IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received November 6, 2019, accepted November 24, 2019, date of publication December 10, 2019,
date of current version January 17, 2020.

Digital Object Identifier 10.1109/ACCESS.2019.2958599

Hybrid Deep Neural Networks for Friend Recommendations in Edge Computing Environment

JIBING GONG[1,2,3,4], YI ZHAO[1,2], SHUAI CHEN[1,3], HONGFEI WANG[5], LINFENG DU[5], SHULI WANG[6], MD ZAKIRUL ALAM BHUIYAN[7], HAO PENG[8], AND BOWEN DU[5]

[1]School of Information Science and Engineering, Yanshan University, Qinhuangdao 066004, China
[2]The Key Laboratory for Computer Virtual Technology and System Integration, Yanshan University, Qinhuangdao 066004, China
[3]State Key Laboratory of Mathematical Engineering and Advanced Computing, Wuxi 214000, China
[4]Key Laboratory for Software Engineering of Hebei Province, Yanshan University, Qinhuangdao 066004, China
[5]School of Computer Science and Technology, Beihang University, Beijing 100083, China
[6]Department of Applied Mathematics, Yanshan University, Qinhuangdao 066044, China
[7]Department of Computer and Information Sciences, Fordham University, New York City, NY 10458, USA
[8]School of Cyber Science and Technology, Beihang University, Beijing 100083, China

Corresponding author: Hao Peng (penghao@act.buaa.edu.cn)

This work was supported in part by the National Key Research and Development Program of China under Grant 2018YFB2101003, in part by the NSFC program under Grant 51822802, Grant U1811463, and Grant 51778033, and in part by the China Geological Survey under Grant DD20190637.

ABSTRACT With the rising popularity of social networks and service recommendations, research on new methods of friend recommendation have become a key topic, especially when based on quality-driven resource processing in an edge computing environment. Traditional methods seldom systematically combine static attributes (e.g., interests, geographical locations, and common friends), dynamic behaviors (e.g., liking, making comments, forwarding and @), and network structures (e.g., social ties) to recommend a new friend to a target user. Meanwhile, with the advent of deep learning, it has become more challenging to integrate these features into a deep neural network framework for friend recommendation. For example, how do we optimally make use of these features to form a united framework and what type of deep neural network architecture should be introduced into a novel recommendation method in an edge computing environment? In this paper, we propose DFRec++, a hybrid deep neural network framework combining attribute attention and network embeddings to make social friend recommendations with the help of both interactive semantics and contextual enhancement. More specifically, we first utilize the latent dirichlet allocation (LDA) topic model to generate common interest topics between users and compute the similarity of the explicit static attribute vector representation of topics, locations, and common friends. Then we feed dynamic behavior attributes into a convolutional neural network (CNN) to obtain the implicit vector representation of the interactions and context between two users. Subsequently, a multi-attention mechanism is designed to further improve the deep vector representation of the attribute information. Next, the LINE-based network embeddings algorithm is applied to embed the network structure into a low-dimensional vector. Finally, the attribute attention vector and the network embeddings are concatenated to form a deep feature representation, which is subsequently fed to a fully connected neural network (FCNN) to capture the probability of friendship between two users. The output of FCNN indicates the probability of two users becoming friends. We conducted experiments on a real-world Weibo dataset and the results show that DFRec++ outperforms several existing methods.

INDEX TERMS Friend recommendation, deep neural network, attribute-specific multi-attention mechanism, network embedding, convolutional neural network.

I. INTRODUCTION

In the last few years, social networks, such as for example Weibo and Facebook have been growing exponentially. One of the critical tasks in social networks is friend recommendation. The enthusiasm for research on new methods of friend recommendation and applications thereof has never faded. There are numerous commonly used recommendation techniques, such as collaborative filtering, tag-based recommendation methods, and content-based methods. A typical

The associate editor coordinating the review of this manuscript and approving it for publication was Honghao Gao.

idea is to extract interest representations from social text information and then calculate the similarities of the interest representations. In addition to static information, which includes profiles, the text of participants' posts and their geographical locations, and user interactions such as liking and forwarding are also worth taking into account. It is important for friend recommendation models to learn implicit feature interactions behind user click behaviors. The basic user and interaction information can be recorded as static attributes and dynamic behaviors, which are attribute information obtained from the user's own point of view. In addition, extensive users can constitute a social network of their own due to their numerous interactions. Some researchers extract latent structural patterns from the social network data and then use a ranking algorithm to make friend recommendations. The network structure also contains additional feature information. However, some challenges still exist:

1) The user's attribute information (posts and forwarding) is usually complicated and cumbersome to capture adequately, with high dimensions and strong correlations. The general feature extraction method is relatively shallow. Therefore, the question of how to take advantage of these features to build a unified model still constitutes a substantial challenge.

2) The social network structure contains plenty of important and irreplaceable information. This, however, leaves the question of how to embed the network structure into low-dimensional intensive features for friend recommendations.

3) Users' attribute information is diverse, and the network structure information is heterogeneous. It is a major challenge to combine the advantages of these two aspects to design a holistic friend recommendation framework model.

Social text data conceals users' interests, hobbies, and opinions. Effective topic extraction is therefore a key point for friend recommendation, as users with high topical similarity are more likely to become friends. Nowadays, the LDA algorithm excels at extracting document themes from texts [1]. Some researchers generate users' subjects and interests using the LDA and then recommend friends based on these with satisfying results. Meanwhile, user interaction is an extremely important feature that helps the model better portray the individual user. Dynamic behavior characteristics are high-dimensional, sparse and the processing method is tricky. However, with the excellent performance of neural networks in different fields [2], many new feature extraction methods have emerged, for example: CNN [3], RNN, and so on. At the same time, the concept of ''attention'' from neuro-machine translation has recently gained popularity in training neural networks, as it allows models to learn alignments between different modes.

In this paper, we propose a hybrid neural network framework DFRec++ to combine the advantages of attention and network embedding. First, the users' topics and dynamic behaviors are processed using the LDA and CNN, and then attention is used to represent the users' static attribute information. This framework, however, lacks the characteristic information of social networks. LINE [4] does an exceptional job of extracting network embeddings, therefore the LINE structure is utilized to embed the social network into a low-dimensional feature vector. Finally, the two features are combined, giving a full connection neural network which is designed to calculate the likelihood of two users becoming friends. The gradient descent method is then used to update the parameters of the full connection layer until an empirically optimal model is obtained. The higher the probability of the multi-layer perceptron (MLP) output, the greater the likelihood of the targeted users becoming friends.

The proposed DFRec++ model has several unique innovations as compared with previous methods. First, we consider more aspects of the user's information, including static attributes and dynamic behaviors. Second, we use a convolutional network to learn feature representations rather than manual processing. Third, the attention model guarantees the completeness and depth of the feature extraction, and the embedding of the social network structure ensures the integrity of the features. In this paper, we systematically investigated how to optimally integrate the features representing both static attributes and dynamic behaviors for a good performance in friend recommendation. A brief summary of our work is provided below.

1) To our knowledge, we are the first researchers to systematically combine static attributes, dynamic behaviors, and network structures to make friend recommendations using a deep learning framework.

2) We propose a novel hybrid deep neural network architecture, which consists of CNN and FCNN. CNN is utilized to deal with dynamic behaviors and obtain the network embeddings. We use fully connected neural network (FCNN) to compute the score of the probability of becoming friends between the target user and a candidate user.

3) We provide a novel attribute-specific multi-attention mechanism to improve network embeddings of the original attribute information, and utilize the LINE [4] method to embed social network structure information, as this method preserves both the local and the global network structures.

The rest of this paper is organized as follows. The details of our framework are in Section 2. In Section 3, several experiments are described which are designed to validate our model. In Section 4, related work is briefly outlined. Finally, Section 5 illustrates our conclusions.

We summarized the notations used in this paper in Table 1.

II. OUR APPROACH: DFRec++

Weibo, as one of the biggest platforms with a large number of active users, is a place where users generate a significant amount of information including profiles, geographical locations, new posts, likings, forwarding of posts, and so on. This attribute information can be summarized into two categories: static attributes and dynamic behaviors. The former consists of structural text information and the latter represents the users' interactions habits. If users have great similarities in both static attributes and dynamic behaviors, the probability of them becoming friends as a result is high.

TABLE 1. Notations.

