros are added to corresponding encoding results of other attributes to unify dimension of all the attributes. Dimension number of attributes is denoted as D, thus content factor of attributes for item v_j can be represented as:

$$\boldsymbol{C}_j = \left[c(a_{j1}), c(a_{j2}), \cdots, c(a_{jz})\right] \tag{12}$$

where z is the number of attributes for item v_j and $c(a_{j1}) \in \mathbb{R}^D$.

2) Attribute Correlations

For attribute $a_{j\alpha}$ of item v_j, it is supposed to aggregate correlation information from other attributes and make it encoded. The encoding vector of attribute correlations for item v_j is expressed as:

$$\boldsymbol{E}_j = \sum_{\alpha=1}^{z} \sum_{\substack{\beta=1 \\ \beta \neq \alpha}}^{z} \boldsymbol{\mathcal{A}}_j(\beta, \alpha) \cdot \boldsymbol{\mathcal{M}} \tag{13}$$

where $\boldsymbol{\mathcal{M}}$ is the transformation function, and $\boldsymbol{\mathcal{A}}_j(\beta, \alpha)$ is the weight of directed correlation edge from attribute $a_{j\beta}$ to attribute $a_{j\alpha}$. Specifically, $\mathcal{A}_j(\alpha, \beta)$ is inferred as:

$$w_j(\alpha, \beta) = \frac{Count(a_{j\alpha} \cap a_{j\beta}) / Count(a_{j\beta})}{\sum_\gamma Count(a_{j\alpha} \cap a_{j\gamma}) / Count(a_{j\gamma})} \tag{14}$$

$$\boldsymbol{\mathcal{A}}_j(\alpha, \beta) = \begin{cases} w_j(\alpha, \beta) & , if\ \alpha \neq \beta \\ 0 & , else \end{cases} \tag{15}$$

where $Count(a_{j\alpha} \cap a_{j\beta})$ is the cooccurrence frequency of attribute $a_{j\alpha}$ and attribute $a_{j\beta}$, $Count(a_{j\beta})$ is occurrence frequency of attribute $a_{j\beta}$, and γ is another index number for attributes of item v_j ($\gamma \neq \alpha, \beta$). As for $\boldsymbol{\mathcal{M}}$, each correlation edge assigned a unique transformation weight will lead to complex computation and redundant time consumption. Following [26], each attribute $a_{j\alpha}$ is assigned an input matrix $\boldsymbol{\mathcal{M}}_{j\alpha}^{(IN)}$ and an output matrix $\boldsymbol{\mathcal{M}}_{j\alpha}^{(OUT)}$. As for a directed correlation edge from attribute $a_{j\alpha}$ to attribute $a_{j\beta}$, state information is transformed from $\boldsymbol{\mathcal{M}}_{j\alpha}^{(OUT)}$ to $\boldsymbol{\mathcal{M}}_{j\beta}^{(IN)}$, which is expressed as:

$$\boldsymbol{\mathcal{M}}_{j(\alpha \to \beta)} = \boldsymbol{\mathcal{M}}_{j\alpha}^{(OUT)} \cdot \boldsymbol{\mathcal{M}}_{j\beta}^{(IN)} \tag{16}$$

Similarly, as for a directed correlation edge from attribute $a_{j\beta}$ to attribute $a_{j\alpha}$, state information is calculated as:

$$\boldsymbol{\mathcal{M}}_{j(\beta \to \alpha)} = \boldsymbol{\mathcal{M}}_{j\beta}^{(OUT)} \cdot \boldsymbol{\mathcal{M}}_{j\alpha}^{(IN)} \tag{17}$$

Thus, Eq. (13) is rewritten as:

$$\boldsymbol{E}_j = \sum_{\alpha=1}^{z} \sum_{\substack{\beta=1 \\ \beta \neq \alpha}}^{z} \boldsymbol{\mathcal{A}}_j(\beta, \alpha) \cdot \boldsymbol{\mathcal{M}}_{j\beta}^{(OUT)} \cdot \boldsymbol{\mathcal{M}}_{j\alpha}^{(IN)} + \boldsymbol{b}_E \tag{18}$$

where $\boldsymbol{b}_E$ is bias vector.

