Applied Intelligence (2022) 52:8209–8225
https://doi.org/10.1007/s10489-021-02716-5

FEUI: *F*usion *E*mbedding for *U*ser *I*dentification across social networks

Lidong Wang[1] · Yin Zhang[2] · Keyong Hu[1]

Accepted: 15 July 2021 / Published online: 22 October 2021

Abstract
Recognizing identical users across different social networks remains challenging in recent years. Clearly, cross-platform user identification can play promising roles for many applications, such as user behavior prediction, public opinion analysis and e-commerce applications. Representation learning (RL) based methods have received more and more attention in recent years. However, most existing RL based methods only focus on the local structures (i.e., neighbors of vertices), and ignore label information and global structure patterns. Also, the current RL based methods tend to design the user identification and the embedding learning into two separate steps, which will neglect the complex correlations of different information sources. In this paper, we propose a novel approach, named as FEUI (Fusion Embedding for User Identification), by embedding the user-pair-oriented graph (UGP) through jointly integrating network structures, node attribute information and node labels to achieve robust embedding features and predict node labels simultaneously. The FEUI framework contains two modules, dual attribute embedding and joint embedding. These two modules leverage the strong representation ability of an extended auto-encoder and an one-input and two-outputs deep neural network to represent the complex correlations of different information sources. We evaluate our model on two social network datasets with collected user pairs. The experimental results show that the FEUI model can achieve better performance compared with the state-of-the-art approaches.

Keywords User identification · Representation learning · Network embedding · Deep neural network

1 Introduction

Recently, social networks have attracted enormous attention in people's daily lives. People can register as a user to enjoy different services in multiple platforms simultaneously. User identification across social networks, which is also refereed to as Social Network Alignment (SNA), has attracted more and more attention considering its significant value in friend recommendation, product recommendation, and network analysis [1] etc. If we collect one user's public postings form different social networks, we can provide an in-depth analysis on his/her interests. Thus, user identification across different social platforms is a precondition for the real-world applications across social networks, such as accurate commercial activity recommendation, user behavior analysis and public sentiment warning. However, the user attribute data is large, noisy and even incomplete. These social networks do not have direct links to each other. It is still challenging to recognize users across social networks with recent studies.

Most existing methods tend to use user profile information to identify users across networks. If the task depends only on attribute features, it may cause uncertainty results. For example, if one person creates a user "Violet Wang" on Twitter and a user "Vio Wang" on Facebook, we can not determine them to be the same person only by username. If the neighbors of these two users have similar usernames, these two users are more potential to be the same user. A lot of research has proved that the joint using of various information sources may lead to better results [2–4]. For example, Jain et al. [2] explored network attribute based identity search algorithm and concluded that the inclusion of more than one kind of features can improve the accuracy of an identity resolution process. Zhang et al. [3] incorporated attribute and structure information into an energy based model. Bartunov et al. [4] combined user attributes and social structure by CRF (Conditional Random Fields) .

✉ Yin Zhang
yinzh@zju.edu.cn

[1] Qianjiang College, Hangzhou Normal University, Zhejiang 310018, Hangzhou, China

[2] College of Science and Technology, Zhejiang University, Zhejiang 310012, Hangzhou, China

In recent years, more and more research focuses on the network embedding user identification methods. For instance, Man et al. [5] proposed a supervised social network alignment model by linking identities in the embedding space. Zhang et al. [6] analyzed on each user's ego networks embedding by combining profile attributes and social structure. These methods have the same idea that the users from two networks are mapped into a latent space first, then a projection function is learned to identify the same users. However, the network data is often heterogeneous, incomplete and even noisy, which brings great difficulties for obtaining a unified and informative representation. Although the network embedding based methods have been demonstrated to be effective, there are also remain some unsolved challenges: (a)***How to identify users across networks when two networks have large structural differences?*** Embedding on two networks independently may results in a great difference between two identical users' vector space, since the network structure features from two networks have large differences. It is necessary to integrate two social networks into one network to effectively extract common structure information. (b)***How to preserve the information of network structures (including global structures and local structures), node attributes, and labels in a common vector space?*** Most existing RL based methods do not consider the global patterns and label information for embedding. Network structure, node attributes, node labels are heterogeneous information sources, thus it is challenging to preserve their properties in a common vector space and incorporate these information into a unified embedding model; (c)***How to propose a unified approach to integrate embedding vector learning and user identification?*** Most previous methods separate the user identification task into two steps, which are embedding and aligning. These two steps are independent and serial, which can not capture the latent relationship between different information sources. Therefore, it is necessary to design a unified framework that can conduct network embedding and recognize identical users simultaneously.

Based on the above analysis, we combine two social networks to generate several user-pair-oriented graphs (UPGs) to effectively extract their common structure information. We propose a novel approach, termed as FEUI (Fusion Embedding for User Identification), by embedding the user-pair-oriented graph (UGP) through jointly integrating network structures, node attribute information and node labels to achieve robust embedding learning and user pair identification simultaneously. More specifically, ***to embed the information of network structures, node attributes, and partially labeled nodes in a common vector space***, we propose a unified framework, termed as FEUI, composed of two serial modules (dual attribute embedding module and joint embedding module) to jointly integrate network structures, node attribute information and label information to learn robust representations. These two modules leverage the strong representation ability of an extended auto-encoder and a deep neural network to represent the complex and deep correlations between different information sources; ***To embed the structure features of two social networks in one framework***, we generate a UPG from two social networks as the input of the FEUI model, and employ skip-gram model to represent the global structure relationship between the input node and the context; ***To integrate embedding vector learning and user identification simultaneously,*** we design a joint embedding module with one input and two outputs, which incorporates the label information into the context generation scheme, to conduct node label prediction and structure embedding learning simultaneously. The main contributions of our paper are listed as follows:

(1) We propose a novel unified framework FEUI. Different from previous works, FEUI first generates a UPG from two networks, then embeds the UPG by two modules, dual attribute embedding and joint embedding. Important factors (attribute information, local structures, global structures, labels) are formulated into a unified framework which allows embedding learning and user identification to be achieved simultaneously.
(2) The dual attribute embedding module, designed as an unsupervised setting, employs a dual attribute auto-encoder which can integrates attribute information and local structural proximity into a low-dimensional spaces; The joint embedding module is designed as a deep neural network with one input and two outputs to learn node embeddings and to predict node labels simultaneously. The joint embedding module is designed as a semi-supervised learning setting. The input of joint embedding module is the difference between two users' attribute embedding vectors. The two output layers are designed to predict the label of the node and the context in the UPG respectively. Due to the semi-supervised learning setting, our model does not need a large number of labeled data.
(3) We conduct extensively experiments on SNS and SZ (Sina Micro-blog and Zhihu) datasets. The experimental results demonstrate that the FEUI model compares favorably performance against the state-of-the-art methods.

2 Related works

Current research on our task can be divided into two main categories, surface feature based approaches and representation learning based approaches. The former approaches directly employ the surface feature of network structures (degree, clustering coefficient, common neighbors etc.) and user attributes (gender, education, username etc.) to identify users. In recent years, more and more researchers focus on the representation learning methods with the emerging of network embedding. The latter approaches aim to represent each node as a unique low-dimensional feature vector, and directly use some machine learning algorithms for our task with these learned vectors. In the following, we introduce the recent research about these two categories.

2.1 Surface feature based approaches

Profile attributes are most commonly used surface features for user identification [7–9]. For instance, Liu et al. [7] relied only on username to identify users and proposed two username features that are surface features and comparison features. Liang et al. [8] first extracted candidate user pairs, then adopted profile attributes similarity to further detect these candidate pairs by a greedy strategy. Wang et al. [9] extracted several attribute features, such as real name, username and education, for the users on Twitter and Github, and then applied Levenshtein distance to measure the user attribute similarity.

Structure features of a social network are more stable than users' attribute features, since some attributes are not public for privacy protection. To analyze privacy, Narayanan et al. [10] proposed NS method, based solely on node structure information. Some researches [11] found that some people may set up a similar friend circle in different networks, and then adopted a friendship structure (neighborhood structure) of seed user pairs to analyze cross-platform user identification. Due to the difficulty of collecting prior knowledge, Zhou et al. [12] proposed a friend relationship-based unsupervised scheme. Korula et al. [13] designed an efficient structure based parallel algorithm to identify the same user. Li et al. [14] fully analyzed the contributions of *k-hop* neighbors to user identification. The above methods do not explore deeper structure information from both local and global perspectives.

Excluding the attribute-based and structure-based methods, some researchers explore the hybird methods based on the combination of various information sources, such as profile information, text contents and structural patterns. Lee et al. [15] presented a visual comparison for user identification in terms of user attribute information, user text information, network structure, and the combination of these sources. Bartunov et al. [4] proposed a joint attribute-link approach that is extremely suitable for the condition that user profile data is noisy or incomplete. Zhang [3] incorporated local features and network structure features into an energy-based model. Liu et al. [16] proposed a normalized-margin-based linkage function formulation, and learned the linkage function by modeling behavior and structure factors across different platforms. These methods are supervised patterns that need labeled user pairs as prior knowledge. Liu et al. [17] first extracted potential seed pairs through attribute matching, and then constructed user relational matrix to overcome the social relation difference across networks. Fu et al. [18] incorporated graph structure and attributes by a user similarity measurement for deanonymization. This method overcomes the difficulty of collecting seed mappings, but at the cost of high running time.

2.2 Network embedding (NE) based approaches

Network embedding (NE) has attracted considerable attention in recent studies, which has been proved to be effective in some specific tasks, such as community detection, link prediction, and user clustering [19–21]. NE based methods can preserve the inter-connection relationships among the nodes. In recent years, network embedding based user identification has attracted considerable attention [6, 22, 23]. In the following, we discuss the recent NE based methods and give a detailed comparison on them.