Symbol	Description
Tsim(u,v)	Topic similarity between *u* and *v*
Gsim(u,v)	Geography similarity between *u* and *v*
Cf(u,v)	Common friend similarity between *u* and *v*
F_{static}	feature represents of users' static attributes
$F_{dynamic}$	feature represents of users' dynamic behaviors
F_{link}	concatenate F_{static} and $F_{dynamic}$
$\tilde{W}$	input matrix
A_u	Attention vector
$\tilde{x}_u$	Final attention vector
$\vec{\mu}_u$	Network embeddings vector
θ	all hyper parameters
ϕ_t	time decay function
$Fd(u,v)$	similarity of forwarding between *u* and *v*
$Cm(u,v)$	similarity of common friends between *u* and *v*
$Le(u,v)$	similarity of liking between *u* and *v*
$At(u,v)$	similarity of @ between *u* and *v*

In addition to the attribute information, the behavioral relationships between users can constitute a social network and this network structure is equally rich in information. Embedding network structure information into a new feature helps us to improve friend recommendations. Several different methods are used to extract features from attribute and structure information. First, we use the powerful attention model as the feature extractor to deal with attribute information. In detail, we define and analyze the static attribute information using methods like LDA and then use CNN to conduct a preliminary analysis of the dynamic behavior data. Second, in terms of network structure, we adopt the LINE method to learn the embedded representations of the social network structure information. Finally, we condense these features into a vector which serves as the input of a dense neural network. The output of the network represents the probability of target users becoming friends with other users. It is easy to choose the top users from the generated ranking for friend recommendations. The entire architecture of the model is given in Figure 1. In the following sections, we first describe how attribute attention works and then give the details of network embedding.

A. STATIC ATTRIBUTES

In this study, we first use the static attributes, which are topics, geographical location, and common friends. After calculating the similarity of each of these attributes for two users, we use these three values to form the vector F_{static}. These features are explained in detail below.

1) TOPICS

Common topics implicitly reflect similar interests among users and can be found using the LDA. The more similar the topics of two users, the more likely they are to become friends. For example, if one of the hobbies of a user is traveling, he or she might prefer to become friends with another travel enthusiast. Similarly, two users working in the same industry may want to talk about their professional field. People are more inclined to talk with their peers, rather than people they have little or nothing in common with.

2) GEOGRAPHICAL LOCATION

Geographical information reflects a user's spatial characteristics. While social networks have lowered some geographical restrictions, studies still show that two people who live in the same city have a higher probability of becoming friends than those who do not. Additionally, people have a tendency to share amusing things which happen around their location, which users in the same city may find more interesting or have also experienced personally.

3) COMMON FRIENDS

Structural balance theory explains social networks well. In this theory, interests, ages, and other features are transitive among friends, which means that the friends of a user's friends are also potential friends for the user himself/herself. Normally, users who have more mutual friends, are more likely to establish a true friendship.

Social text data implies users' interests, hobbies, and world views. In this study, we use the LDA to discover semantic information with the algorithm shown in Figure 2. Using the LDA algorithm, we can infer a user's interests, age, social status, and other characteristics. More similarities in this data indicate that two users may have more of the same interests and therefore their probability of becoming friends are higher.

In Figure 2, m denotes a document with N_m words, M is the total number of documents, n denotes a word in document m, k stands for a specific topic, and K is the number of topics. The detailed steps of the LDA algorithm are given as follows:

Step 1. $\overrightarrow{\alpha} \rightarrow \overrightarrow{\theta_m} \rightarrow z_{m,n}$: According to the Dirichlet prior parameter $\overrightarrow{\alpha}$, a multinomial topic distribution $\overrightarrow{\theta_m}$ corresponding to document m is generated. Then, topic $z_{m,n}$ is selected for word n $\overrightarrow{\theta_m}$.

Step 2. $\overrightarrow{\beta} \rightarrow \overrightarrow{\varphi_k} \rightarrow w_{m,n}|k = z_{m,n}$: Word multinomial distribution $\overrightarrow{\varphi_k}$ is obtained from Dirichlet prior parameter $\overrightarrow{\beta}$ and topic $z_{m,n}(k)$. Furthermore, word n is selected from multinomial distribution $\overrightarrow{\varphi_k}$.

Here, $\overrightarrow{w} = (\overrightarrow{w_1}, \ldots, \overrightarrow{w_M})$ denotes the corpus, and the words in document m form the vector $\overrightarrow{w_m}$. In addition, $\overrightarrow{z} = (\overrightarrow{z_1}, \ldots, \overrightarrow{z_M})$ represents the topic matrix corresponding to $\overrightarrow{w}$, where each word only corresponds to one topic. The process of generating each topic is independent, therefore we can calculate the probability of the topic in the corpus using 1.

$$p(\vec{z} \mid \vec{\alpha}) = \prod_{m=1}^{M} p(\vec{z_m} \mid \vec{\alpha}) \tag{1}$$

Words with the same topic k are clustered as $\overrightarrow{w_{(k)}}$, $\overrightarrow{w}' = (\overrightarrow{w_{(1)}}, \ldots, \overrightarrow{w_{(K)}})$ denotes all words that have been classified. In addition, $\overrightarrow{z}' = (\overrightarrow{z_{(1)}}, \ldots, \overrightarrow{z_{(K)}})$ is the topic matrix corresponding to $\overrightarrow{w}'$, where each component in $\overrightarrow{z_{(k)}}$ is K. The process of generating words is independent and it can be

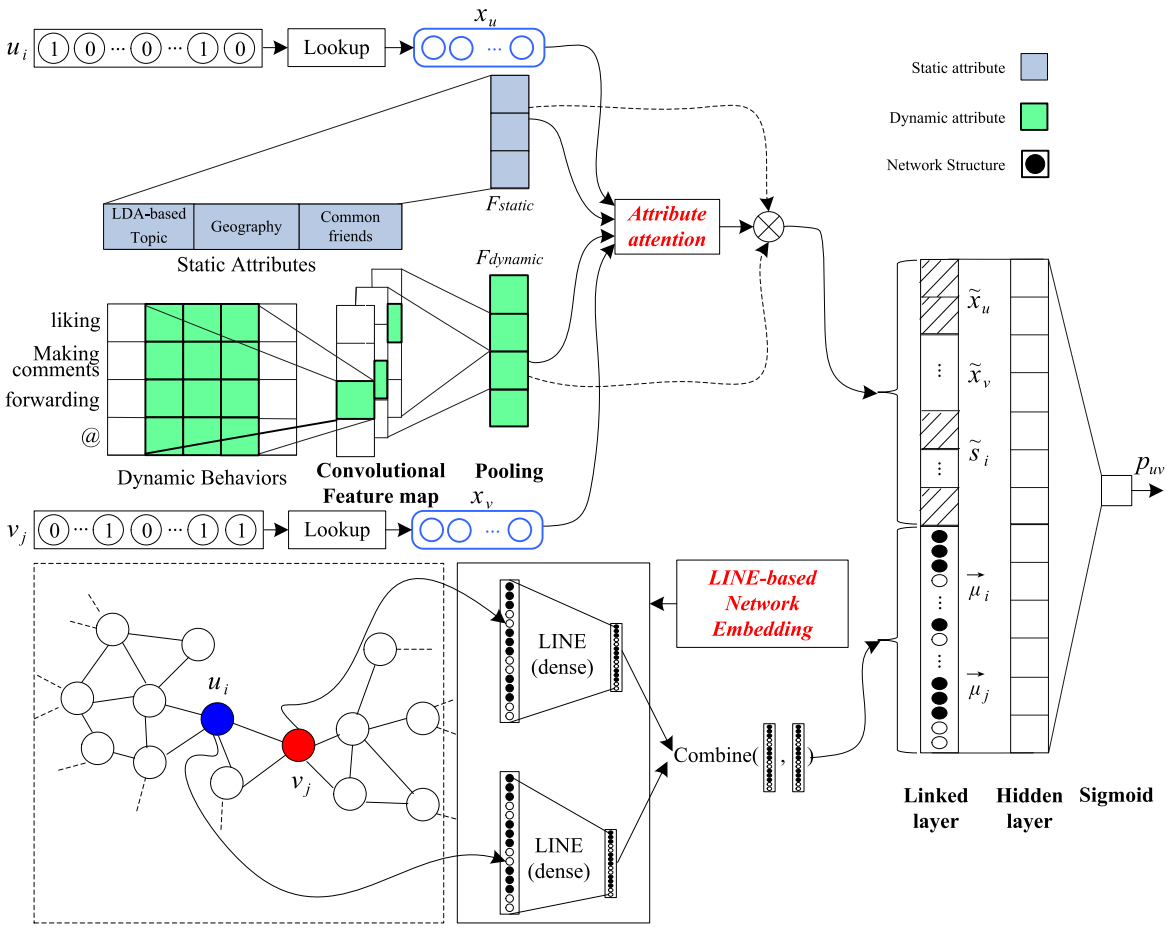


FIGURE 1. DFRec++ Architecture: The powerful attention mode is used as the feature extractor to deal with attribute information. LDA and CNN is used to analyze the static attribute information and dynamic behavior data in detail. The LINE method is adopted to learn the embedding representation of the social network structure information. Then, these features are transferred into a vector, which serves as the input for a dense neural network. The output of the network represents the probability of targeted users becoming friends with other users.