3) Concatenation

For item v_j, z attributes are viewed as nodes and correlations among attributes are viewed as edges in a graph structure. In order to reduce computational complexity, here we aggregate attribute correlations for each attribute into vector $\boldsymbol{E}_j$ that is regarded as edges in the graph. Given graph structure $\mathcal{G}(\boldsymbol{C}_j, \boldsymbol{E}_j)$, item feature space is constructed through updating factors of attribute contents and attribute correlations. Let $C_{j\alpha}$ denote α-th element in $\boldsymbol{C}_j$ and $E_{j\alpha}$ denote the α-th element in $\boldsymbol{E}_j$. Updating process of attribute contents factor C_j is shown as:

$$C_{j\alpha}^{(t)} = \sigma_2\left[C_{j\alpha}^{(t-1)} + \sum_{\alpha=1}^{z} \sum_{\substack{\beta=1 \\ \beta \neq \alpha}}^{z} \boldsymbol{h}_j^{(t-1)}\right] \tag{19}$$

where $\sigma_2(\cdot)$ is the sigmoid function expressed as:

$$\sigma_2(x) = \frac{1}{1 + e^{-x}} \tag{20}$$

$\boldsymbol{h}_j^{(t-1)}$ is hidden neighborhood vector at t-1-th iteration with respect to all attribute pairs, and can be computed with consideration of elements in vector $\boldsymbol{E}_j$:

$$\boldsymbol{h}_j^{(t)} = \sigma_1\left[\boldsymbol{W}_4 \begin{bmatrix} C_{j\beta}^{(t)} \\ E_{j\alpha}^{(t)} \end{bmatrix} + \boldsymbol{b}_4\right] \tag{21}$$

where $\boldsymbol{W}_4$ and $\boldsymbol{b}_4$ are parameter vectors. The above iterative process can be also used to update $E_{j\alpha}^{(t)}$ as following formulas:

$$E_{j\alpha}^{(t+1)} = \sigma_2\left[E_{j\alpha}^{(t)} + E_{j\alpha}^{\prime\,(t)}\right] \tag{22}$$

$$E_{j\alpha}^{\prime\,(t+1)} = \sigma_1\left[\boldsymbol{W}_5\left(C_{j\alpha}^{(t)} + C_{j\beta}^{(t)}\right) + \boldsymbol{b}_5\right] \tag{23}$$

where $\boldsymbol{W}_5$ and $\boldsymbol{b}_5$ are parameter vectors. Finally, the item feature space can be obtained:

$$\boldsymbol{V}_j = C_{j\alpha}^{(t)} \tag{24}$$

And $\boldsymbol{V}_j$ is a $z \times m$-dimensional matrix.

C. Rating Prediction

Then, missing rating values in user-item matrix can be completed through dot product of those two latent matrices. Due to $\boldsymbol{U}_i \in \mathbb{R}^{n \times n}$ and $\boldsymbol{V}_j \in \mathbb{R}^{z \times m}$, dot product operations cannot be conducted between them. Therefore, a transition matrix $\boldsymbol{W}_{UV} \in \mathbb{R}^{n \times z}$ is defined to be located between the two matrices to match their dimensions. The rating prediction process can be expressed as:

$$\hat{r}_{ij} = \boldsymbol{U}_i \cdot \boldsymbol{W}_{UV} \cdot \boldsymbol{V}_j \tag{25}$$

Conventionally, matrix factorization method appropriates missing rating values by solving the following optimization problem:

$$\min_{U_i, W_{UV}, V_j} \frac{1}{2}\left(\hat{r}_{ij} - r_{ij}\right)^2 \tag{26}$$

where r_{ij} is real observed rating values. To avoid overfitting, it is supposed to introduce regularization terms related to $\boldsymbol{U}_i$, $\boldsymbol{W}_{UV}$, and $\boldsymbol{V}_j$. $\boldsymbol{U}_i$ is concatenation of $\boldsymbol{h}_i^I$ and $\boldsymbol{h}_i^S$, and $\boldsymbol{V}_j$ is concatenation of $\boldsymbol{C}_j$ and $\boldsymbol{E}_j$. Thus, Eq. (26) can be rewritten as

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.2020.2986316, IEEE Transactions on Industrial Informatics

$$\min_{U_i, W_{UV}, V_j} \frac{1}{2}\left\|\hat{r}_{ij} - r_{ij}\right\|_F^2 + \frac{\lambda_1}{2}\left(\|\boldsymbol{h}_i^I\|_F^2 + \left\|\boldsymbol{h}_i^S\right\|_F^2\right) + \frac{\lambda_2}{2}\left(\left\|\boldsymbol{C}_j\right\|_F^2 + \left\|\boldsymbol{E}_j\right\|_F^2\right) + \frac{\lambda_3}{2}\|\boldsymbol{W}_{UV}\|_F^2 \quad (27)$$

where λ_1, λ_2, and λ_3 are regularization parameters satisfying $\lambda_1, \lambda_2, \lambda_3 > 0$, and $\|\cdot\|_F^2$ is the Frobenius norm [27].