IONE [23] and PALE [5]only employed structure information for embedding. More specifically, PALE was designed as a supervised algorithm, which captured the major structural patterns from observed user pairs as supervised information, and further learned a mapping function between two social sites for predicting other user pairs. IONE was designed as a semi-supervised approach that collaboratively modeled the input and output context of each user, and used a small set of annotated user pairs for embedding vector learning. These two methods do not explore the deep global structure regularities. Mego2vec [6] employed multi-view attribute embedding mechanism and convolved the adjacency matrix to capture structure information. LHNE [22] learned the representations of users in a unified space with both structural and content information, which also performed effectively without structural information. However, the content information of anchor users from different networks may vary greatly due to the different function of the networks. REGAL [24] proposed a matrix factorization-based method xNetMF to incorporate structure information and user attributes in an unsupervised way, which achieved good performance in both speed and accuracy. Different from our approach, these methods do not leverage label information. Moreover, the above methods contain two stages, embedding stage and matching stage, which can not capture the complex correlations of different information sources. SNNA [25] adopted TADW [26] to embeds the text attributes and topology information to an embedding vector, and performed our task in the distribution level, then proposed several adversarial learning algorithms to learn a projection function with a small fraction of labeled user pairs. Inspired by these recent works, we propose a novel framework FEUI to identify users across two networks. Our proposed method integrates embedding stage and matching stage into one framework. Table 1 shows the differences between FEUI and the above research.

Table 1 A comparison of FEUI with other methods. √ means using the corresponding type of information while × indicates the opposite

Methods	Structure	Attributes	Content	Labels	Semi-supervised
LHNE [22]	√	×	√	×	×
Mego2Vec [6]	√	√	×	×	×
IONE [23]	√	×	×	×	√
PALE [5]	√	×	×	×	×
SNNA [25]	√	√	×	×	×
REGAL [24]	√	√	√	×	×
FEUI	√	√	×	√	√

3 Problem definition

We use $G = (U, E, \mathbf{X})$ to represent a social network, where $U = \{u_1, u_2, \ldots, u_n\}$ is the user set and E represents the topology information, $e_{ij} = (u_i, u_j) \in E$. $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$ is the attribute information matrix, which is extracted from the user's profile information, such as education, usernames, affiliations and locations. $G^A = (U^A, E^A, \mathbf{X}^A)$ denotes network A, and G^B denotes network B. Table 2 shows some frequently used notations in this paper. If two users u_k^A and u_k^B form a candidate user pair, we denote it as $p_k = \left(u_k^A, u_k^B\right)$. The candidate user pairs are the inter-connected user pairs that have some possibilities to be the same user.

Definition 1 (user-pair-oriented graph) We denote $UPG = (U_{UPG}, E_{UPG})$ as a user-pair-oriented Graph, where $u'_i \in U_{UPG}$, $e'_{ij} = (u'_i, u'_j) \in E_{UPG}$. Let$u'_1 = \left(u_1^A, u_1^B\right) \in U_{UPG}$ and $u'_2 = \left(u_2^A, u_2^B\right) \in U_{UPG}$ be two nodes, these two nodes have an edge if

$$u_1^A \in N\left(u_2^A\right), u_1^B \in N\left(u_2^B\right) \tag{1}$$

where $N(\cdot)$ represents the neighborhood set. After candidate user pair selection, we set each pair as the node u'_i. Based on this, we define our task as follows:

Problem 1 (user identification across networks) Given a UPG generated from two social networks G^A and G^B, and a small set of labeled pairs Y^L, we decide whether u_i^A and u_j^B refer to a correct matching. The objection of our task is to learn a function to determine y_i for each node u'_i in the UPG:

$$f : \left(UPG, Y^L\right) \rightarrow Y \tag{2}$$

As shown in Fig. 1, white pairs connected with a dotted line are known labeled pairs, gray pairs are not matched, our task is to discover the pink pairs according to the learned function.

4 Methodology

We argue that the performance of our task can be improved if we consider various information sources simultaneously, such as the node attributes, the topology information, and the label information. To solve our task, we integrate two networks into several UPGs, where a node represents two users from two networks. Given a UPG, we intend to represent each node as a unified embedding space, where such space preserves important information from attribute space, local structural information, global structural information and label information. Thus, we propose a unified embedding model seamlessly

Table 2 Notations

Symbol	Description
$G=(U,E,\mathbf{X})$	A social network with user set U, relationship set E and attribute matrix $\mathbf{X}$.
$UPG=(U_{UPG},E_{UPG})$	User-pair-oriented Graph with user set U_{UPG} and relationship set E_{UPG}.
$P=\{p_1,p_2,\ldots,p_k\}$	A set of candidate user pairs.
u_i^A	User i in network A, $u_i^A \in U^A$
u'_i	node i in a UPG,$u'_i = \left(u_i^A, u_i^B\right)$, $u'_i \in U_{UPG}$
$Y=\{y_1,y_2,\ldots,y_k\}$	A set of binary indicators for the nodes in U_{UPG}.
Y^L	A set of labeled identical user pairs, $Y^L \in Y$
$y_i \in \{1,0\}$	The binary indicator for u'_i, which represents whether two users in u'_i is a correct matching or not
$\mathbf{h}_i$	The learned embedding vector of node u'_i

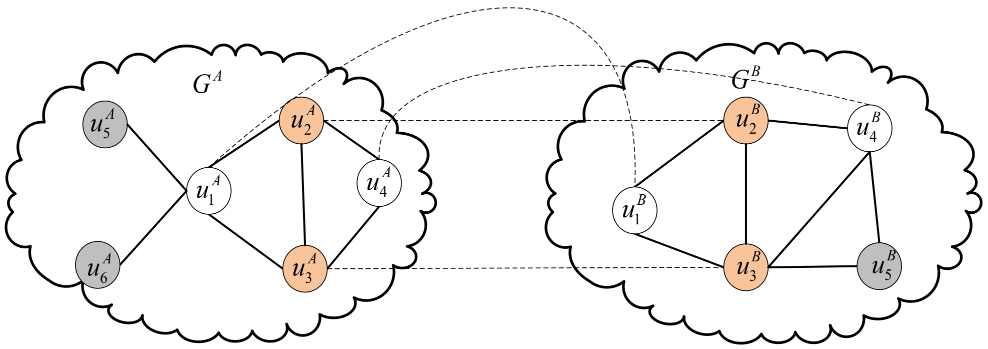


Fig. 1 Illustration of user identification across two networks

integrating network structural proximity, node attributes proximity and node labels on the UPGs. The unified framework is named as FEUI with two serial modules. The process of the FEUI is shown in Fig. 2.

As shown in Fig. 2, we use several generated UPGs as the input of FEUI. The framework consists of two modules: dual attribute embedding and joint embedding. The dual attribute embedding module adopts node attributes ($\mathbf{x}_i^A$ and $\mathbf{x}_i^B$) and its neighbors' attributes($\mathbf{x}_{Ni}^A$ and$\mathbf{x}_{Ni}^B$)from networks G^A and G^B as input, and designs an extended auto-encoder to generate two latent spaces $\mathbf{z}_i^A$ and $\mathbf{z}_i^B$ for u_i^A and u_i^B respectively. The joint embedding module is designed as a deep neural model with one-input layer and two-output layers. For user matching, we believe that the embedding space of two identical user should be close to each other. Thus, we set the difference of the embedding vectors $|\mathbf{z}_i^A - \mathbf{z}_i^B|$ as the attribute of the node u'_i, and use $|\mathbf{z}_i^A - \mathbf{z}_i^B|$ as the input of joint embedding. Based on this, we employ skip-gram model and incorporate label information to learn the final embedding. The final embedding of an node in UPG is trained to predict the label of the node and the context simultaneously. Thus, the output of joint embedding includes the label prediction and the graph context prediction. Our model can capture various important factors in a unified model, and can achieve robust representation learning ($\mathbf{h}_i$) and predict node label (y_i) simultaneously. In the following, we will explain these two modules.

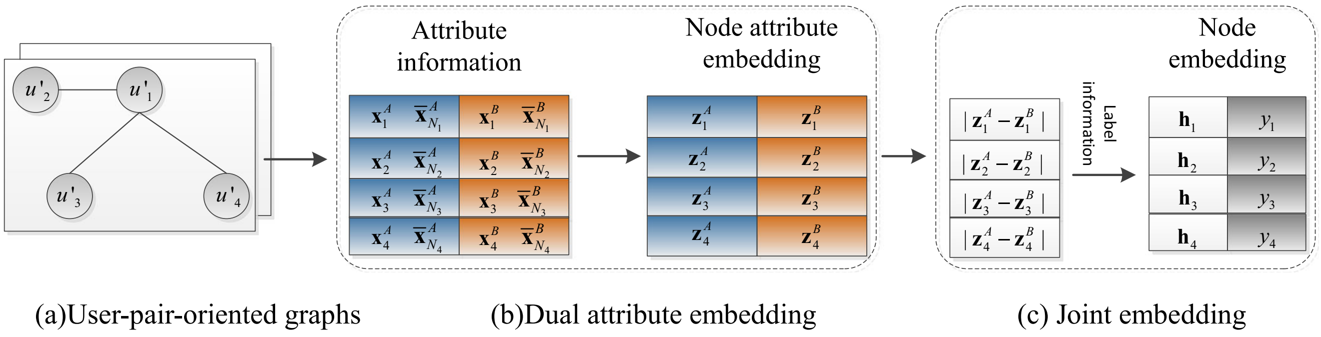
4.1 User-pair-oriented graph generation

The generated UPG will be very large if all possible node pairs are imported. Here, we adopt the method from our previous work [27] to select proper candidate user pairs.

We give a case of UPG generation based on the Definition 1. (u_1^A, u_1^B) and (u_3^A, u_3^B) are two candidate user pairs achieved by username similarity calculation. We then extend to their friendships, and the pair (u_2^A, u_2^B) is added in the UPG. We have three nodes in the generated UPG, where node u'_1 is a labeled node pair, and our method has to predict the label of other nodes(Fig. 3).