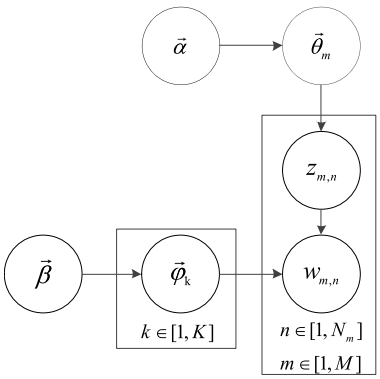


FIGURE 2. LDA principle.

expressed by Eq. 2.

$$p\left(\vec{w} \mid \vec{z}, \vec{\beta}\right) = \prod_{k=1}^{K} p\left(w_{(k)}^{\vec{}} \mid \vec{z_k}, \vec{\beta}\right) \tag{2}$$

Using Eqs. 1 and 2, we can deduce the joint distribution probability.

$$p\left(\vec{w}, \vec{z} \mid \vec{\alpha}, \vec{\beta}\right) = p\left(\vec{w} \mid \vec{z}, \vec{\beta}\right) p\left(\vec{z} \mid \vec{\alpha}\right) \tag{3}$$

Gibbs sampling is used to calculate the parameters that can subsequently be used to predict the topics of the users' posts.

A representation of the users' interests can be obtained in the manner described above. After that, we can calculate the similarity of several users. Topic similarity $Tsim(u, v)$ between u and v is defined as follows:

$$Tsim\left(u, v\right) = \frac{\sum_T Tu \bigcap Tv}{\sum_T Tu \bigcup Tv} \tag{4}$$

Here, T is the set of topics and Tu is the set of posts user u interacted with. In this paper, when a user interacts with a post, it means that the post is the user's original post, or it has been associated with the user through dynamic behaviors. The numerator is the number of common topics about which u and v have interacted and the denominator represents all posts with which u and v interact.

Simultaneously, we calculate the geographical location similarity $Gsim(u, v)$. The equation is expressed as follows:

$$Gsim\,(u, v) = \frac{1}{1 + log(dist\,(l_u, l_v)) + 1} \tag{5}$$

Here, l_u and l_v represent user u and user v's customary log-in location, respectively.

In a social network, people can make friends with any stranger. Usually, a stranger is at most separated by six friends based on the Six Degrees of Separation. The more mutual friends two users have, the more likely they are to become friends themselves. To determine $Cf(u, v)$, the similarity of mutual friends, we simply determine the number of mutual friends of two users and calculate their similarity as follows:

$$Cf\ (u, v) = \frac{Fu \bigcap Fv}{Fu \bigcup Fv} \tag{6}$$

Here, Fu and Fv are the friend sets of users u and v, respectively. The numerator determines the number of mutual friends of the two users, and the denominator is the size of the intersection of their friend sets.

After calculating the similarity of the static attributes of two users, three similarity values can be combined into a vector F_{static}:

$$F_{static} = [T_{sim}; G_{sim}; Cf_{sim}] \tag{7}$$

This vector will be integrated with the vector containing the information of the dynamic behaviors, and together they will serve as the input for the deep neural network. We will discuss dynamic behavior in detail in later chapters.

B. DYNAMIC BEHAVIOR

Interactions between users help form a link structure among users in a social network, so we define dynamic behaviors as interactions. Forwarding or liking posts, mentioning other users, and writing or replying to comments are taken into consideration as well. We use CNN to map users' dynamic behavior features. The details are presented below.

Weibo posts which users participate in, can be divided into two categories: posts sent by themselves and posts sent by other users. For every user, if a second user is interacting with their Weibo posts, they are more likely to become friends than when they are involved in a third user's Weibo posting. Based on this, we built an input matrix $w \in R^{d \times M}$. M denotes all the Weibo posts that the target user u participated in and d represents four dynamic behaviors: liking, forwarding, making, or replying to comments, and mentioning other users. After analyzing the dynamic behaviors of the candidate user c, the value of the matrix is calculated as follows:

$$\tilde{W}_{i,j} = \begin{cases} 2k & \text{both user } u, c \text{ done} \\ k & \text{either } u, c \text{ done} \\ 0 & \text{none } u, c \text{ done} \end{cases} \tag{8}$$

i is one of four possible dynamic behaviors, and j stands for a specific Weibo post. k is a constant with two categories: it is set to 0.75 when the post is sent by the user u, and otherwise to 0.5. After these steps, the generated input matrix $\tilde{W}$ is prepared to be introduced to the convolutional network.

C. ATTRIBUTE ATTENTION

1) STATIC ATTRIBUTE ATTENTION

Given a target user u_i and a candidate user v_j, the static attributes connecting them provide the important context of their common interests, which are likely to affect the original representations of both users. Giving the original user latent embeddings x_u and x_v, and the static attribute based context embedding $e_{u\to v}$ for the interests between u and v, we use a single layer network to compute the attention vectors A_u and A_v for users u and v as follows:

$$\begin{aligned} A_u &= f(W_u^{(1)} x_u + w_{u\to v} e_{u\to v} + b_u) \\ A_v &= f(W_v' x_v + w_{v\to u} e_{u\to v} + b_v') \end{aligned} \tag{9}$$

where w_* and b_u denote the weight matrix and bias vector for the user $u \to v$ attention layer, w_*' and b_u' denote the weight matrix and bias vector for the user $u \to u$ attention layer. Similarly, $f()$ is set to be the ReLU. Then, the final representations of the users u and v are computed using an Hadamard product '$\odot$' of the attention vectors:

$$\begin{aligned} \tilde{x_u} &= A_u \odot x_u \\ \tilde{x_v} &= A_v \odot x_v \end{aligned} \tag{10}$$

The attention vectors A_u and A_v are used for improving the original user embedding and are conditioned on the calibrated static attribute-based context.

By combining the two parts of the attention components, our model improves the original representations for users in semantics and contextual enhancement ways. We call such an attention mechanism Static-Attribute-Attention. To the best of our knowledge, few social friend recommendation methods are able to learn explicit representations in this way.

2) DYNAMIC BEHAVIOR ATTENTION

Now that the static-attribute-attention mechanism has been implemented, we further try to finish the dynamic-behavior-attention (also, dynamic-attribute-attention) mechanism according to these observations: 1) the improved effectiveness of a model depends on the flavor which each attention mechanism gives, 2) attention is employed to alter the original representation either by re-weighting or realigning, 3) most neural network architectures only leverage one type of attention or alignment function, and 4) most related work uses each attention operation to aim at a different type of user feature [5]. Thus, our work achieves dynamic-behavior-attention through casting multi-attention with dynamic behaviors, which include liking, making comments, forwarding and @ operations, respectively.

Let s be the original vector of the four dynamic attributes described above. And $\hat{s}$ is the representation of s after applying multi-attention. The attention features for the multi-

attention operators are:

$$\begin{aligned} g_1 &= G([\tilde{s} \leftarrow s]) \\ g_2 &= G(\tilde{s} \cdot s) \\ g_3 &= G(\tilde{s} - s) \end{aligned} \tag{11}$$

where $[\leftarrow]$ is the concatenation operator and $\cdot$ is the Hadamard product. $G()$ is a **compression function** used to reduce the features to a scalar. Here, s and $\tilde{s}$ have interactive semantics. Intuitively, we do not hope that subsequent layers will be expanded with a high dimensional vector which consequently leads to parameter costs in subsequent layers.

We use CNN to extract latent structural patterns of dynamic behavior similarity among users. The input matrix $\tilde{W}$ is derived from the description above and served as the input to a CNN. As described in Fig. 1, we apply filters $f^{d\times m}$ to the input matrix $\tilde{W}$. The operation is as follows:

$$C_i = tanh(\tilde{W}^T * f_i + bias_i) \tag{12}$$

where $\tilde{W}^T$ denotes the transpose of the input matrix $\tilde{W}$, which is the result of Eq 8. f_i, $bias_i$ stands for the i-th filter and bias, respectively. Finally, c_i is the result of this calculation, i-th are the convolutional feature maps. In addition, a padding technique is adopted to better handle local features at boundaries while giving the same attention to the center matrix and to keep dimensions consistent.

We want the feature representation to learn to be non-linear, such that activation function is utilized to process the convolutional result. The most common choices of activation functions are the following functions: the sigmoid function $\alpha(x)$, the hyperbolic tangent function $tanh(x)$, the rectified linear unit function $ReLU(x)$ and the variants $leak\ ReLU$. We select $tanh$ as the activation function in our CNN network to ensure non-linear features are learned.