Stochastic gradient descent (SGD) is utilized to solve the optimization problem. SGD seeks out minimum of the objective function through calculating partial derivatives of parameters and updates them iteratively to reduce empirical error. Let L denote component of objective function, partial derivatives are calculated as following formulas:

$$\frac{\partial L}{\partial \boldsymbol{h}_i^I} = \left(\hat{r}_{ij} - r_{ij}\right)\frac{\partial \hat{r}_{ij}}{\partial \boldsymbol{h}_i^I} + \lambda_1 \boldsymbol{h}_i^I \quad (28)$$

$$\frac{\partial L}{\partial \boldsymbol{h}_i^S} = \left(\hat{r}_{ij} - r_{ij}\right)\frac{\partial \hat{r}_{ij}}{\partial \boldsymbol{h}_i^S} + \lambda_1 \boldsymbol{h}_i^S \quad (29)$$

$$\frac{\partial L}{\partial \boldsymbol{C}_j} = \left(\hat{r}_{ij} - r_{ij}\right)\frac{\partial \hat{r}_{ij}}{\partial \boldsymbol{C}_j} + \lambda_2\left(\boldsymbol{C}_j + E_{j\alpha}\frac{\partial \boldsymbol{E}_j}{\partial \boldsymbol{C}_j}\right) \quad (30)$$

$$\frac{\partial L}{\partial \boldsymbol{E}_j} = \left(\hat{r}_{ij} - r_{ij}\right)\frac{\partial \hat{r}_{ij}}{\partial \boldsymbol{E}_j} + \lambda_2\left(\boldsymbol{E}_j + \boldsymbol{C}_j\frac{\partial \boldsymbol{C}_j}{\partial \boldsymbol{E}_j}\right) \quad (31)$$

$$\frac{\partial L}{\partial \boldsymbol{W}_{UV}} = \left(\hat{r}_{ij} - r_{ij}\right)\frac{\partial \hat{r}_{ij}}{\partial \boldsymbol{W}_{UV}} + \lambda_3 \boldsymbol{W}_{UV} \quad (32)$$

Then, the computed partial derivatives are regarded as search directions at each iterative round to search for optimal solutions until convergence.

IV. Experiments and Analysis

A. Datasets

Datasets utilized in this research are three classical real-world datasets that are common in the field of SoR: **Epinions** [28], **Yelp** [29], **Flixster** [30]. We present descriptions of them and necessary preprocessing steps in this subsection.

Epinions—It was collected by Paolo Messa *et al.* from famous online shopping website epinions[1] through Internet crawlers in the year of 2003. In this website, users are allowed to publish preference ratings towards items with an integer ranging from 1 to 5, and to set up trust relationships with others to facilitate making decisions. We filter out ratings whose values are less than four, and remove users releasing less than 5 ratings and items with no more than 5 ratings. As the initial Epinions contains no information of attribute contents and correlations, we crawl metadata (product attributes) from famous shopping website Amazon[2] according to product information in Epinions. Then, correlation information of crawled attributes is quantified by searching co-occurrence records of attribute pairs. And the attribute correlation is represented as binary value of 0 or 1, where 1 denotes the existence of correlation and 0 otherwise.

Yelp—Yelp[3], a famous business review website in San Francisco, the United States, was founded in the year of 2004. It this website, users can rate merchants, submit comments, exchange shopping experience, etc. The Yelp dataset was collected and published by Yelp official, in order to call for innovative data mining methods for commercial development. The dataset contains information of users, information of merchants, preference ratings of users towards merchants, and reviews of users on merchants, etc. Among, merchants in the dataset are regarded as items, and correlations of attribute pairs can be extracted from information of merchants. We firstly select data just with category of restaurants for evaluation as total amount of data is too large, and then filter out users whose number of rating records is less than three. Besides, social relationships cannot be observed directly in Yelp dataset, users commonly reviewing one same item are viewed as being socially related to each other.

Flixster—Flixster[4] is an online social website concerning movies where users can share preference ratings and meet others. The Flixster dataset was collected by researchers from Arizona State University. It contains rich preference ratings of users towards items and social relationships among users. As for attributes of items, we crawl metadata from a famous online movie database named IMDb[5] and extract correlations of attributes. Note that ratings in Flixster are real values from 0.5 to 5.0 with step 0.5, so we further transform the nonintegral rating values into integers.