4.2 Dual attribute embedding

Recently, auto-encoder [28] and its variants [29, 30] have been proved to be a powerful way to encode the inputs into some representations which can revert to their original forms. In an auto-encoder model, the objective function should make sure that the input feature vector has to be as similar to the reconstruction space as possible. The dual attribute embedding, which extends from traditional deep auto-encoder model, can be considered as an unsupervised part of our model. By incorporating the neighborhoods into the input, the attribute information and the local structural information are tightly inter-connected, which ensure that the low-dimensional embedding feature space can preserve the attribute and local



u'_1
u'_2
u'_3
u'_4
Attribute information
$\mathbf{x}_1^A$ $\bar{\mathbf{x}}_{N_1}^A$ $\mathbf{x}_1^B$ $\bar{\mathbf{x}}_{N_1}^B$
$\mathbf{x}_2^A$ $\bar{\mathbf{x}}_{N_2}^A$ $\mathbf{x}_2^B$ $\bar{\mathbf{x}}_{N_2}^B$
$\mathbf{x}_3^A$ $\bar{\mathbf{x}}_{N_3}^A$ $\mathbf{x}_3^B$ $\bar{\mathbf{x}}_{N_3}^B$
$\mathbf{x}_4^A$ $\bar{\mathbf{x}}_{N_4}^A$ $\mathbf{x}_4^B$ $\bar{\mathbf{x}}_{N_4}^B$
Node attribute embedding
$\mathbf{z}_1^A$ $\mathbf{z}_1^B$
$\mathbf{z}_2^A$ $\mathbf{z}_2^B$
$\mathbf{z}_3^A$ $\mathbf{z}_3^B$
$\mathbf{z}_4^A$ $\mathbf{z}_4^B$
$|\mathbf{z}_1^A - \mathbf{z}_1^B|$
$|\mathbf{z}_2^A - \mathbf{z}_2^B|$
$|\mathbf{z}_3^A - \mathbf{z}_3^B|$
$|\mathbf{z}_4^A - \mathbf{z}_4^B|$
Label information
Node embedding
$\mathbf{h}_1$ y_1
$\mathbf{h}_2$ y_2
$\mathbf{h}_3$ y_3
$\mathbf{h}_4$ y_4
(a)User-pair-oriented graphs
(b)Dual attribute embedding
(c) Joint embedding

Fig. 2 The process of the FEUI. (a) Generate several user-pair-oriented graphs(UPGs) from two social networks; (b) Input the attributes of the nodes and the average attributes of their neighborhoods to dual attribute embedding module to learn node attribute embedding vectors; (c) Input the difference of attribute embedding vectors to joint embedding module to learn the topology embedding vectors and predict the labels

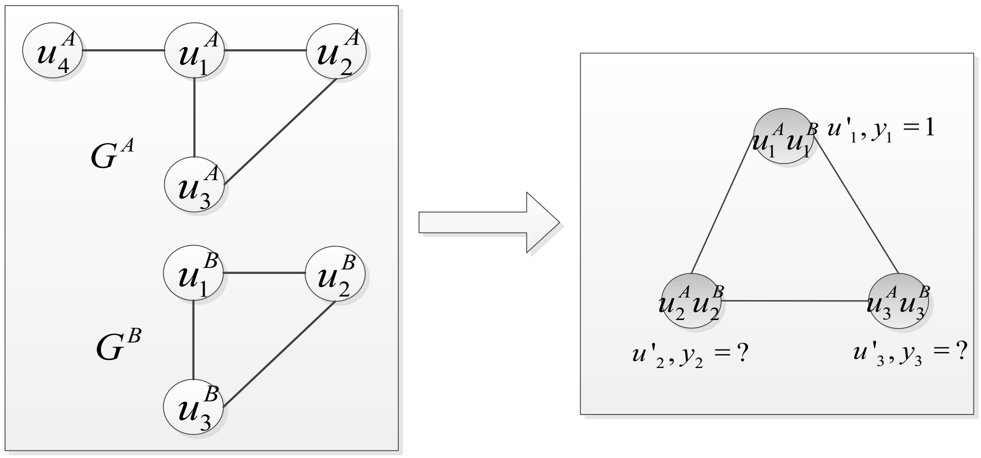
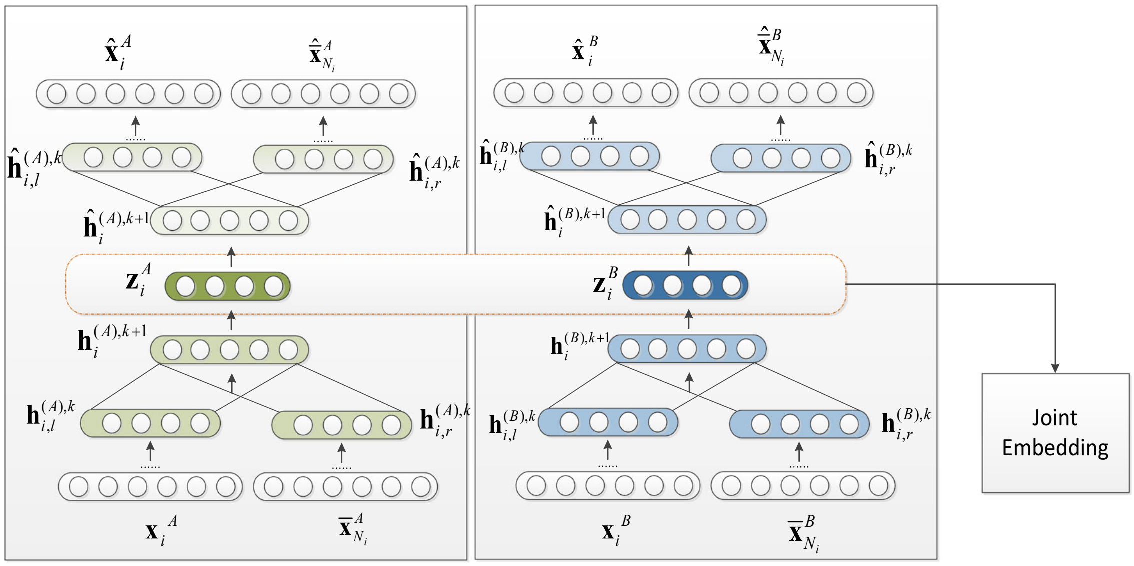


Fig. 3 A case of user-pair-oriented graph generation

structural proximity. By incorporating the neighborhood attributes, this embedding model can smooth out the incompleteness and noisy of attribute feature. For example, not all attributes are available for some users. Thus, the model can rely more on neighborhood attributes for better representation results.

Many users register in social sites with rich attribute information, including gender, education and country. We convert discrete attributes to a set of binary features via one-hot encoding and transform continuous textual attributes to a real-valued vector via TF-IDF, which is similar to the previous work [31]. Then, we adopt an attribute auto-encoder model for two networks G^A and G^B respectively. The attribute auto-encoder model has two inputs, the attributes of the node $\mathbf{x}_i$ and the average attributes of its neighborhoods $\mathbf{x}_{N_i}$. It generates the low-dimensional embedding vector based on $\mathbf{x}_i$ and $\mathbf{x}_{N_i}$ independently, which will be further combined to reconstruct the final feature vectors that are close to the input vectors.

As shown in the left part of Fig. 4, two separated encoders are implemented on $\mathbf{x}_i$ and $\mathbf{x}_{N_i}$ simultaneously. Their latent representation vectors of nodeu_i^A are represented as $\left\{\mathbf{h}_i^{(A),1}, \mathbf{h}_i^{(A),2}, \ldots \mathbf{h}_i^{(A),k}\right\}$ for different hidden layers {1, 2, …, k} in the encoder component. The differences between our model and traditional auto-encoder are the combination part and the dispatch part. The former part combines the node attribute vector and neighborhood average attribute vector to obtain the embedding vector $\mathbf{z}_i^A$ for user u_i^A, which can preserve both attribute proximity and local structural proximity. The latter component dispatches the embedding vector $\mathbf{z}_i^A$ back to the hidden vectors, which are represented as $\left\{\widehat{\mathbf{h}}_i^{(A),1}, \widehat{\mathbf{h}}_i^{(A),2}, \ldots \widehat{\mathbf{h}}_i^{(A),k}\right\}$. Formally, the relationship between



$\hat{\mathbf{x}}_i^A$
$\hat{\mathbf{x}}_{N_i}^A$
$\hat{\mathbf{h}}_{i,l}^{(A),k}$
$\hat{\mathbf{h}}_{i,r}^{(A),k}$
$\hat{\mathbf{h}}_i^{(A),k+1}$
$\mathbf{z}_i^A$
$\mathbf{h}_i^{(A),k+1}$
$\mathbf{h}_{i,l}^{(A),k}$
$\mathbf{h}_{i,r}^{(A),k}$
$\mathbf{x}_i^A$
$\overline{\mathbf{x}}_{N_i}^A$
$\hat{\mathbf{x}}_i^B$
$\hat{\overline{\mathbf{x}}}_{N_i}^B$
$\hat{\mathbf{h}}_{i,l}^{(B),k}$
$\hat{\mathbf{h}}_{i,r}^{(B),k}$
$\hat{\mathbf{h}}_i^{(B),k+1}$
$\mathbf{z}_i^B$
$\mathbf{h}_i^{(B),k+1}$
$\mathbf{h}_{i,l}^{(B),k}$
$\mathbf{h}_{i,r}^{(B),k}$
$\mathbf{x}_i^B$
$\overline{\mathbf{x}}_{N_i}^B$
Joint Embedding

Fig. 4 The architecture of dual attribute embedding module

these layers in the encoder component are represented as follows:

$$\begin{cases} \mathbf{h}_{i,l}^{(A),1} = \sigma\left(\mathbf{W}_l^{(A),1}\mathbf{x}_i^A + \mathbf{b}_l^{(A),1}\right) \\ \mathbf{h}_{i,r}^{(A),1} = \sigma\left(\mathbf{W}_r^{(A),1}\overline{\mathbf{x}}_{N_i}^A + \mathbf{b}_r^{(A),1}\right) \\ \mathbf{h}_{i,l}^{(A),m} = \sigma\left(\mathbf{W}_l^{(A),m}\mathbf{h}_{i,l}^{(A),m-1} + \mathbf{b}_l^{(A),m}\right), m \in \{2,3,\ldots,k\} \\ \mathbf{h}_{i,r}^{(A),m} = \sigma\left(\mathbf{W}_r^{(A),m}\mathbf{h}_{i,r}^{(A),m-1} + \mathbf{b}_r^{(A),m}\right), m \in \{2,3,\ldots,k\} \\ \mathbf{h}_i^{(A),k+1} = \sigma\left(\mathbf{W}_l^{(A),k+1}\mathbf{h}_{i,l}^{(A),k} + \mathbf{b}_l^{(A),k+1} + \mathbf{W}_r^{(A),k+1}\mathbf{h}_{i,r}^{(A),k} + \mathbf{b}_r^{(A),k+1}\right) \\ \mathbf{z}_i^A = \sigma\left(\mathbf{W}^{(A),k+2}\mathbf{h}_i^{(A),k+1} + \mathbf{b}^{(A),k+2}\right) \end{cases} \tag{3}$$

During the above equation, $\sigma(\cdot)$ denotes a non-linear activation function. $\mathbf{W}^m$ and $\mathbf{b}^m$ are the weights and biases in the $m-th$ layer. We use the subscript l to denote the variables for the input $\mathbf{x}_i$, and use the subscript r to denote the variables for the input $\mathbf{x}_{N_i}$. For example,$\mathbf{h}_{i,l}^{(A),1}$ denotes the latent representation vector at hidden layer 1 for the input $\mathbf{x}_i$.