After the convolutional layer, it passes through the pooling layer which aggregates the features and further reduces the representation. In detail, we apply maximum pooling instead of the average pooling for each convolutional feature map. Then we concatenate the short vectors of the pooling layers output as $F_{dynamic}$. The operation is:

$$F_{dynamic} = \begin{bmatrix} pool(c_1) \\ \dots \\ pool(c_2) \end{bmatrix} \tag{13}$$

Finally, we finished the Multi-Attention mechanism for dynamic-behavior-attention. According to the characteristics of different behaviors, for each target-candidate user pair, we apply (1) Co-Attention with max pooling (2) Co-Attention with alignment pooling and (3) Co-Attention with mean pooling, respectively. Additionally, we apply Intra-Attention to both user and behavior content, individually. Each attention produces the same scalars ($k = 4$) which are concatenated with the s embedding. The final cast feature vector is $z \in \mathbb{R}^{16}$. As such, for each behavior vector s_i, the new dynamic attribute representation becomes $\tilde{s_i} = [s_i; z_i]$

D. NETWORK EMBEDDING

According to LINE [4], our model preserves the first-order proximity and the second-order proximity separately, and then introduces a simple way to combine the two proximities. We define the joint probability between user u_i and v_j as follows:

$$p1 = (U_i, v_j) = \frac{1}{1 + \exp(-\vec{\mu}_i^T \cdot \vec{\mu}_j)} \tag{14}$$

where $\mu_i \in \mathbb{R}^d$ is the low-dimensional vector representation of user u_i. p1 is equivalent to describe the intimate degree of points from the perspective of embedding. The distribution $p()$ spans over the space $V \times V$. Considering that Weibo Social Network is based on undirected graphs, to preserve the first-order proximity, we have:

$$O_1 = d(\hat{p}1(,), p1(,)) \tag{15}$$

where $\hat{p}1$ is the empirical probability of $p(,)$ and can be defined as $\hat{p}1(v_i, v_j) = \frac{\omega_{ij}}{W}$, where $w = \sum_{(i,j)\in\mathbb{E}} \omega_{i,j}$. $d(,)$ is the distance between two distributions and $W_{i,j}$ represents the weight of the edge between the points i and j.

The second-order proximity indicates that the more connections two users share, the more similar the two users are. Thus, to further preserve the second-order proximity, we use:

$$O_2 = -\sum_{(i,j)\in\mathbb{E}} \omega_{i,j} \log p2(v_j|v_i) \tag{16}$$

where $p2(v_j|v_i)$ is the probability of ''context'' u_j generated by vertex v_i and is defined as:

$$p2(v_j|v_i) = \frac{\exp(\vec{\mu}_j^T \vec{\mu}_i)}{\sum_{k=1}^{|V|} \vec{\mu}_j'^T \vec{\mu}_i} \tag{17}$$

where V is the number of users or ''contexts''. To minimize the difference between distribution p1 and p2, O_2 needs to be optimized. By learning that $\{\vec{\mu}_j\}_{i==1|V|}$ and $\{\vec{\mu}_j'\}_{i==1|V|}$ minimize this objective, we are able to represent every vertex v_i with a d-dimensional vector $\vec{\mu}_i$.

E. FULLY CONNECTED NETWORK

Thus far, given a common topic of interest between user u and user v, we obtained the embeddings for user u and v. We combine the three embedding vectors into a unified representation of the current context and use interactive semantics as below:

$$X_{u,v} = \tilde{x}_u \parallel \tilde{x}_v \parallel \tilde{s}_i \parallel \vec{\mu}_i \parallel \vec{\mu}_j \tag{18}$$

where $\parallel$ denotes the vector concatenation operator, $\tilde{x}_u$ and $\tilde{x}_u$ (Eq. 10) denote the improved embeddings of the target user u and the candidate user v over the static attributes, respectively. And $\tilde{s}_i$ denotes the new dynamic attribute representation, $\vec{\mu}_i$ and $\vec{\mu}_j$ denote the network embeddings of the target user u and the candidate user v, respectively. $X_{u,v}$ encodes the information on common interests from three aspects: the involved static attributes, the involved dynamic behaviors and the network structure information based on user interactions.

Following [6], we feed $X_{u,v}$ into a MLP component:

$$\hat{d}_{u,v} = \text{MLP}(X_{u,v}) \tag{19}$$

where the MLP component is implemented with hidden layers with ReLU as the activation function and an output layer with the Sigmoid function. The larger the value of the MLP output, the greater the probability that the two users will become friends. Furthermore, we learn the parameters of our model with negative sampling and the objective for a common interest $< u, v >$ can be formulated as follows:

$$\ell_{u,v} = -\log \hat{d}_{u,v} - E_{j \sim P_{fu}}[\log(1 - \hat{d}_{u,v})] \tag{20}$$

where the first term models the observed common interests and interactions, and the second term models the negative feedback drawn from the noise distribution P_{fu}. In short, from another perspective, the DFRec++ framework in Figure 1 that we aim to infer, can be structured as a hybrid neural network, where $F = [f_1, f_2, \ldots, f_{|y|}] \in \mathbb{R}^{dvx|V|}$ and $M = [\mu_1, \mu_2, \ldots, \mu_{|\mathrm{R}|}] \in \mathbb{R}^{d^{\epsilon} \times |\mathrm{R}|}$. Each pair of users gets their respective embeddings through the dense layer. Considering that ReLU function is easy to calculate, we selected it as the hidden layer activation function of MLP. Hidden layers can help the network learn complex decision functions. [7] As an illustration, deep neural networks are learning to extract patterns from the data, rather than memorizing patterns. To prevent overfitting, a variety of regularization techniques can be used. Dropout prevents feature co-adaptation by setting (dropping out) a portion of hidden units to zero during the forward phase, when the activation function at the Sigmoid output layer is computed. We summarize all the steps of this algorithm and show it in **Algorithm 1**.

Algorithm 1 DFRec++ Algorithm

Initialize: Random initial θ with Gaussian, initial Network Embedding with LINE

Requires: Hyper-parameters like learning rate λ, dropout-rate

1: $u \Leftarrow$ Target user
2: **for** Epoch **do**
3: Select condidate user v
4: Calculate F_{static} (Eq. 4,5,6,7)
5: Static Attribute Attention $\tilde{x}_u$, $\tilde{x}_u$: (Eq. 9,10)
6: $\tilde{W} \Leftarrow$ The input matrix (Eq. 8);
7: Calculate $F_{dynamic}$ (Eq. 11,12,13)
8: Dynamic behavior Attention $\tilde{s}_i$
9: Network Embedding $\tilde{\mu}_u$, $\tilde{\mu}_v$: (Eq. 14,15,16,17)
10: Calculate $X_{u,v}$ (Eq. 18)
11: Forward propagation
12: Update DFRec++ with AdaDelta
13: **end for**
14: $P_{u,v} \Leftarrow$ Probability that users u and v become friends;

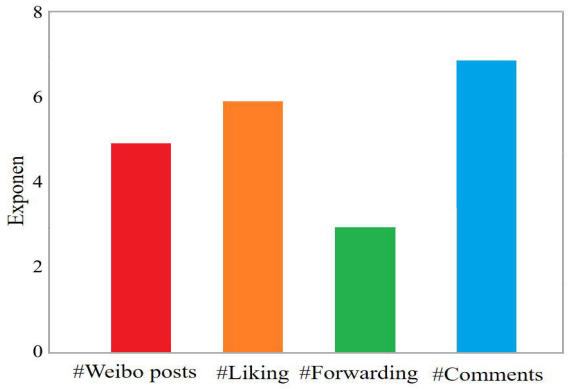


FIGURE 3. Statistics of user behaviors in the dataset.

III. EXPERIMENTS AND EVALUATIONS

A. EXPERIMENTAL SETUP

In this section, we compare the proposed DFRec++ model with several other friend recommendation methods. The experimental dataset consists of the information from 1,800,000 Weibo users, including their profiles, posting data, links, forwards, and comments. The data is complex and used to support the validity of these model methods. The statistics of the user behaviors in this dataset are illustrated in Figure 3.

We first removed all data from users with too low activities from the dataset, which added up to more than 7% of the data. Then we selected 10% of the data as the verification set and the test set, respectively. We implemented our model using the Python library of TensorFlow. We adopted a mini-batch gradient descent to minimize the loss function with negative sampling, where each mini-batch contained sampled edges. The AdaDelta algorithm is the algorithm we used to update trainable parameters. We also used the user embeddings which were pretrained by the homogeneous network embedding algorithm LINE [4] to initialize the user embeddings into our method. All the experiments were conducted on a machine with two GPUs (NVIDIA GTX-1080Ti * 2) and two CPUs (Intel Xeon E5-2690 * 2). We adopted several metrics (P@k, Recall, MAP, AUC, and F1-Measure) to evaluate the performance of DFRec++ from different perspectives. Equations and description of these metrics are given as follows:

1) P@k

Indicates the precision rates of the first k results that the system generated in response to input query keywords.

$$P@k = \frac{the\ number\ in\ the\ first\ k\ results}{k} \tag{21}$$

2) RECALL

Indicates the recall rates of the first k results that the system generated in response to input query keywords; it is defined as formula 22.

$$Recall = \frac{number\ of\ recommendation\ cases}{the\ number\ of\ newly\ added\ friends}. \tag{22}$$

3) AUC

Area Under the relative operating Characteristic (AUC) is a metric for binary classification. The value of AUC ranges from 0.5 to 1, with high values indicating good performance.