The statistics of post-processed datasets are listed in Table II.

TABLE II
Statistics of Post-processed Datasets

Dataset	Epinions	Yelp	Flixster
Number of Users	19503	25040	21058
Number of Items	25216	13682	20867
Number of Ratings	435671	260374	905772
Number of Social Links	402218	220197	313870
Number of Attributes in Each Item	10	10	10
Number of Attribute Correlations	235276	112341	199345
Rating Density	0.089%	0.076%	0.206%
Social Density	0.063%	0.035%	0.071%

B. Experimental Settings

To evaluate efficiency of the proposed GNN-SoR framework, we employ three widely used evaluation metrics in RS: RMSE, MAE and NDCG. RMSE refers to root mean squared error, and is a usual index to measure diverse degree between real values and prediction values. MAE refers to mean absolute error, and is defined as mean value of absolute error. As for RMSE and MAE, a lower value denotes better performance. NDCG refers to normalized discounted cumulative gain, and measures the quality of ranking through discounted importance based on positions. A higher value of

[1] http:// www.epinions.com/
[2] http://www.amazon.com/
[3] http://www.yelp.com/
[4] http://www.flixster.com/
[5] http://www.imdb.com/

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.2020.2986316, IEEE Transactions on Industrial Informatics

NDCG denotes better performance. Detailed definitions of these metrics are demonstrated in [31] and [32]. And four classical methods concerning SoR are selected as baselines: TrustMF [6], SocialMF [8], TrustSVD [14], and AutoRec [33]. Their detailed descriptions are illustrated in corresponding literatures.

The proposed GNN-SoR is implemented with the aid of TensorFlow, a well-known programming toolkit for deep learning. Number of iterative rounds ψ and ζ respectively in Eq. (3) and Eq. (11) are set to 50, batch size is initially set to 2000, trade-off parameters τ_1 and τ_2 in Eq. (7) are both set to 0.5, learning rate in SGD is set to 0.01, and penalty parameters λ_1, λ_2, and λ_3 are set to 0.35, 0.35, and 0.3 respectively.

C. Results and Analysis

TABLE III, IV and V respectively illustrates the experimental results on three datasets. In each dataset, proportion of training data is set two values: 60% and 80%. For each evaluation metric, we measure its values under two different recommendation size 5 and 10. It can be observed from the three tables that the proposed GNN-SoR is able to perform better than baselines in most of the experiments except some a few situations. Even so, the gap with best performance is quite small. Overall, GNN-SoR performs better than other four baselines, which can be attributed to two aspects of reasons: 1) Graph neural network method is utilized in this paper to obtain a better representation of various complex relationships. 2) We model correlations of item attributes as an encoded vector and integrate it into total feature space, improving granularity of the feature space.

TABLE III
EXPERIMENTAL RESULTS ON EPINIONS DATASET WITH DIFFERENT PARAMETER SETTINGS

Training	Algorithms	RMSE@5 (rank)	RMSE@10 (rank)	MAE@5 (rank)	MAE@10 (rank)	NDCG@5 (rank)	NDCG@10 (rank)
Epinions (60%)	TrustMF	0.895 (5)	0.887 (2)	0.821 (2)	0.829 (3)	0.675 (5)	0.706 (3)
	SocialMF	0.891 (4)	0.916 (4)	0.842 (3)	0.833 (4)	0.690 (3)	0.699 (4)
	TrustSVD	0.875 (3)	0.892 (3)	0.869 (5)	0.818 (2)	0.684 (4)	0.692 (5)
	CUNE	0.861 (2)	0.926 (5)	0.858 (4)	0.843 (5)	0.697 (2)	0.710 (2)
	GNN-SoR	**0.852 (1)**	**0.880 (1)**	**0.802 (1)**	**0.791 (1)**	**0.711 (1)**	**0.724 (1)**
Epinions (80%)	TrustMF	0.812 (2)	0.839 (4)	**0.827 (1)**	**0.814 (1)**	0.566 (5)	0.581 (4)
	SocialMF	0.847 (5)	0.847 (5)	0.844 (3)	0.852 (5)	0.575 (4)	0.574 (5)
	TrustSVD	0.834 (3)	0.826 (3)	0.862 (5)	0.836 (3)	0.592 (2)	0.591 (3)
	CUNE	0.843 (4)	0.823 (2)	0.854 (4)	0.845 (4)	0.581 (3)	0.608 (2)
	GNN-SoR	**0.805 (1)**	**0.812 (1)**	0.833 (2)	0.821 (2)	**0.607 (1)**	**0.621 (1)**