Meanwhile, we input the embedding vector $\mathbf{z}_i^A$ and output the reconstructed vector $\widehat{\mathbf{x}}_i$ in the decoder component. The relationship between the latent vectors $\left\{\widehat{\mathbf{h}}_i^{(A),1}, \widehat{\mathbf{h}}_i^{(A),2}, \ldots \widehat{\mathbf{h}}_i^{(A),k}\right\}$ are represented as follows.

$$\begin{cases} \widehat{\mathrm{h}}_i^{(A),k+1} = \sigma\left(\widehat{\mathrm{W}}^{(A),k+2} z_i^A + \widehat{\mathrm{b}}^{(A),k+2}\right) \\ \widehat{\mathrm{h}}_{i,l}^{(A),k} = \sigma\left(\widehat{\mathrm{W}}_l^{(A),k+1} \widehat{\mathrm{h}}_i^{(A),k+1} + \widehat{\mathrm{b}}_l^{(A),k+1}\right) \\ \widehat{\mathrm{h}}_{i,r}^{(A),k} = \sigma\left(\widehat{\mathrm{W}}_r^{(A),k+1} \widehat{\mathrm{h}}_i^{(A),k+1} + \widehat{\mathrm{b}}_r^{(A),k+1}\right) \\ \widehat{\mathrm{h}}_{i,l}^{(A),m-1} = \sigma\left(\widehat{\mathrm{W}}_l^{(A),m} \widehat{\mathrm{h}}_{i,l}^{(A),m} + \widehat{\mathrm{b}}_l^{(A),m}\right), m\ \varepsilon\{2,3,\cdots,k\} \\ \widehat{\mathrm{h}}_{i,r}^{(A),m-1} = \sigma\left(\widehat{\mathrm{W}}_r^{(A),m} \widehat{\mathrm{h}}_{i,r}^{(A),m} + \widehat{\mathrm{b}}_r^{(A),m}\right), m\ \varepsilon\{2,3,\cdots,k\} \end{cases} \tag{4}$$

The output of the decoder component can be represented with:

$$\begin{cases} \widehat{\mathbf{x}}_i^A = \sigma\left(\widehat{\mathbf{W}}_l^{(A),1} \widehat{\mathbf{h}}_{i,l}^{(A),1} + \widehat{\mathbf{b}}_l^{(A),1}\right) \\ \widehat{\overline{\mathbf{x}}}_{N_i}^A = \sigma\left(\widehat{\mathbf{W}}_r^{(A),1} \widehat{\mathbf{h}}_{i,r}^{(A),1} + \widehat{\mathbf{b}}_r^{(A),1}\right) \end{cases} \tag{5}$$

To minimize the distance between the input vectors and the output vectors, the loss term for G^A can be represented as:

$$L^{(A)} = \sum_i \left(\left\| \mathrm{x}_i^A - \widehat{\mathrm{x}}_i^A \right\|^2 + \left\| \overline{\mathrm{x}}_{N_i}^A - \widehat{\overline{\mathrm{x}}}_{N_i}^A \right\|^2 \right) \tag{6}$$

Similarly, the loss function for G^B can be denoted as $L^{(B)}$.

$$L^{(B)} = \sum_i \left(\left\| \mathrm{x}_i^B - \widehat{\mathrm{x}}_i^B \right\|^2 + \left\| \overline{\mathrm{x}}_{N_i}^B - \widehat{\overline{\mathrm{x}}}_{N_i}^B \right\|^2 \right) \tag{7}$$

4.3 Joint embedding

4.3.1 Overview of joint embedding

As shown above, the dual attribute embedding module can integrate attributes and local network structure in a unified model. In this part,we investigate how to jointly modeling of class label prediction and global structure embedding, and how labels can be modeled and incorporated to preserving better proximity among nodes. This is a challenging task since labels are completely different with the node structure information. We design a deep neural network with one input and two outputs. The input can be formulated as the difference of their embeddings $\mathbf{z}_i^A$ and $\mathbf{z}_i^B$, i.e., $\mathbf{d}_i = |\mathbf{z}_i^A - \mathbf{z}_i^B|$. In this module, the embedding of an node in UPG is trained to predict the node label and the context simultaneously.

Building a tower structure has been proved to improve the quality of representation vector. As depicted in Fig. 5, we use a tower structure with a small number of hidden layers for input layer, to learn deeper representative features of the input data. The left output layers are designed as the supervised part of joint embedding, and the right output layers are designed as the unsupervised part.

The *m-th* layer of the model can be represented as:

$$\begin{aligned} \mathbf{h}_i^m &= \delta\left(\mathbf{W}^m\mathbf{h}_i^{m-1} + \mathbf{b}^m\right) \\ \mathbf{h}_{i,l}^{m+1} &= \delta\left(\mathbf{W}_l^{m+1}\mathbf{h}_i^m + \mathbf{b}_l^{m+1}\right) \\ \mathbf{h}_{i,r}^{m+1} &= \delta\left(\mathbf{W}_r^{m+1}\mathbf{h}_i^m + \mathbf{b}_r^{m+1}\right) \end{aligned} \tag{8}$$

where $\delta(\cdot)$ represents the sigmoid function,$\mathbf{W}^m$ and $\mathbf{b}^m$ are the weights and biases in the *m-th* layer, $m \geq 1$. Moreover,$\mathbf{W}_l^{m+1}$ denotes the weights of the left branch layers in the *(m + 1)-th* layer, while$\mathbf{W}_r^{m+1}$ denotes the weights of the right branch layers in the *(m + 1)-th* layer. The left output layer predicts the label y_i. $y_i = 1$ means the node u_i^A and u_i^B are the same user, otherwise not. In this way, the user identification problem is transformed to a binary classification problem since the nodes in the UPG only have two kinds of labels. The right output layer is designed as an unsupervised component, which adopts skip-gram [32] model to predict the context of the network.

For some popular social networks, previous studies have already investigated that users can have similar friends across different social sites [12]. It has been confirmed that the relationship among users in Twitter are similar to their relationships in Facebook [33]. Thus, we can consider the friendship relationship in a social network. If user u_i^A is matched to u_j^B,

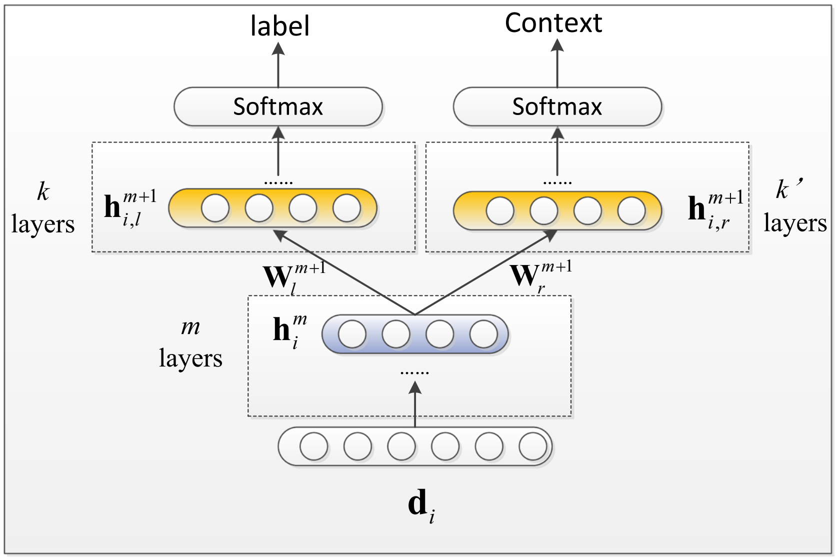


Fig. 5 The architecture of joint embedding

user u_i^A 's first-order or second-order friends in G^A can probably be matched tou_j^B 's friends in G^B. It means that the two close nodes in a UPG would have high probability to have same label. Thus, we can apply skip-gram for relationship modeling between two close nodes through the whole UPG. One node is considered as the target node (input), and the other is considered as the context node (output). Moreover, if two nodes have the same label, then we hope these two nodes should have similar representations. Considering about this, we integrate the label prediction with context generation into a unified model. In this way, FEUI model preserves node attributes, local network structure, global network structure information and label information in a unified framework.

4.3.2 Loss function for joint embedding

As shown in Fig. 5, we apply a canonical multilayer perceptron to be the left part of the joint embedding module. Its loss function is:

$$L^{(L)} = -\sum_{i\in U_{UPG}^L} \log p(y_i|\mathbf{d}_i) \tag{9}$$

where U_{UPG}^L denotes the labeled nodes in the UPG, $p(y_i|\ \mathbf{d}_i)$ can be represented as:

$$p(y_i|\mathbf{d}_i) = \frac{\exp\left(\left(\mathbf{h}_{i,l}^{m+k}\right)^T \mathbf{W}_l^{y_i}\right)}{\sum_{y_j\in[0,1]} \exp\left(\left(\mathbf{h}_{i,l}^{m+k}\right)^T \mathbf{W}_l^{y_j}\right)} \tag{10}$$

where $\mathbf{W}_l$ is the weight matrix of the softmax layer, k is the number of hidden layers. $\mathbf{h}_{i,l}^{m+k}$ is the input value of the softmax layer for label prediction, which can be calculated as:

$$\mathbf{h}_{i,l}^{m+k} = \delta\left(\mathbf{W}_l^{m+k}\mathbf{h}_{i,l}^{m+k-1} + \mathbf{b}_l^{m+k}\right) \tag{11}$$