4) MAP

Denotes the average accuracy value that corresponds to the query keywords of each user. Specifically, given an existing query keyword, the average value of the precision rates will be computed according to the precision rates of the first k results. Namely, MAP is the average value of precision of the entire test dataset.

$$MAP = \frac{\sum_{kisrelevant} p@k}{times\ of\ \ recommendation\ cases}. \tag{23}$$

5) F1-MEASURE

The weighted average of P@k and Recall; it is defined as formula 24.

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{24}$$

B. OBSERVATIONS

Before proposing DFRec++, we first investigated whether different types of dynamic behaviors have the same influence on friend recommendations, as the motivation of our work is to discover the underlying factors that affect friend recommendations. Here, we analyze the influences of different dynamic behaviors on friend recommendation performance. Dynamic behaviors include liking something, making comments, forwarding, and @. We also see replies as types of comments.

We define the term expectation index to indicate which influences all kinds of dynamic behaviors have. The expectation index refers to the probability that different dynamic behaviors influence the outcome of friend recommendations. In detail, to compute this index, we first randomly select user t from the target user set T. We separate the dynamic behaviors of target user t from the set $C_i (i = 1, 2, 3, 4)$ and randomly select 100 additional users from the set C_i for recommendations to user t. And then we count the number N_i of users in the set C_i who are friends of the target user t. Finally, we calculate the ratio of N_i to C_i to get the expectation index α, which is then calculated according to the following formula:

$$\alpha = \frac{N_i}{M} \quad (i = 1, 2, 3, 4\ M = 100) \tag{25}$$

Then, we repeat the steps described above k times (here we take to be $k = 10$) and calculate the average expectation index as follows:

$$\overline{\alpha} = \frac{\sum_{k=1}^{10} \alpha_k}{N} (N = 10) \tag{26}$$

Finally, the mean expectation indexes of each of the dynamic behaviors are shown in Figure 4:

It can be seen from Figure 4 that the order of the influences of these four behaviors on friend recommendation performance from big to small are: making comments, @, forwarding and clicking a like. Through further analysis, it becomes obvious that making comments is an interaction between users, and @ is the behavior in which one user has the intent to interact. Forwarding and clicking a like are unilateral behaviors of one user. Accordingly, the expectation index for making comments is the largest, followed by @, then forwarding, and finally clicking a like. This is consistent with the actual situation, which means it is consistent with the results of [12]. Interaction is the most important factor affecting the relationship between friends.

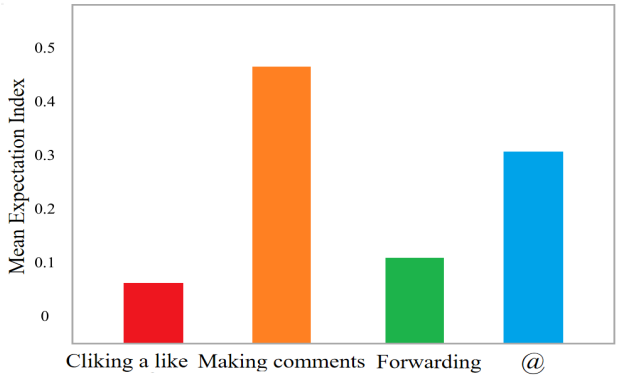


FIGURE 4. The mean expectation index of each dynamic behavior.

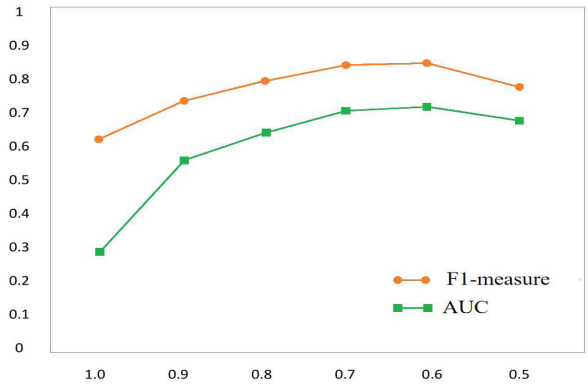


FIGURE 5. AUC and F1-measure comparison of dropout.

C. PARAMETERS SETTINGS

1) DROPOUT

Dropout, a technique for addressing overfitting, refers to the probability that a neuron is kept in the network. Dropout is a regularization technique to compromise the precision and the complexity of the neural network and thereby reducing overfitting. We set the dropout to be 1.0, 0.9, 0.8, 0.7, 0.6, and 0.5. This is a common and effective way to choose the parameter. As visible in the results shown in Figure 5, DFRec++ is able to reach its best performance when the dropout is 0.6, according to the metrics of the AUC and F1-Measure. The result shows that adding reasonable randomness to the model can strengthen the robustness of DFRec++.

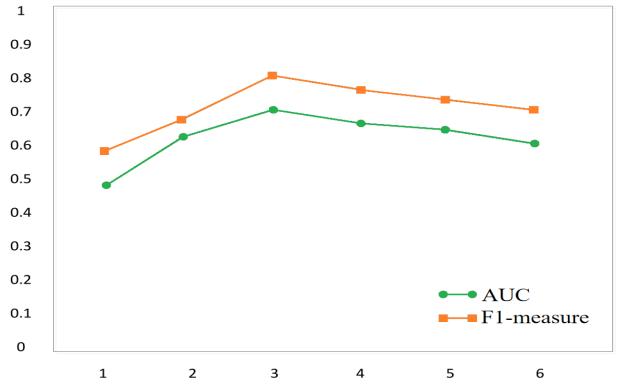


FIGURE 6. AUC and F1-measure comparison of number of hidden layers.

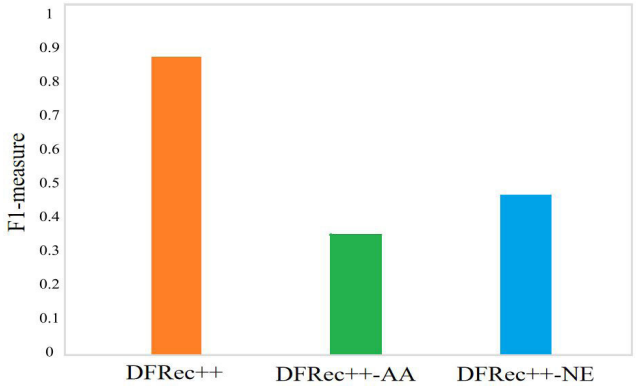


FIGURE 7. A comparison of the values of the of the F1-Measures of DFRec++, DFRec++-AA and DFRec++-NE.

2) NUMBER OF HIDDEN LAYERS

The number of hidden layers also affects the performance of the model to some extent. The number of hidden layers in the neural network affects the learning ability of the model. We chose the number of hidden layers from 1-6 implying that the deeper neural network requires more training time. As presented in Figure 6, the bigger the number of hidden layers, the better the DFRec++ performs. However, the performance deteriorates if the number of hidden layers keeps increasing after the performance has reached the highest value. This phenomenon is due to overfitting.

D. FACTOR ANALYSIS

In DFRec++, we consider the input features according to two factors: attribute attention (AA) and network embedding (NE). We designed two more experiments with different variations to compare the influences of both factors. In detail, we keep the MLP part of the DFRec++ model and then build two additional models, one with AA and one with NE. The method without the attribute attention part is denoted as DFRec++-AA and DFRec++-NE denotes the model without the network embedding part. All three models are trained on the same data set. Figure 7 compares the performance of the three different models in terms of the F1-measure.

It can be seen that the originally proposed model performs better than both of the comparison models. Neither DFRec++-AA nor DFRec++-NE perform satisfactorily. The F1-measures of DFRec++-AA and DFRec++-NE decrease sharply to 0.34 and 0.46, respectively. Although the F1-measure of the DFRec++-NE model is slightly better than that of the DFRec++-AA model, it's overall performance is still poor.

These experimental results reveal the advantage of our DFRec++ model. They indicate that the proposed DFRec++ model improves the accuracy and the precision of friend recommendations. By combining attribute attention and network embedding, the performance increased by 53% and 41% as compared with DFRec++-AA and DFRec++-NE, respectively. The experiment verifies that the AA mechanism and the network embedding are effective in deepening the feature extraction of users from a side perspective. Using only one of these features is a one-sided comparison for friend recommendations. The overall model architecture design maximizes the role of various forms of features, thereby greatly improving the effectiveness of friend recommendations.