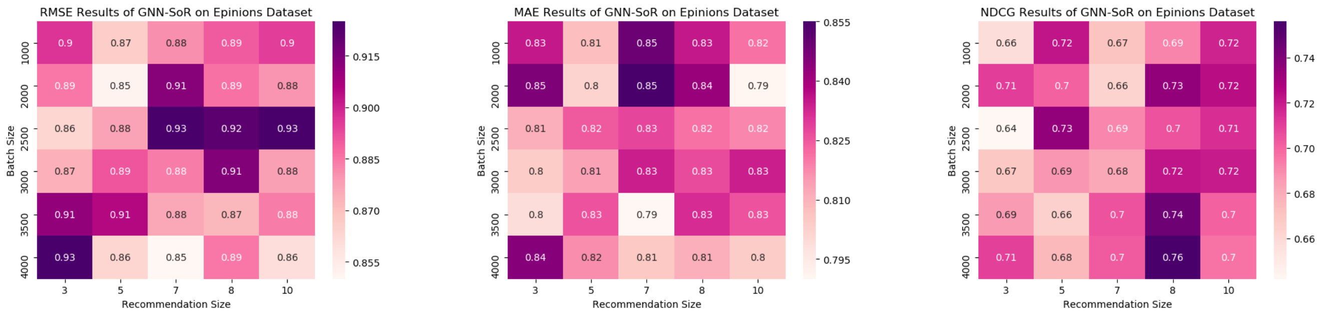
TABLE IV
EXPERIMENTAL RESULTS ON YELP DATASET WITH DIFFERENT PARAMETER SETTINGS

Training	Algorithms	RMSE@5 (rank)	RMSE@10 (rank)	MAE@5 (rank)	MAE@10 (rank)	NDCG@5 (rank)	NDCG@10 (rank)
Yelp (60%)	TrustMF	0.951 (4)	0.977 (3)	0.861 (2)	**0.868 (1)**	0.670 (5)	0.684 (4)
	SocialMF	0.974 (5)	0.996 (5)	0.904 (5)	0.895 (4)	0.699 (2)	0.709 (2)
	TrustSVD	0.937 (2)	0.982 (4)	0.875 (3)	0.910 (5)	0.691 (3)	0.677 (5)
	CUNE	0.945 (3)	0.951 (2)	0.898 (4)	0.883 (3)	0.682 (4)	0.690 (3)
	GNN-SoR	**0.928 (1)**	**0.946 (1)**	**0.859 (1)**	0.872 (2)	**0.705 (1)**	**0.712 (1)**
Yelp (80%)	TrustMF	0.897 (5)	0.829 (2)	0.887 (4)	0.883 (3)	0.644 (2)	0.651 (5)
	SocialMF	0.871 (3)	0.833 (3)	0.893 (5)	0.894 (5)	0.641 (3)	0.679 (3)
	TrustSVD	0.862 (2)	0.851 (5)	0.876 (3)	0.890 (4)	0.629 (5)	0.662 (4)
	CUNE	0.883 (4)	0.844 (4)	**0.858 (1)**	0.879 (2)	0.638 (4)	0.685 (2)
	GNN-SoR	**0.856 (1)**	**0.820 (1)**	0.865 (2)	**0.871 (1)**	**0.654 (1)**	**0.687 (1)**