To capture the global structure information in joint embedding module, we employ skip-gram to capture the relationship between the input node and the context nodes in the right part of this module. Given the node u'_i with its attribute feature vector $\mathbf{d}_i$, we define the loss function for all random walk context $c_i \in C$ as follows:

$$L^{(R)} = -\sum_{i=1}^{n} \sum_{u'_j\epsilon c_i} \log p\left(u'_j|d_i\right) \tag{12}$$

where $n = \ |U_{UPG}|$, c_i denotes the context set of node u'_i, $u'_j \in c_i$. $p(u'_j|\ \mathbf{d}_i)$ is denoted as:

$$p\left(u'_j|\mathrm{d}_i\right) = \frac{\exp\left(\left(\mathrm{h}_{i,r}^{m+k'}\right)^T \mathrm{W}_r^j\right)}{\sum_{o=1}^n \exp\left(\left(\mathrm{h}_{i,r}^{m+k'}\right)^T \mathrm{W}_r^o\right)} \tag{13}$$

where $\mathbf{W}_r$ is the weight matrix of the softmax layer, $\mathbf{h}_{i,r}^{m+k'}$ is the input values of the softmax layer for the context prediction:

$$\mathrm{h}_{i,r}^{m+k'} = \delta\left(\mathrm{W}_r^{m+k'}\mathrm{h}_{i,r}^{m+k'-1} + \mathrm{b}_r^{m+k'}\right) \tag{14}$$

4.3.3 Context nodes sampling

It has to be noted that our model is different from traditional deep walk models. We incorporate the label information into the embeddings through label context generation. The context of a node is divided into two categories [34], which are structure context and label context. The structure context is derived based on the network structure. We generate structure context by truncated random walks on the UPG network to encode the global structure information. The label context is defined as the nodes having the same label, from which we use to inject label information into the embeddings. For example, the context of the labeled node u'_i can be generated by uniformly sampling from nodes with label y_i. Specifically, we generate w context nodes by the sampling process illustrated in Algorithm 1 for each node u'_i. We use the parameter α to set the ratio of label context. Specifically, for a labeled node, we generate $w * \alpha$ labeled context nodes, $w * (1 - \alpha)$ structure context nodes. For an unlabeled node, we generate w context nodes.

Algorithm 1 Context nodes generation

```
Input: $UPG = (U_{UPG}, E_{UPG})$, target node $u'_i$, parameter $\alpha$, number of context nodes $w$
Output: Context $c_i$
1  $c_i = \varnothing$
2  If $y_i \in Y^L$ then
3      $S_i = \{j \mid y_j \in Y^L \ and \ y_i = y_j\}$
4      $c_i \leftarrow$ uniformly sample $w * \alpha$ label context nodes from $S_i$
5      $c_i \leftarrow c_i \bigcup$ generate $w * (1 - \alpha)$ context nodes
6  else
7      $c_i \leftarrow c_i \bigcup$ generate $w$ context nodes
8  return $c_i$
```

4.4 FEUI model training

The final loss function of FEUI framework contains the loss terms from dual attribute embedding module and joint embedding module. The former component includes the loss term $L^{(A)}$ for the auto-encoder in G^A (see Eq.(6)) and the loss term $L^{(B)}$ for the auto-encoder in G^B; The latter component includes the loss terms for the left supervised part $L^{(L)}$ (see Eq.(9)) and the loss terms for the right unsupervised part $L^{(R)}$ (see Eq.(12)). The final loss function can be defined as:

$$L = L^{(A)} + L^{(B)} + \beta \cdot L_{reg} + L^{(L)} + L^{(R)} \tag{15}$$

The regularization term $Lreg$ is added to avoid over-fitting, which is defined as follows:

$$L_{reg} = L_{reg}^{(A)} + L_{reg}^{(B)}$$

$$L_{reg}^{(A)} = \sum_i^{k+2} \left(\left\| \mathbf{W}_l^{(A),i} \right\|_F^2 + \left\| \widehat{\mathbf{W}}_l^{(A),i} \right\|_F^2 + \left\| \mathbf{W}_r^{(A),i} \right\|_F^2 + \left\| \mathbf{W}_r^{(A),i} \right\|_F^2 \right) \tag{16}$$

$$L_{reg}^{(B)} = \sum_i^{k+2} \left(\left\| \mathbf{W}_l^{(B),i} \right\|_F^2 + \left\| \widehat{\mathbf{W}}_l^{(B),i} \right\|_F^2 + \left\| \mathbf{W}_r^{(B),i} \right\|_F^2 + \left\| \mathbf{W}_r^{(B),i} \right\|_F^2 \right)$$

To optimize the objective function, we describe the optimization procedures for each respective part. Firstly, we adopt Mini-Batch SGD (Mini-Batch Stochastic Gradient Descent) to optimize $L^{(A)}$ and $L^{(B)}$. Specifically, the training step contains several epochs. In each epoch, we randomly sample a mini-batch of the training data to update the parameters with SGD. The step stops when it reaches the max training epochs.

For the left part of Fig. 5, it is easily to use back-propagation and SGD to train the MLP. However, It is usually intractable to directly compute $\nabla p(u'_j | \mathbf{d}_i)$, which needs the normalization over iterating through all the nodes. Negative sampling was introduced to address this issue [35], which samples negative examples to approximate the normalization term. Thus, the loss can be rewritten as:

$$\widetilde{L}_{(R)} = -\sum_{i=1}^{n} \sum_{u' \epsilon c_i} \left[\log \sigma \left(\left(\mathrm{h}_{i,r}^k \right)^T \mathrm{W}_r^{u'} \right) + \sum_{\widetilde{u} \in U_{neg}^{u'}} \log \sigma \left(- \left(\mathrm{h}_{i,r}^k \right)^T \mathrm{W}_r^{\widetilde{u}} \right) \right] \tag{17}$$

where σ is the sigmoid function, and$U_{neg}^{u'}$ denotes the t -randomly selected negative samples. To optimize the loss function for joint embedding, we jointly minimize $L^{(L)}$ and $\widetilde{L}_{(R)}$ in our model. We adopt Mini-Batch SGD and alternate update the parameters between the left and the right of the model., We jointly train this module with a batch size of b_1 for the left part and a batch size of b_2 for the right part. As illustrated in Algorithm 2, the process stops when it reaches the max iterations.

Algorithm 2 Learning for joint embedding module

```
Input: $UPG = (U_{UPG}, E_{UPG})$, $Y^L$ $b_1, b_2, t$
Output: Embedding $\mathbf{H} = (\mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_n)$ for each node
1. Random initialization for all parameters in the model.
2. While not reaching max iterations do
3.     Sample $b_1$ labeled nodes from $Y^L$.
4.     Calculate $\nabla L^{(L)}$ and update the weights and biases.
5.     $C = \varnothing, U_{neg} = \varnothing$
6.     Sample $b_2$ instances from $U_{UPG}$, denoted as $U_b$
7.     For each node $i$ in $U_b$ do
8.         Use Algorithm 1 to obtain $c_i$
9.         $C = C \bigcup c_i$
10.        For each node $u'_i \in c_i$ do
11.            $U_{neg}^{u'} \leftarrow$ randomly sample $t$ negative nodes from $U_{UPG}$
12.            $U_{neg} \leftarrow U_{neg} \cup U_{neg}^{u'}$
13.    Compute $\nabla \tilde{L}_{(R)}$ according to Eq.(17) and update weights and biases.
14. return $\mathbf{H} = (\mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_n)$
```

5 Experiments

5.1 Datasets

We verified FEUI on two datasets: SNS dataset [3] and SZ dataset.

5.1.1 SNS dataset

The SNS dataset contains five popular social sites. We select Twitter and Myspace for experiments (see Table 3). We collect screen name, account name, location, etc. to construct attribute feature vectors. We symbolize each screen name and account name into a set of segments, then transform them to a real-valued vector via TF-IDF. For the training data, the positive instances are collected through Google Profile Service (GPS). All negative instances are sampled from candidate user pairs. We finally collect 24,515 user pairs (including 6495 matching pairs) from Twitter-Myspace dataset for training and testing.

Table 3 Statistics of the datasets

Datasets	Social Networks	Node	Link
SZ	Sina	12,1396	19,8992
	Zhihu	19,5349	136,8723
SNS	Twitter	40,171,624	1,468,365,182
	MySpace	854,498	6,489,736

5.1.2 SZ dataset

We collect the Sina Micro-blog dataset from the famous micro-blogging website in China. We also collect user information from Zhihu,[1] a famous Chinese Q&A website, including username, education, individual resume and location by a python crawler.[2] The details of SZ dataset (Sina & Zhihu) are summarized in Table 3. In our experiment, we select a pair of subgraphs from several identical users and extract first-order and second-order neighbors through a breadth-first search. We finally collect 7325 user pairs for training and testing, including 2213 manually labeled matching pairs. We choose username, gender, location, education, individual resume, etc. to construct attribute feature vectors.

5.2 Baseline methods

(1) SVM: The SVM model is trained based on the attributes (see Section 4.1) of labeled user pairs, and then adopt the trained model to determine the

[1] www.zhihu.com
[2] https://github.com/cpselvis/zhihu-crawler

label of each candidate user pair. We use libsvm with version 3.24[3] as the classifier.

(2) COSNET [3]: This method proposes a unified optimization framework by using an energy-based model, which captures both attributes and structure features. Different from the original procedure, we only consider local consistency part in our experiments.

(3) MEgo2Vec [6]: A novel embedding based model that generates matched ego networks for two users first, then output the embedding of the nodes in matched ego networks by combining the user attributes and the structure features.

(4) REGAL [24]:This method proposes a node representation algorithm xNetMF to utilize structure similarity and node attributes in an unsupervised way. The hyperparameters, such as the embedding dimension, the weights of structure and attribute similarity, are set to default values as depicted in original paper.

(5) SNNA [25]: An adversarial learning based method that discusses user matching through distribution level. This weakly-supervised method considers all the users as a whole and perform identification in a new perspective. For data pre-processing, we use ANRL [36] to jointly embedd the user attributes and topology information to a low-dimensional vector.

(6) Auto-encoder: We build an auto-encoder based on the attribute feature vectors for each network. To train the model, we minimize the distance between the embedding vectors of two nodes from a labeled user pair.

(7) Variants of our model: We implement five variants of our model to evaluate the performance of different modules.