E. PERFORMANCE AND COMPARISON

We compared DFRec++ with several other existing friend recommendation methods. We selected the RW [9],common neighbors (CN) [10], LDA-based similarity (LDAS) [11], Friend++ [12], RWCFR [13], CVAE [14], TBDM [15], DT [16] as comparison algorithms. A brief description of these methods is given below.

1) RW

This method utilizes an individual intimacy feature to compute the similarity between target users and the candidate users through a RW and then obtains the top-N recommended friends.

2) CN

This method finds and measures the mutual friends of the target user and his or her second-degree friends.

3) LDAS

Specifically, this LDA-based method only considers the topic attributes and ignores other information when modeling the recommendations. Then, according to the topic similarity, it recommends friends to target users.

4) Friend++

Two features (network and node features) are considered in the Friend++ method. It employs a RW algorithm to recommend friends.

5) RWCFR

In addition to check-in data, we consider location information and the user's profile data to obtain a ranked recommendation list.

TABLE 2. Comparative evaluation results.

Method	P@5	P@10	P@15	Recall
RW	0.753	0.442	0.281	0.621
CN	0.777	0.527	0.207	0.584
LDAS	0.671	0.421	0.243	0.836
Friend++	0.782	0.518	0.268	0.845
RWCFR	0.530	0.291	0.199	0.364
CVAE	0.786	0.538	0.312	0.927
TBDM	0.791	0.370	0.219	0.813
DT	0.719	0.330	0.188	0.754
DFRec++	0.813	0.603	0.307	0.921

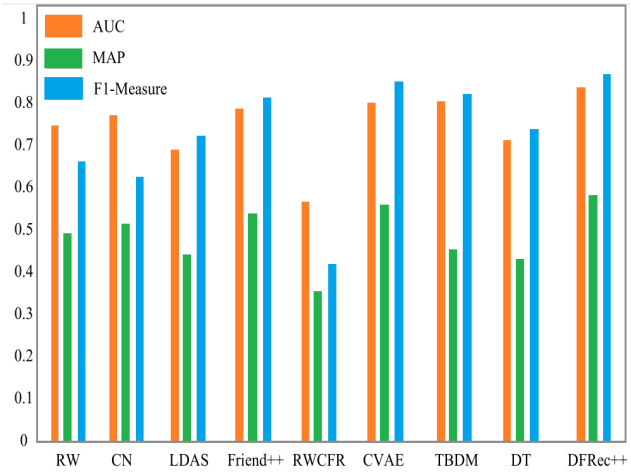


FIGURE 8. Experimental results.

6) CVAE

This is a Bayesian probabilistic generative model that unifies the collaborative and content information through deep learning models.

7) TBDM

It mainly extracts the user interest features from large corpora and rarely considers interactions.

8) DT

This is a friend recommendation algorithm based on the Decision Tree with AdaBoost.

We implemented the methods described above on our dataset and utilized several metrics to evaluate the performance of DFRec++ with respect to the other methods. The experimental results are listed in Table 2.

Table 2 lists the P@k results of the eight baseline methods and the method proposed in this study on the same dataset. The results clearly show that our proposed DFRec++ model outperforms all eight baseline algorithms in terms of P@5 and P@10. The performance on the recall indicator is very close to the best one. Furthermore, we compare the recall, MAP, and F1-measure of DFRec++ with those of the eight baseline methods in Figure 8.

Figure 8 clearly shows that DFRec++ outperforms the eight methods we compared it with, with respect to all three metrics: AUC, MAP, and F1-measure. Overall, the RWCFR method performs the worst, mainly because it is more suitable for mobile social data. The performance of RW, CN, LDAS and DT is similar, but lower than that of the Friend++ method. The gap between MAP is quite large. These methods are not full deep learning methods and they also consider fewer features than DFRec++. Compared with our method, the difference between the AUC and the F1 values of the TBDM method is small, however its performance on the P@5 and P@10 indicators, which lead to its MAP value, is very low. This indicates that the stability of this method is poor. The recall and P@15 values of CVAE are slightly better than that of the DFRec++ method, but in terms of the AUC and the F1 values, our method is still better by 1%. Overall, considering the advantages of richer features and deeper and more efficient extraction, our model achieved excellent performance in the comparative experiments.

F. DISCUSSION

In this section, we describe the design of a variant method of DFRec++ to further illustrate the effects of the answers to the following questions: how the attention and network embedding mapping layer affect the representation.

FRec++: We investigate the effects of the feature extractor on DFRec++ by comparison to FRec++ [17]. The idea of the comparison is to use the manual features instead of the neural network features extractor. DFRec++ consists of three parts: an attention model extracting attribute information, a LINE-based network embedding, an MLP input vector concatenating attribute features, and an embedding feature. We remove the network embedding part and keep the MLP. Instead of using attention as the feature extractor for the attribute information, we use fully manual features. The input of FRec++ is a vector that combines static attributes calculated in the same way as in DFRec++ and dynamic behavior features. The representation of the dynamic behavior feature in FRec++ is a vector consisting of four values: Li, Fd, Cm, At. The equations of similarities in forwarding behavior are determined by Eq.27:

$$Fd\,(u,v) = \frac{\sum_{Ti} \phi_t\,(\alpha \times Fb\,(u,v) + \beta \times Fa\,(v,v))}{\sum_{Ti} \phi_t \times Fc\,(u) + \sum_{Ti} \phi_t \times Fc\,(v)} \tag{27}$$

Here, ϕ_t is a time decay function. Ti states a series of time. $Fb\,(u,v)$ is the number of direct forwarding behaviors between users while $Fa\,(u,v)$ stands for the number of indirect forwarding behaviors among users. $Fc\,(u)$ represents the number of forwarding behaviors of user u and $Fc\,(v)$ denotes the number of forwarding behaviors of user v. α, β are the weights of the direct and indirect forwarding behaviors, respectively. Related studies show that direct forwarding behaviors contribute more to friend recommendations than indirect forwarding behaviors. α is greater than β and the sum of the weights is one. The parameter α is set to 0.7, based on experimental evaluations. The molecule refers to the related forwarding behaviors, and the denominator is the sum of the number of their forwarding behaviors.

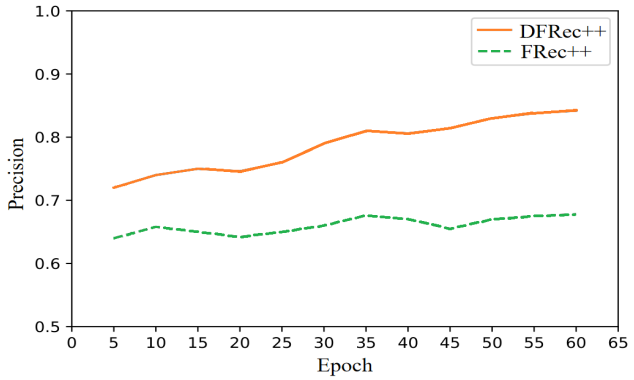


FIGURE 9. Experimental results.

The equations of similarity in making comments and replies is given in Eq.28:

$$Cm\,(u,v) = \frac{\sum_{Ti}\phi_t\,(\alpha \times Cb\,(u,v) + \beta \times Ca\,(u,v))}{\sum_{Ti}\phi_t \times Cc\,(u) + \sum_{Ti}\phi_t \times Cc\,(v)} \tag{28}$$

The denotes of parameters are same as the Fd. Following the same arguments, the similarity formulas for liking and mentioning other users are as follows:

$$Le\,(u,v) = \frac{\sum_{Ti}\phi_t \times Lc\,(u,v)}{\sum_{Ti}\phi_t \times Lc\,(u) + \sum_{Ti}\phi_t \times Lc\,(v)} \tag{29}$$

The equations for mentioning other users' similarity is given in Eq. 30:

$$At\,(u,v) = \frac{\sum_{Ti}\phi_t \times Ac\,(u,v)}{\sum_{Ti}\phi_t \times Ac\,(u) + \sum_{Ti}\phi_t \times Ac\,(v)} \tag{30}$$

$Le(u, v)$ is the similarity of liking behaviors. The molecule is the number of common liking behaviors, and the denominator is the sum of users' liking behaviors. In terms of $At(u, v)$, $Ac(u)$ denotes the number of friends that user u @ has. The $F_{dynamic}$ in FRec++ consists of the values: $Le(u, v)$, $Cm(u, v)$, $Fd(u, v)$, $At(u, v)$. Then, the static attributes feature, and the dynamic behavior feature are concatenated as F_{link} for the input of a fully connected neural network. The network structure of the other layers stays consistent with DFRec++. The comparison of the experimental results is shown in Figure 9.