TABLE V
EXPERIMENTAL RESULTS ON FLIXSTER DATASET WITH DIFFERENT PARAMETER SETTINGS

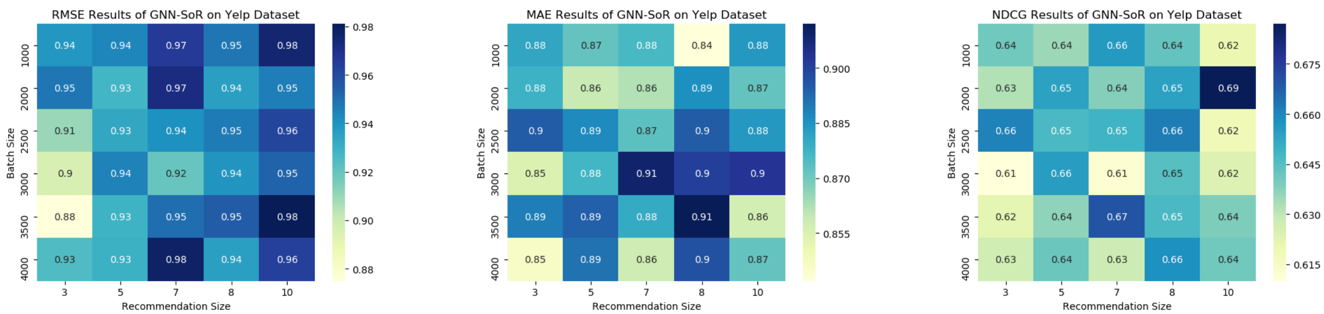
Training	Algorithms	RMSE@5 (rank)	RMSE@10 (rank)	MAE@5 (rank)	MAE@10 (rank)	NDCG@5 (rank)	NDCG@10 (rank)
Flixster (60%)	TrustMF	0.937 (4)	0.913 (5)	**0.866 (1)**	0.892 (5)	0.567 (4)	0.565 (5)
	SocialMF	0.931 (3)	0.898 (3)	0.897 (5)	0.877 (2)	0.545 (5)	0.589 (3)
	TrustSVD	0.944 (5)	0.906 (4)	0.884 (3)	0.884 (3)	**0.592 (1)**	0.574 (4)
	CUNE	0.926 (2)	0.895 (2)	0.892 (4)	0.890 (4)	0.573 (3)	0.598 (2)

	GNN-SoR	**0.910 (1)**	**0.887 (1)**	0.873 (2)	**0.865 (1)**	0.586 (2)	**0.606 (1)**
Flixster (80%)	TrustMF	0.917 (5)	0.880 (4)	0.907 (5)	0.901 (5)	0.559 (4)	0.580 (2)
	SocialMF	0.905 (3)	0.874 (2)	0.895 (4)	0.877 (2)	0.560 (3)	0.576 (3)
	TrustSVD	0.911 (4)	0.896 (5)	0.872 (2)	0.882 (3)	0.546 (5)	0.563 (5)
	CUNE	0.902 (2)	0.879 (3)	0.878 (3)	0.893 (4)	0.567 (2)	0.572 (4)
	GNN-SoR	**0.895 (1)**	**0.863 (1)**	**0.871 (1)**	**0.859 (1)**	**0.571 (1)**	**0.594 (1)**



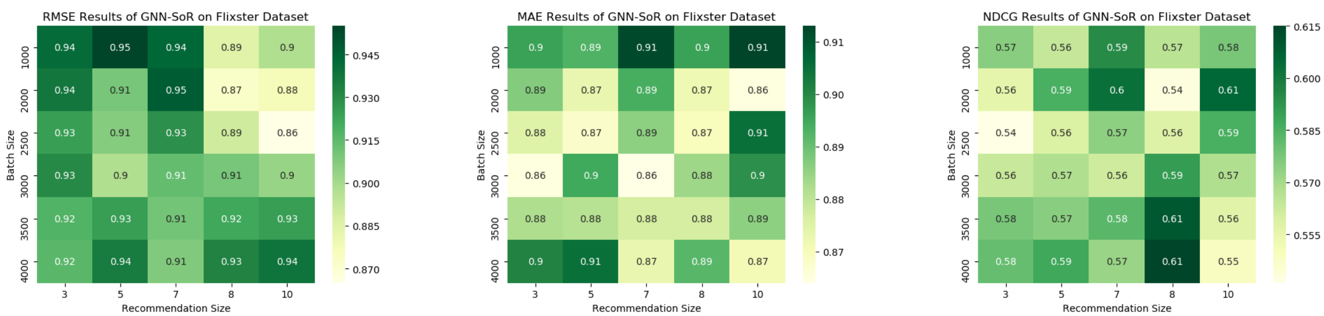
(a) RMSE Results (b) MAE Results (c) NDCG Results

Fig. 4. Experimental Results of Parameter Sensitivity on Epinions Dataset.



(a) RMSE Results (b) MAE Results (c) NDCG Results

Fig. 5. Experimental Results of Parameter Sensitivity on Yelp Dataset.



(a) RMSE Results (b) MAE Results (c) NDCG Results

Fig. 6. Experimental Results of Parameter Sensitivity on Flixster Dataset.

Then, batch size is set to 1000, 2000, 2500, 3000, 3500 and 4000, and recommendation size is set to 3, 5, 7, 8 and 10. We further conduct a set of experiments to evaluate parameter sensitivity performance of the GNN-SoR. Fig. 4 illustrates experimental results of GNN-SoR on Epinions dataset under different value combinations of the two parameters. It contains three subfigures, corresponding to results of RMSE, MAE and NDCG. Similarly, Fig. 5 and Fig. 6 respectively illustrates results of GNN-SoR on Yelp dataset and Flixster dataset with parameters changing. These figures are color-based heatmaps, where depth of colors represents values of indexes. It can be directly observed from these figures that chromaticity difference among squares inside each subfigure is relatively small, reflecting excellent stability of GNN-SoR. This is because the GNN-SoR comprehensively formulates feature space and is robust to different parameter settings.