Dual attribute embedding To evaluate the effect of dual attribute embedding component, we remove the joint embedding component. We denote this model as FEUI$_{\mathrm{de}}$. This model only combines the attribute features and the local structure features. We implement the embedding of two users $\mathbf{z}_i^A$ and$\mathbf{z}_i^B$ based on Fig. 4, then we apply a MLP f on the difference between their attribute embeddings $\mathbf{d}_i$ to predict the label value. Finally, we define the loss function as follows. If we do not use the average attributes of its neighborhoods$\mathbf{x}_{N_i}$ as an input, this model degrades to a traditional auto-encoder.

$$L=\frac{1}{n}\sum_{i=1}^{n} g(f(\mathbf{d}_i),y_i) \tag{18}$$

where $g(\cdot)$ represents cross-entropy function.

Joint embedding To evaluate the performance of joint embedding component, we only embed the global structure embedding (as illustrated in Fig. 5). In this way, we have three kinds of inputs for joint embedding. The first one is his/her adjacency vector, which is denoted as FEUI$_{\mathrm{aj}}$.. FEUI$_{\mathrm{aj}}$ only combines the label information and the global structure information. The second is the difference between there original attribute features$|\mathbf{x}_i^A-\mathbf{x}_i^B|$, which is denoted as FEUI$_{\mathrm{oa}}$. FEUI$_{\mathrm{oa}}$ combines the label, the global structure and the attribute information. The third one is the concatenation of their embedding attribute features$\left(\mathbf{z}_i^A,\mathbf{z}_i^B\right)$, which is denoted as FEUI$_{\mathrm{ea}}$. Besides, we remove the label prediction branch for topology embedding vector learning using the input $\mathbf{d}_i$, then we apply a MLP f on the node's learned embedding $\mathbf{h}_i$ to predict the label of the node. We denote it as FEUI$_{\mathrm{rl}}$, which captures the local structure, the global structure and the attribute information.

FEUI This model is our final model. It contains two modules. The input of joint embedding module is the output of dual attribute embedding $\mathbf{d}_i$.

We do not compare against the network embedding methods for other tasks. We implement COSNET, MEgo2VEC and REGAL with the source code released in original paper, and implement SNNA by ourselves. The parameters for the above baselines are tuned to be optimal. The embedding size is set as 128 for SNS dataset and 128 for SZ dataset in the dual attribute embedding module. The effect of the number of hidden layers are discussed in Section 5.5. In our comparison experiments with baselines, we use 7 hidden layers ($k=2$) in Fig. 4 (3 hidden layers for encoding, 1 fusion hidden layer, and 3 hidden layers for decoding). For the component of joint embedding, the embedding size is set as 64. We use 2 hidden layers for left output branch and 1 hidden layer for right output branch in our comparison experiments. Furthermore, we set window size w to 10 in context nodes generation (see Algorithm 1 and set negative samples t as 5. For the other parameters, such as ratio of label context α, size of mini-batch b_1 and b_2 in Algorithm 2, are tuned for the best performance. We perform our experiments on Tensorflow under a Linux server with 2.0 GHz CPU (Intel Xeon E5–2620).

5.3 User identification performance analysis

For two datasets, the candidate user pairs are selected when the username similarity is above a threshold.[4] Based on this, we construct several UPGs for each dataset since not all the candidate user pairs can construct a dense community. All matching pairs are considered as ground truth, and a fraction λ of them are selected to train different models. Table 4 shows the overall performance on two datasets with

[3] www.csie.ntu.edu.tw/~cjlin/libsvm

[4] The threshold is set to 0.7 in our experiment.

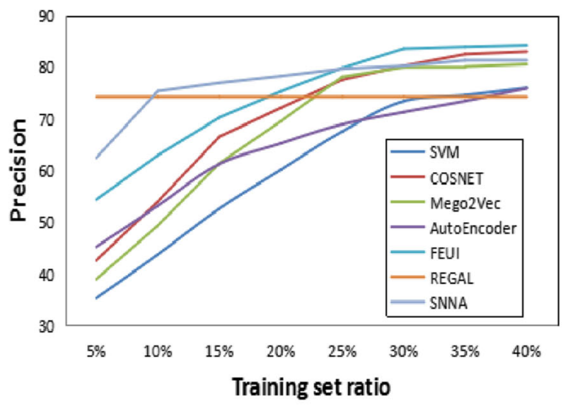
Table 4 User identification on two datasets

Method	Sina-Zhihu			Twitter-MySpace		
	Pre.	Rec.	F1	Pre.	Rec.	F1
SVM	73.42	68.23	70.73	78.35	70.29	74.10
COSNET	81.25	80.29	80.77	77.56	68.32	72.65
Mego2Vec	79.96	76.48	78.18	87.52	71.95	78.97
AutoEncoder	71.29	70.34	70.81	70.29	67.28	68.75
REGAL	74.25	70.43	72.29	78.23	68.54	73.06
SNNA	80.24	79.82	80.03	85.43	73.96	79.28
FEUI	**83.56**	**83.47**	**81.46**	**88.74**	**73.98**	**80.69**

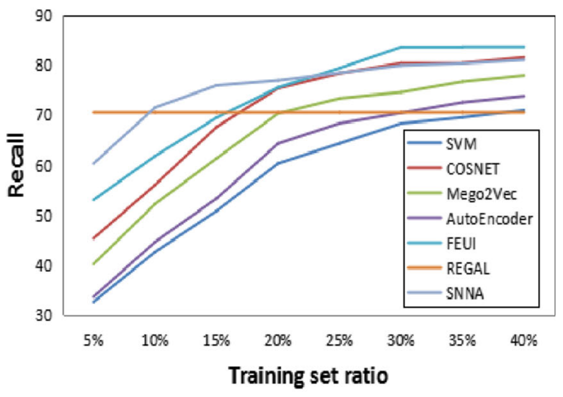
the training set ratio $\lambda = 30\%$. Also, we use training sets with λ ranging from 5% to 40% and conduct 5 times of experiments and record the average results in Fig. 6.

Overall prediction performance For these experiments, we set $\lambda = 30\%$ for training and then predict others as testing. It shows the superiority of FEUI over other methods. In terms of F1-score, FEUI achieves a 13% improvement over SVM and auto-encoder. Direct using of the original attribute vectors on SVM model would neglect the structure information and also suffer the sparse and poor representation problem. Yet, we observe that auto-encoder is a little weaker than SVM, which indicates the traditional dimension reduction procedure may lose some useful information. FEUI performs much better than other jointly modeling methods, such as COSNET, Mego2Vec and REGAL. Although COSNET utilizes both attribute information and structure information, it supposes that "neighborhood-preserving matching" is effective in the real world networks, which only considers the local structure information. MEgo2Vec seamlessly incorporates the local structure information into the neighbors' attributes embedding, but it can not incorporate the label information into the process of embedding. As an unsupervised algorithm, REGAL is simpler than the other semi-supervised methods, but at the cost of accuracy performance. SNNA performs identification by minimizing the wasserstein distance between two users form different networks, which leads to the sensitivity of performance on the quality of embedded vector.

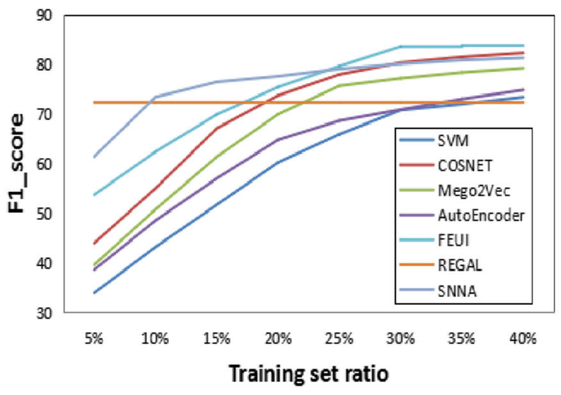
We also examine the effect of training set fractions by varying λ from 5% to 40%. In Fig. 6, the performance raises steadily when the training set ratio increases. FEUI performs best on both two datasets compared with other supervised and



(a) Precision



(b) Recall



(c) F1_score

Fig. 6 Performance comparison on the Sina-Zhihu dataset (a) Precision, (b) Recall, (c)F1_score

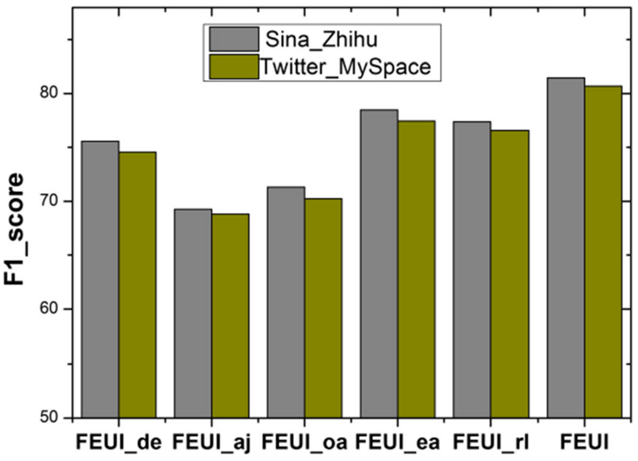


Fig. 7 User identification performance of model variants

semi-supervised methods, especially when the training set ratio is small. For example, when the training set ratio is 5%, FEUI outperforms 20% by SVM and outperforms 15% by auto-encoder on F1_score. This phenomenon indicates that our semi-supervised method can obtain better performance even with small ratio of ground truth. However, when the training ratio becomes greater, the increasement speed of FEUI slows down. That is to say, FEUI can lead to good performance with relatively less supervised information. Another interesting phenomenon is that COSNET performs slightly better than FEUI when the training set ratio is 40%. But it does not show that COSNET is more effective since it needs more training data. Most supervised methods can not achieve satisfactory accuracy with low training set ratio except SNNA. SNNA performs much better than other methods with low ratio, but the performance does not improve apparently when the training set ratio increases.

Besides, we show the performance of the variants of FEUI on two datasets in Fig. 7 and give a detailed analysis as follows.

Dual attribute embedding effect To verify the performance of dual attribute embedding module, we first compare FEUI$_{oa}$ with FEUI. Compared with FEUI$_{oa}$, we add dual attribute embedding modules in FEUI, while FEUI$_{oa}$ uses original attribute difference as the input of joint embedding. We find that FEUI performs much better than FEUI$_{oa}$. This results indicate that the user attribute information captured by dual attribute embedding model can indeed benefit the task. Secondly, we compare the model FEUI$_{de}$ with FEUI$_{aj}$. FEUI$_{de}$ performs much better then FEUI$_{aj}$ on both two datasets, which demonstrates that the jointly embedding of attribute information and local neighborhood structure information can improve our performance. Thirdly, we compare two variants of joint embedding module, FEUI$_{oa}$ and FEUI$_{aj}$. We see that FEUI$_{oa}$ performs better than FEUI$_{aj}$. This is due to that FEUI$_{oa}$ combines the label, global structure and attribute information, while FEUI$_{aj}$ does not incorporate the attribute information. This comparison shows that the attribute features of users indeed play an important role for our tasks.