As shown in Figure 9, the precision of FRec++ was found to be higher than expected. This shows that the design and usage of manual features is still reasonable. However, compared with DFRec++, the stability of the model is poor, and the performance is very low even with an increase in training. The precision of DFRec++ increases with a deepening of the training and follows a relatively stable trend. This experiment demonstrated that using the attention model and the embedded network as feature extractors makes using the complicated and cumbersome process of manual features unnecessary, and also improves the representation learning of the features so that the recommendations of friends are improved.

IV. RELATED WORK

Recently, an increasing number of researchers have been developing individual friend recommendation methods that combine several different algorithms based on social data such as relations [8], interests [23], and locations. Location-based recommendation (RWCFR) [13] is a typical state-of-the-art friend recommendation method that is similar to DFRec++. However, RWCFR's architecture is based on a random walk (RW), instead of on a hybrid architecture, which is based on LDA and weighted averages. Both methods are location-based, but RWCFR performs better in mobile social networks [12], while DFRec++ is better suited to online social networks. In addition, the DFRec++ model considers dynamic behaviors and static attributes while RWCFR only employs check-in data.

Inspired by developments in text mining, some researchers have used LDA in friend recommendation. For instance, Xu *et al.* [19] generate a user's subjects and interests using LDA and further recommends friends by finding other users with similar subjects. Others model the daily lives of users as life documents, their lifestyles as topics, and their activities as words to achieve satisfactory recommendation results. These studies prove that there are many gaps in friend recommendation methods using LDA. Meanwhile, the idea of embedding raw data features into a low-dimensional intensive vector has gained attention. Inspired by word embedding, some researchers have investigated document embedding technologies [4], [20] and dynamic network embedding [21]. TopicVec [20] is a generative model combining word embedding and LDA, with the aim of exploiting the word collection patterns both at the level of the local context and the global document. TopicVec learns high-quality document representations but lacks the idea of interactive information when applied to friend recommendation.

Also, deep-learning-based approaches have recently been proven to be effective in friend recommendations [24], [25]. Distributed representations of words can thoroughly capture semantic information [26], [27], [39], [46]. Convolutional neural network (CNN) has been proven a powerful approach to learn feature representations [18], [43], and it has the potential to learn sophisticated feature interactions [28].

Another important component in friend recommendation is to compute similarities among users. There exist numerous methods to perform this task; some examples are common neighbors (CN), Google PageRank ranking [30], and the Pearson correlation coefficient in collaborative filtering [31]. The authors of [29] consider the nodes within two hops of a target node to model dynamic social networks based on CN. However, methods that only consider a single factor are not effective. In terms of breadth, it has become a trend in friend recommendation to consider several attributes such as geographical locations, tags, and interests instead of only one attribute [32], [33]. Furthermore, it is equally important to reveal more user-related information about each attribute, i.e., and in-depth analysis is necessary.

In language and graph analysis, embedding representation has superior performance on sparse representations of several downstream tasks. In fact, embedding is also widely used in the recommendation field. Okura *et al.* [34] proposed an embedding-based method, which uses distributed representations in a three-step end-to-end manner. It significantly improves the efficiency of news recommendation. A novel way to generate embedding of customers was proposed by Chamberlain [35]. This method addresses several issues of the real e-commerce system. Wang *et al.* [36] pointed out that the methods based on the graph embedding framework resolve three challenges of Taobao, the largest online consumer-to-consumer (C2C) platform in China. Embedding can learn a low-dimensional intensive representation of features, which is also instructive for friend recommendation issues.

Attention mechanism is mainly used for task focusing. By decomposing tasks, different network structures (or branches) are designed to focus on different sub-tasks, and the learning ability of the network is reallocated, which reduces the difficulty of the original task and makes the network easier to train. Compared with traditional LSTM, the attention model can learn long-term dependency features better. Nowadays, the Attention model is more and more frequently used in recommendation systems [37], [38]. The Attention mechanism gives the model the ability to focus and enhance mission performance as a powerful feature extractor. Based on previous research, Attention will be more widely used in the field of friend recommendation.

There are other methods for friend recommendations. Traditional methods like collaborative filtering [40], [41] and content-based recommendations [42], [44] were widely used in the early development of social networks. While the former method is unsuitable for high-dimensional data and suffers from the cold-start problem, the latter considers a few of these attributes. Individual friend recommendation includes tag-based, emotion-based, and location-based methods. Tags can improve the accuracy of individual recommendations by scope list [45]. Emotion-based recommendation models extract a user's emotional words to obtain his/her emotional tendencies and compute his/her similarity with other users. Additionally, few studies have been conducted on how to apply deep neural network (DNN) methods to the friend recommendation task. One reason is that neural networks lack a corresponding pairwise ranking model. BayDNN [47] is a deep neural network for friend recommendation using only network structure information. Based on the approaches from the above studies, we propose DFRec++ to improve the accuracy of recommendations.

V. CONCLUSION

In this paper, we proposed a novel hybrid network framework combining Attention and network embedding. The DFRec++ method provides a method for deep extraction and representation of static attributes, dynamic behaviors, and social network structure information. We utilize the LDA algorithm to generate semantic topics and further form the static attribute features. CNN network extracts latent deep structural feature representations from the dynamic behaviors. Then we use the Attention to perform feature extraction on the attribute information. The algorithm LINE can help us learn the structure of the network and embed it into DFRec++. The experimental results showed that DFRec++ outperforms comparable methods with respect to precision(P@k), recall, and F1-measure. Considering these results, the following conclusions can be drawn: 1) Attention can learn the deep representation of attribute information 2) network structure embedding can improve the performance of friend recommendations 3)The provided method for feature extraction and representation is effective. The proposed method innovatively utilizes various kinds of information in Social Networks for friend recommendation, which brings inspirations to some researchers. Although the model works effectively and is compact, it is still slightly complicated. In the future, we hope to use attentional mechanisms at the full connectivity layer to further improve the predictive performance and we will study a simpler and more generalized network structure for friend recommendation.

REFERENCES

[1] Z. Liu, Y. Zhang, and Y. Edward, "PLDA+: Parallel latent Dirichlet allocation with data placement and pipeline processing," *Trans. Intell. Syst. Technol.*, vol. 2, no. 3, pp. 1–26, 2011.

[2] Z. Liu and X. Han, "Deep learning in knowledge graph," in *Proc. Deep Learn. Natural Lang. Process.*, 2018, pp. 117–145.

[3] R. Collobert, J. Weston, and L. Bottou, "Natural language processing (almost) from scratch," *J. Mach. Learn. Res.*, vol. 12, no. 1, pp. 2493–2537, 2011.

[4] J. Tang, M. Qu, and M. Wang, "Line: Large-scale information network embedding," in *Proc. 24th Int. Conf. World Wide Web*, 2015, pp. 1067–1077.

[5] Y. Tay, L. A. Tuan, and S. C. Hui, "Multi-cast attention networks," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining.*, 2018, pp. 2299–2308.

[6] B. Hu, C. Shi, and W. X. Zhao, "Leveraging meta-path based context for Top-*N* recommendation with a neural co-attention model," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2018, pp. 1531–1540.

[7] D. Arplt, "A closer look at memorization in deep networks," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2017, pp. 350–359.

[8] Y. Liu, H. Peng, J. Li, Y. Song, Y. Liu, and X. Li, "Event detection and evolution in multi-lingual social streams," *J. Frontiers Comput. Sci.*, 2019.

[9] J. Gong, X. Gao, and Y. Song, "Individual friends recomme dation based on random walk with restart in social networks," in *Proc. CCIS*, 2016, pp. 123–133.

[10] L. Yao, L. Wang, and L. Pan, "Link prediction based on common-neighbors for dynamic social network," in *Proc. 6th Int. Conf. Sustain. Energy Inf. Technol. (SEIT)*, 2016, pp. 82–89.

[11] V. Tran and G. Michael, "A spatial LDA model for discovering regional communities," in *Proc. IEEE/ACM Int. Conf. Adv. Social Netw. Anal. Mining*, 2013, pp. 162–168.

[12] J. Gong, X. Gao, and H. Cheng, "Integrating a weighted-average method into the random walk framework to generate individual friend recommendations," *Sci. China Inf. Sci.*, vol. 60, no. 11, 2017, Art. no. 110104.

[13] H. Bagci and P. Karagoz, "Context-aware friend recommendation for location based social networks using random walk," in *Proc. 25th Int. Conf. Companion World Wide Web (WWW)*, 2016, pp. 531–536

[14] X. Li and J. She, "Collaborative variational autoencoder for recommender systems," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2017, pp. 305–314.