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.2020.2986316, IEEE Transactions on Industrial Informatics

V. CONCLUSIONS

In the upcoming era of 5G, IoT will become an important part in daily life of people, especially various social and recreational activities. This research manages to investigate application of SoR for future IoT users. In order to model fine-grained features in SoR, this paper exploits correlations of item attributes, and proposes a novel framework GNN-SoR for this purpose. First, user feature space and item feature space are abstracted as two graph networks and respectively encoded via graph neural network method. Next, two encoded spaces are embedded into two latent factors of matrix factorization to complete missing rating values in user-item rating matrix. Finally, a large amount of experiments are conducted on three real-world datasets to verify efficiency and stability of the proposed GNN-SoR.

REFERENCES

[1] M. Jiang *et al.,* "Social recommendation with cross-domain transferable knowledge," *IEEE transactions on knowledge and data engineering*, vol. 27, no.11, pp. 3084-3097, 2016.

[2] H. Ma *et al.*, "Recommender systems with social regularization," in *Proc. of the Fourth ACM International Conference on Web Search and Data Mining*, Hong Kong, China, 2011, pp. 287-296.

[3] X. Yang, H. Steck, Y. Liu, "Circle-based recommendation in online social networks," in *Proc. of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, Beijing, China, 2012, pp. 1267-1275.

[4] W. Yao *et al.,* "Modeling dual role preferences for trust-aware recommendation," in *Proc. of the 37th international ACM SIGIR conference on Research & development in information retrieval*, Gold Coast, Queensland, Australia, 2014, pp. 975-978.

[5] H. Ma *et al.*, "Sorec: social recommendation using probabilistic matrix factorization," in *Proc. of the 17th ACM conference on Information and knowledge management*, Napa Valley, California, USA, 2008, pp. 931-940.

[6] B. Yang *et al.,* "Social collaborative filtering by trust," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no.8, pp. 1633-1647, 2016.

[7] Y. Shen, R. Jin, "Learning personal+ social latent factor model for social recommendation," in *Proc. of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, Beijing, China, 2012, pp. 1303-1311.

[8] M. Jamali, M. Ester, "A matrix factorization technique with trust propagation for recommendation in social networks," in *Proc. of the fourth ACM conference on Recommender systems*, Barcelona, Spain, 2010, pp. 135-142.

[9] H. Ma, I. King, M. R. Lyu, "Learning to recommend with social trust ensemble," in *Proc. of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, Boston, Massachusetts, USA, 2010, pp. 135-142.

[10] J. Tang, H. Gao, H. Liu, "mTrust: discerning multi-faceted trust in a connected world," in *Proc. of the fifth ACM international conference on Web search and data mining*, Seattle, Washington, USA, 2012, pp. 93-102.

[11] A. J. B. Chaney, D. M. Blei, T. Eliassi-Rad, "A probabilistic model for using social networks in personalized item recommendation," in *Proc. of the 9th ACM Conference on Recommender Systems*, Vienna, Austria, 2015, pp. 43-50.

[12] H. Fang, Y. Bao, J. Zhang, "Leveraging decomposed trust in probabilistic matrix factorization for effective recommendation," in *Proc. of the Twenty-Eighth AAAI Conference on Artificial Intelligence*, Québec City, Québec, Canada, 2014, pp. 30-36.

[13] X. Wang *et al.,* "Learning personalized preference of strong and weak ties for social recommendation," in *Proc. of the 26th International Conference on World Wide Web*, Perth, Australia, 2017, pp. 1601-1610.

[14] G. Guo, J. Zhang, N. Yorke-Smith, "TrustSVD: collaborative filtering with both the explicit and implicit influence of user trust and of item ratings," in *Proc. of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, Austin, Texas, USA, 2015, pp. 123-129.

[15] S. Deng *et al.,* "On deep learning for trust-aware recommendations in social networks," *IEEE transactions on neural networks and learning systems*, vol. 28, no.5, pp. 1164-1177, 2016.

[16] Z. Qu *et al.,* "An Efficient Recommendation Framework on Social Media Platforms Based on Deep Learning," in *Proc. of the 2018 IEEE International Conference on Big Data and Smart Computing*, Shanghai, China, 2018, pp. 599-602.