Joint embedding effect First, we compare FEUI with FEUI$_{de}$. Experimental results show that FEUI outperforms FEUI$_{de}$ on two datasets, which shows that adding the joint embedding module can improve the performance, and also demonstrates that the incorporation of global structure and label information can benefit for our task. Compared with FEUI$_{de}$, FEUI achieves an improvement of 5.9% on Twitter-Space dataset and an improvement of 6.2% on Sina-Zhihu dataset, while compared with FEUI$_{oa}$, FEUI achieves an improvement of 10.1% and 10.4% on Sina-Zhihu dataset and Twitter-Space dataset respectively. The above comparison shows that attribute features are more effective for user identification than structure features, although some users' attributes are incomplete and noisy. Second, we compare FEUI$_{de}$ with FEUI$_{oa}$. FEUI$_{de}$ perfroms superior to FEUI$_{oa}$, which shows that the performance of dual attribute module performs better than joint embedding module. Moreover, the performance of FEUI$_{de}$ on Sina-Zhihu is superior to that on Twitter-MySpace since attribute features contribute less on Twitter-MySpace than on Sina-Zhihu. We deepen the analysis and find that we can obtain more attributes from Sina-Zhihu dataset than from Twitter-MySpace dataset.

Besides the above comparisons, we conduct other comparisons. FEUI$_{ea}$ performs superior to FEUI$_{oa}$, which demonstrates that the user attribute features extracted by dual attribute embeeding model are more effective than original attribute features. FEUI performs superior to FEUI$_{rl}$, which indicates that the incorporation of label information can also benefit for our task and improve the representation ability of embedding vector. Among all of these variants, FEUI$_{aj}$ performs the worst. This is because it does not consider attribute and label information.

In the following, we highlight some observations:

(1) The dual attribute embedding module and the joint embedding module are all benefit for our task. If we combine these two modules together, we tend to enhance the quality of embedding and obtain better performance. For individual information sources, the improvement by attribute information is more apparent than structure information and label information.
(2) FEUI performs best compared against other network embedding based methods and traditional baselines. One major reason for the performance enhancement is the systematic fusion of heterogeneous information, which can achieve quality embeddings. Based on this, the performance of user identification can be improved by jointly modeling of label prediction and embedding.

5.4 Node classification

We conduct node classification to verify the embedding effectiveness of our algorithm. We compare FEUI with other state-of-the-art embedding algorithms, and divide them into several groups:

(1) Attribute-only: The algorithms in this group only consider vector attributes. We adopt SVM and the traditional auto-encoder in our experiment.
(2) Structure-only: The algorithms in this group only consider structure features. We adopt DeepWalk, Dwns_AdvT [37] and SDNE [28] in our experiment. Dwns_AdvT denotes the implementation of DeepWalk with the regularization by adversarial training process.
(3) Attribute+structure: The algorithms in this group consider both attribute and structure. We adopt SEANO [34] and ANRL [36] in our experiment.
(4) FEUI:FEUI is not suitable to be directly used for node classification. We need to modify the dual attribute embedding module for one social network, not for two social networks. Suppose that the input network isG^A, we can obtain the embedding vector$\mathbf{z}_i^A$. Then, we set$\mathbf{z}_i^A$ as the input of joint embedding module.

For all the baselines, we use the source code released in the original paper. Also, the parameters are tuned to be optimal. We adopt three benchmark datasets for experiments, which are Cora, Citeseer [38] and Pubmed [34]. The embedding size is set to 64 for all datasets. The classification accuracy of all these algorithms are shown in Fig. 8. It shows that: (1) FEUI consistently outperforms other algorithms. The major reason is that our framework systematically incorporates user attributes, label information and models the local and global structure properties, which can generate a better embedding space than other methods. (2) Attribute+structure algorithms perform better than structure-only and attribute-only methods, which is consistent with our user identification task. This verifies the combination of important factors can generate a more representative embedding results. Although adversarial training can bring improvement on DeepWalk, Dwns_AdvT is still worse than the joint modeling methods. (3) Although ANRL shows a relative good performance, it neglects the label factor, which makes the accuracy be slightly worse than our algorithm.

Table 5 Performance on two datasets with different number of layers in two modules

Modules	Layers	SZ	SNS
Joint embedding	$k=1, k'=1, m=1$	80.28	79.83
	$k=2, k'=1, m=1$	81.12	80.13
	$k=1, k'=2, m=1$	80.31	79.82
	$k=2, k'=2, m=1$	81.43	80.62
	$k=2, k'=1, m=2$	81.46	80.64
	$k=4, k'=1, m=3$	81.45	80.66
Dual attribute embedding	$k=1$	80.87	80.14
	$k=2$	81.46	80.69
	$k=3$	81.48	80.69

5.5 Experiments on the number of layers

We discuss the influence of number of layers on dual embedding module and joint embedding module. It has to be proved that increasing the number of hidden layers may increase the generalization performance [39], but it may also cause high running time and learning difficulties, which will result in the performance degradation [40]. Thus, it is necessary to explore the optimal number of layers for two modules(Table 5).

As shown in Fig. 4 and Fig. 5, the value of$k$, $k'$and$m$(see Fig. 4 and Fig. 5) determine the FEUI model's representation ability. When we test the layers for joint embedding, we set

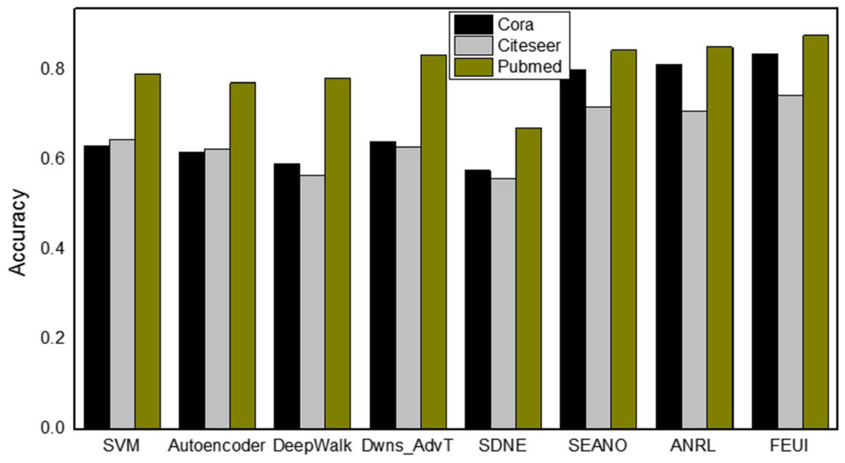


Fig. 8 Classification accuracy of different algorithms

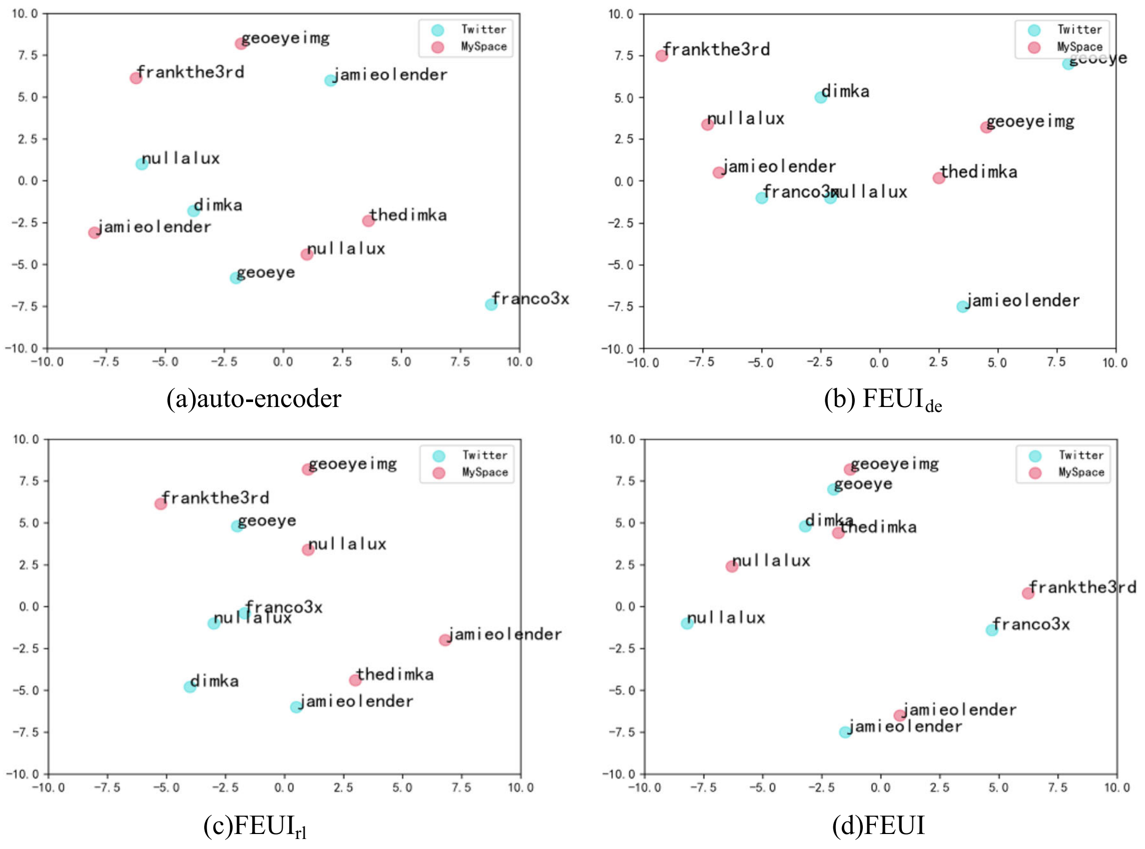


Fig. 9 Case study of learned embeddings for positive instances. (a)auto-encoder, (b) FEUI$_{de}$, (c)FEUI$_{rl}$, (d)FEUI