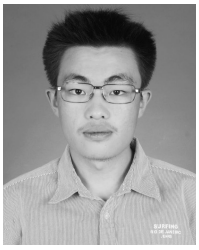
[15] H. Zhu, X. Li, and P. Zhang, "Learning tree-based deep model for recommender systems," in *Proc. 24rd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2018, pp. 1079–1088.
[16] Z. Yang, D. Li, and R. Lin, "An academic social network friend recommendation algorithm based on decision tree," in *Proc. IEEE SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI*, Oct. 2018, pp. 1311–1316.
[17] J. Gong, S. Chen, and X. Gao, "Integrating LDA into the weighted average method for semantic friend recommendation," in *Proc. 6th CCF Conf. Big Data*, 2018, pp. 427–441.
[18] H. Peng, J. Li, Q. Gong, Y. Song, and P. S. Yu, "Fine-grained event categorization with heterogeneous graph convolutional networks," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, 2019, pp. 3238–3245.
[19] K. Xu, X. Zheng, Y. Cai, H. Min, Z. Gao, B. Zhu, H. Xie, and T.-L. Wong, "Improving user recommendation by extracting social topics and interest topics of users in uni-directional social networks," *Knowl.-Based Syst.*, vol. 140, pp. 120–133, Jan. 2018.
[20] S. Li, T. Chua, and J. Zhu, "Generative topic embedding: A continuous representation of documents," in *Proc. 54th Annu. Meeting Assoc. Comput. Linguistics (ACL)*, Aug. 2016, pp. 666–675.
[21] H. Peng, J. Li, H. Yan, Q. Gong, S. Wang, L. Liu, L. Wang, and X. Ren, "Dynamic network embedding via incremental skip-gram with negative sampling," 2019, *arXiv:1906.03586*. [Online]. Available: https://arxiv.org/abs/1906.03586
[22] G. Xun, Y. Li, and J. Gao, "Collaboratively improving topic discovery and word embeddings by coordinating global and local contexts," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2017, pp. 535–543.
[23] X. Xu, J. Wang, H. Peng, and R. Wu, "Prediction of academic performance associated with Internet usage behaviors using machine learning algorithms," *J. Comput. Hum. Behav.*, vol. 98, pp. 166–173, Sep. 2019.
[24] H. Guo, R. Tang, and Y. Ye, "DeepFM: A factorization-machine based neural network for CTR prediction," in *Proc. Int. Joint Conf. Artif. Intell. (IJCAI)*, 2017, pp. 1725–1731.
[25] 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.
[26] Z. Liu, L. Zhang, C. Tu, and M. Sun, "Statistical and semantic analysis of rumors in chinese social media," *Scientia Sinica Inf.*, vol. 45, no. 12, pp. 1536–1546, 2015.
[27] T. Schnabel, I. Labutov, and D. Mimno, "Evaluation methods for unsupervised word embeddings," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2015, pp. 298–307.
[28] Y. Qu, H. Cai, and K. Ren, "Product-based neural networks for user response prediction," in *Proc. IEEE 16th Int. Conf. Data Mining (ICDM)*, Dec. 2016, pp. 1149–1154.
[29] W. Yu and G. Lin, "Social circle-based algorithm for friend recommendation in Online social networks," *Chin. J. Comput.*, vol. 37, no. 4, pp. 801–808, 2014.
[30] S. Guo, W. Zhang, and S. Zhang, "A pagerank-based collaborative filtering recommendation approach in digital libraries," *Tehnicki Vjesnik-Tech. Gazette*, vol. 24, no. 4, pp. 1051–1058, 2017.
[31] Y. Yd, D. Hooshyar, and J. Jo, "Developing a hybrid collaborative filtering recommendation system with opinion mining on purchase review," *J. Inf. Sci.*, vol. 44, no. 3, pp. 331–344, 2018.
[32] J. Zhu and L. Lu, "From interest to location: Neighbor-based friend recommendation in social media," *J. Comput. Sci. Technol.*, vol. 30, no. 6, pp. 1188–1200, 2015.
[33] B.-X. Wu, J. Xiao, and J.-M. Chen, "Friend recommendation by user similarity graph based on interest in social tagging systems," in *Proc. Int. Conf. Intell. Comput.*, vol. 9227. Cham, Switzerland: Springer, Aug. 2015, pp. 375–386.
[34] S. Okura, Y. Tagami, S. Ono, and A. Tajima, "Embedding-based news recommendation for millions of users," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2017, pp. 1933–1942.
[35] B. P. Chamberlain, Â. Cardoso, and C. H. B. Liu, "Customer lifetime value prediction using embeddings," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2017, pp. 1753–1762.
[36] J. Wang, P. Huang, and H. Zhao, "Billion-scale commodity embedding for E-commerce recommendation in alibaba," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2018, pp. 839–848.
[37] X. Wang, L. Yu, and K. Ren, "Dynamic attention deep model for article recommendation by learning human editors' demonstration," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2017, pp. 2051–2059.
[38] Y. Tay, L. A. Tuan, and S. C. Hui, "Multi-pointer co-attention networks for recommendation," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2018, pp. 2309–2318.
[39] Y. Liu, H. Peng, J. Guo, T. He, X. Li, Y. Song, and J. Li, "Event detection and evolution based on knowledge base," in *Proc. KBCOM*, 2018.
[40] J. Liu, M. Tang, Z. Zheng, X. F. Liu, and S. Lyu, "Location-aware and personalized collaborative filtering for Web service recommendation," *IEEE Trans. Services Comput.*, vol. 9, no. 5, pp. 686–699, Sep. 2016.
[41] H. Ma, M. Jia, and D. Zhang, "A content-based recommendation algorithm for learning resources," *Inf. Sci.*, vol. 385, pp. 325–337, Mar. 2016.
[42] L. Yang, B. Li, and X. Zhou, "Micro-blog friend recommendation algorithms based on content and social relationship," in *Proc. Int. Conf. Frontier Comput.*, vol. 422, 2016, pp. 105–121.
[43] H. Peng, J. Li, Y. He, Y. Liu, M. Bao, L. Wang, Y. Song, and Q. Yang, "Large-scale hierarchical text classification with recursively regularized deep graph-CNN," in *Proc. World Wide Web Conf.*, Apr. 2018, pp. 1063–1072.
[44] J. Shu, X. Shen, H. Liu, and Z. Zhang, "A content-based recommendation algorithm for learning resources," *Multimedia Syst.*, vol. 24, no. 2, pp. 163–173, 2018.
[45] H. Ma, M. Jia, and D. Zhang, "Combining tag correlation and user social relation for microblog recommendation," *Inf. Sci.*, vols. 385–386, pp. 325–337, Apr. 2017.
[46] Y. He, J. Li, Y. Song, M. He, and H. Peng, "Time-evolving text classification with deep neural networks," in *Proc. 27th Int. Joint Conf. Artif. Intell.*, 2018, pp. 2241–2247.
[47] D. Ding, M. Zhang, S. Li, and J. Tang, "BayDNN: Friend recommendation with Bayesian personalized ranking deep neural network," in *Proc. ACM*, 2017, pp. 1479–1488.

JIBING GONG received the Ph.D. degree from the Institute of Computing Technology, Chinese Academy of Sciences, China. He is currently a Professor with the School of Information Science and Engineering, Yanshan University. He is the Head of the Knowledge Engineering Group (KEG) research team in Yanshan University. His main research interests include big data analytics, heterogeneous information networks, machine learning, and data fusion.

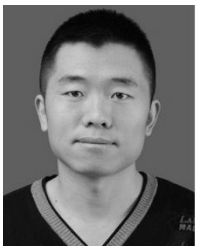
YI ZHAO is currently pursuing the master's degree with the Department of Information Science and Engineering, Yanshan University. His research interests include machine learning and deep learning.

SHUAI CHEN received the bachelor's degree in software engineering. He is currently pursuing the master's degree with the School of Information Science and Engineering, Yanshan University. His main research interests include deep learning and nlp.

HONGFEI WANG is currently pursuing the B.E. degree with the Department of Computer Science and Engineering, Beihang University (BUAA), Beijing, China.



LINFENG DU is currently pursuing the B.E. degree with the Department of Computer Science and Engineering, Beihang University (BUAA), Beijing, China.



SHULI WANG is currently a Lecturer with the Department of Applied Mathematics, Yanshan University. His research interests include machine learning arithmetic and energy power prediction of micro grid.

MD ZAKIRUL ALAM BHUIYAN is currently an Assistant Professor with the Department of Computer and Information Sciences, Fordham University. His research interests include dependable cyber physical systems, WSN applications, network security, urban computing, and sensor-cloud computing. He is an Associate Editor of IEEE ACCESS.

HAO PENG is currently an Assistant Professor with the School of Cyber Science and Technology, Beihang University. His current research interests include social text mining, representation learning, and deep learning.

BOWEN DU is currently a Professor with the School of Computer Science and Engineering, Beihang University. His current research interests include data mining on intelligent information systems, smart city technology, and multisource data fusion.