[17] H. Liu *et al.*, "Collaborative social deep learning for celebrity recommendation," *Intelligent Data Analysis*, vol. 22, no.6, pp. 1375-1394, 2018.

[18] L. Xiao, *et al.*, "A Neural Network Model for Social-Aware Recommendation," in *Proc. of the 2017 Asia Information Retrieval Symposium*, Jeju Island, Korea, 2017, pp. 125-137.

[19] O. Tal, Y. Liu, "TCENR: A Hybrid Neural Recommender for Location Based Social Networks," in *Proc. of the 2018 IEEE International Conference on Data Mining Workshops*, Singapore, 2018, pp. 1186-1191.

[20] M. Li, K. Tei, Y. Fukazawa, "An efficient co-Attention Neural Network for Social Recommendation," in *Proc. of the 2019 IEEE/WIC/ACM International Conference on Web Intelligence*, Thessaloniki, Greece, 2019, pp. 34-42.

[21] D. Rafailidis, F. Crestani, "Recommendation with social relationships via deep learning," in *Proc. of the ACM SIGIR International Conference on Theory of Information Retrieval*, Amsterdam, The Netherlands, 2017, pp. 151-158.

[22] W. Yuan *et al.*, "Socialized healthcare service recommendation using deep learning," *Neural Computing and Applications*, vol. 30, no.7, pp. 2071-2082, 2018.

[23] W. Fan *et al.*, "Graph Neural Networks for Social Recommendation," in *Proc. of the 28th World Wide Web Conference*, San Francisco, CA, USA, 2019, pp. 417-426.

[24] S. Wen *et al.*, "Lag synchronization of switched neural networks via neural activation function and applications in image encryption," *IEEE transactions on neural networks and learning systems*, vol. 26, no. 7, pp. 1493-1502, 2015.

[25] M. C. Yang, H. C. Rim, "Identifying interesting Twitter contents using topical analysis," *Expert Systems with Applications*, vol. 41, no. 9, pp. 4330-4336, 2014.

[26] Z. Cui *et al.*, "Dressing as a Whole: Outfit Compatibility Learning Based on Node-wise Graph Neural Networks," in *Proc. of the 28th World Wide Web Conference*, San Francisco, CA, USA, 2019, pp. 307-317.

[27] X. Peng *et al.*, "Connections between nuclear-norm and frobenius-norm-based representations," *IEEE transactions on neural networks and learning systems*, vol. 29, no. 1, pp. 218-224, 2016.

[28] P. Massa, P. Avesani, "Controversial users demand local trust metrics: An experimental study on epinions. com community," in *Proc. of the 20th national conference on Artificial intelligence*, Pittsburgh, Pennsylvania, USA, 2005, pp. 121-126.

[29] S. Hegde, S. Satyappanavar, S. Setty, "Restaurant setup business analysis using yelp dataset," in *Proc. of the 2017 International Conference on Advances in Computing, Communications and Informatics*, Udupi, India, 2017, pp. 2342-2348.

[30] X. Yang *et al.*, "On top-k recommendation using social networks," in *Proc. of the sixth ACM conference on Recommender systems*, Barcelona, Spain, 2010, pp. 67-74.

[31] C. C. Nisha, A. A. Mohan, "social recommender system using deep architecture and network embedding," *Applied Intelligence*, vol. 49, no. 5, pp. 1937-1953, 2019.
[32] J. Chen *et al.*, "Social recommendation based on users' attention and preference," *Neurocomputing*, vol. 341, pp. 1-9, 2019.
[33] C. Zhang *et al.*, "Collaborative User Network Embedding for Social Recommender Systems," in *Proc. of the 2017 SIAM International Conference on Data Mining*, Houston, Texas, USA, 2017, pp. 381-389.

Zhiwei Guo received the B.E. degree from Zhengzhou University, in 2013, and the Ph.D. degree in communication and information system from Chongqing University, in 2018. In 2017, he gave an oral presentation in the International Joint Conference on Artificial Intelligence (IJCAI 2017). Currently, he is an Assistant Professor with Chongqing Technology and Business University. His current research interests include data mining and pattern recognition.

Heng Wang received the B.S. degree from Henan Agricultural University in 2008, and received his M.S. and Ph.D. degrees in communication and information system from Chongqing University in 2012 and 2015, respectively. He is a lecturer with the College of Mechanical and Electrical Engineering of Henan Agricultural University. His current research interests include Internet of Things and green communication.