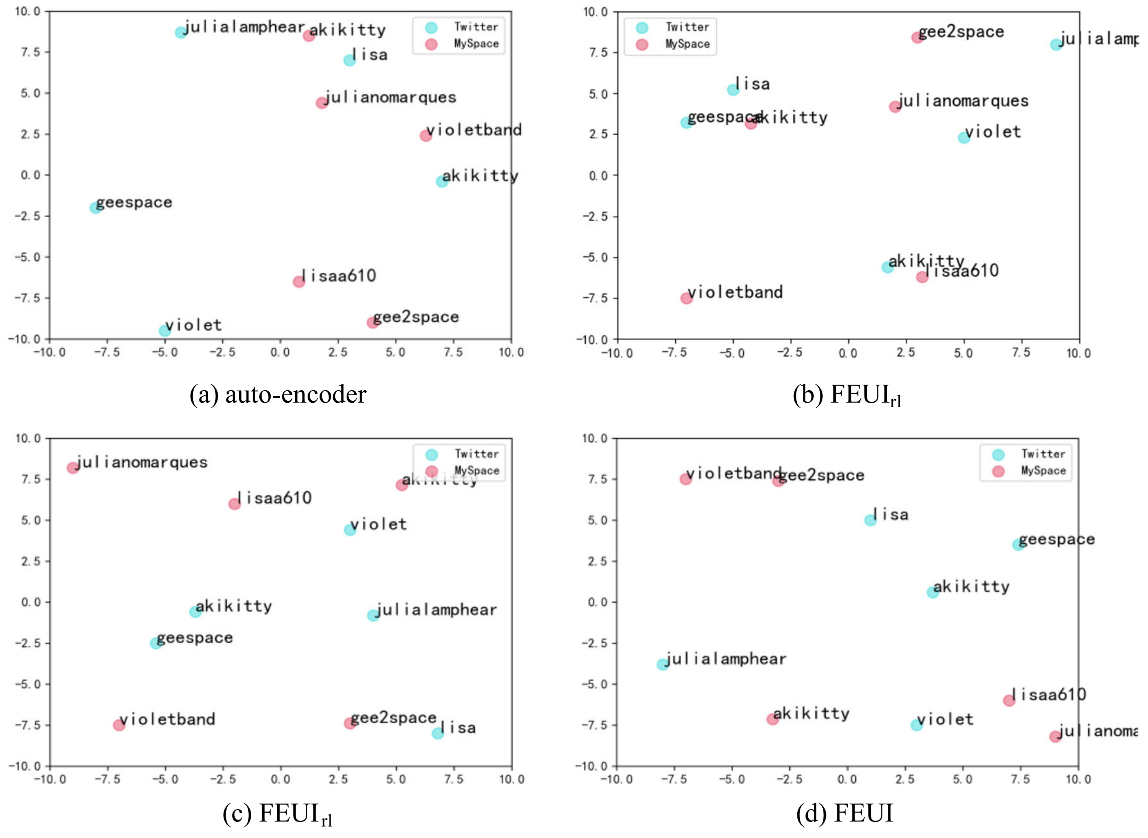


Fig. 10 Case study of learned embeddings for negatives instances, (a)auto-encoder, (b) FEUI$_{de}$, (c)FEUI$_{rl}$, (d)FEUI(In figure 10, the title of Fig10(b) should be FEUI$_{de}$

the number of layers in dual attribute embedding module to $k = 2$. Also, the number of layers in joint embedding is set to $k = 2$, $k' = 1$, $m = 2$ when we test on dual attribute embedding module. First, we can see the trend that the performance can slightly be improved with more layers on two modules. This indicates that the deeper neural networks can improve the representation ability. For joint embedding, the increasement of the number layer of right output branch k' has no positive influence on SNS dataset. Although the increasement of layers can improve the performance, it will increase the time consuming for the training procedure. Specifically, it takes about 30.5 s for the setting $k = 1, k' = 1, m = 1$ for one epoch, while it takes 75.8 s for the setting $k = 2, k' = 1, m = 2$ for one epoch. It has to be noted that the further increasement of layers may not improve the results. This is due to that the fully connection structure of our model may be easily over-fitting with more layers. Thus, we do not explore the effect of more layers in our experiments.

5.6 Case study

The quality of the learned embedding vector can directly determine the performance of label prediction. Here we conduct several cases about the node embeddings by several variants. Figure 9 and Fig. 10 visualize the node embeddings based on auto-encoder, FEUI_{de}, FEUI_{rl} and FEUI on two datesets. We adopt t-SNE [41] method for visualization. Figure 9 shows the learned embeddings for positive instances and Fig. 10 shows the results for negative instances. We randomly select 10 user pairs with neighbors(5 positive instances and 5 negative instances) for visualization. Specifically, (‘jamieolender’,‘jamieolender’), (‘nullalux’,‘nullalux), (‘dimka’,‘thedimka’), (‘geoeye’,‘geoeyeimg’), (‘franco3x’,‘frankthe3rd’) are positive instances. (‘lisa’,‘lisaa610’), (‘violet’,‘violetband’), (‘julialamphear’,‘julianomarques’), (‘geespace’,‘gee2space’), (‘akikitty’, akikitty’) are negative instances. The green circles represent the users in Twitter, and the red circles are their identities in MySpace. As shown in Fig. 9(a) and Fig. 9(b), the embeddings of two users in a pair become much more closer after we add the local information. Also, the embeddings become a little closer after we add the global structure information (see Fig. 9(c)). Furthermore, as shown in Fig. 9(d), the embedding of two users in a pair become closer after the incorporation of label information. The improvement by adding label information is more apparent compared with the improvement by network structures. This may be due to that the selected instances contain small number of neighbors. This observation verifies the apparent benefit of incorporating label information. However, the above improvements do not appear in the results of negative instances in Fig. 10. This results indicate that the label information, the structure information do help the embedding learning.

6 Conclusion

We propose a novel semi-supervised framework FEUI to predict whether two users are the same user or not by embedding the generated user-pair-oriented graph, which is generated from two isolated social networks. Our proposed model is the first attempt to solve the user identification task and user embedding learning simultaneously, and investigates the possibility of jointly embedding several important information sources into a unified space, not only structural and attribute information. It seamlessly combines attributes and the local structure in the dual attribute embedding model, and also captures both the global structure features and the labels in the joint embedding model. The experimental results on two real-world datasets validate the effectiveness of our model.

In our future work, we plan to study the following directions. First, the users in social networks have multiple modalities, so we will enhance our model to jointly embed multi-modal data. Second, most social networks have the nature of evolution, we can improve our model to adapt to the dynamic changes, such as the new follower-ship and new users.

Acknowledgements This study is supported by Zhejiang Provincial Natural Science Foundation of China under Grant No.LY19F020022, China Knowledge Centre for Engineering Sciences and Technology(CKCEST), Zhejiang Provincial Natural Science Foundation of China under Grant No.LHY21E090004.

References

1. Zafarani R, Liu H (2016) Users joining multiple sites: friendship and popularity variations across sites[J]. Information Fusion 28:83–89
2. Jain P, Kumaraguru P, Joshi A (2013) @ i seek'fb. me' identifying users across multiple online social networks[C]//Proceedings of the 22nd international conference on World Wide Web. 1259–1268
3. Zhang Y, Tang J, Yang Z, et al. (2015) Cosnet: Connecting heterogeneous social networks with local and global consistency[C]// Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 1485–1494
4. Bartunov S, Korshunov A, Park S T, et al. (2012) Joint link-attribute user identity resolution in online social networks[C]//proceedings of the 6th international conference on knowledge discovery and data mining, workshop on social network mining and analysis. ACM
5. Man T, Shen H, Liu S, et al. (2016) Predict anchor links across social networks via an embedding approach[C]//IJCAI. 16: 1823-1829
6. Zhang J, Chen B, Wang X, et al. (2019) Mego2vec: Embedding matched ego networks for user alignment across social networks[C]//Proceedings of the 27th ACM International Conference on Information and Knowledge Management. 327–336
7. Liu D, Wu QY, Han WH et al (2015) User identification across multiple websites based on username features[J]. Chinese J Comput 38(10):2028–2040
8. Liang W, Meng B, He X et al (2015) GCM: a greedy-based cross-matching algorithm for identifying users across multiple online

social networks[C]//Pacific-Asia workshop on intelligence and security informatics. Springer, Cham, pp 51–70
9. Wang M, Tan Q, Wang X, et al. (2018)De-anonymizing social networks user via profile similarity[C]//2018 IEEE third international conference on data science in cyberspace (DSC). IEEE, 889–895
10. Narayanan A, Shmatikov V (2009)De-anonymizing social networks[C]//2009 30th IEEE symposium on security and privacy. IEEE, 173–187
11. Zhou X, Liang X, Zhang H et al (2015)Cross-platform identification of anonymous identical users in multiple social media networks[J]. IEEE Trans Knowl Data Eng 28(2):411–424
12. Zhou X, Liang X, Du X et al (2017) Structure based user identification across social networks[J]. IEEE Trans Knowl Data Eng 30(6):1178–1191
13. Korula N, Lattanzi S (2014) An efficient reconciliation algorithm for social networks[J]. Proc VLDB Endow 7(5):377–388
14. Li Y, Su Z, Yang J, Gao C (2020) Exploiting similarities of user friendship networks across social networks for user identification[J]. Inf Sci 506:78–98
15. Lee R K W, Hee M S, Prasetyo P K, et al. (2018) Linky: visualizing user identity linkage results for multiple online social networks[C]// 2018 IEEE international conference on data mining workshops (ICDMW). IEEE, 1453–1458
16. Liu S, Wang S, Zhu F (2015) Structured learning from heterogeneous behavior for social identity linkage[J]. IEEE Trans Knowl Data Eng 27(7):2005–2019
17. Liu X, Chen Y, Fu J (2020) MFRep: joint user and employer alignment across heterogeneous social networks[J]. Neurocomputing 414:36–56
18. Fu H, Zhang A, Xie X (2015) Effective social graph deanonymization based on graph structure and descriptive information[J]. ACM Trans Intell Syst Technol (TIST) 6(4):1–29
19. Dong Y, Hu Z, Wang K, et al. (2020) Heterogeneous network representation learning[C]. IJCAI
20. Tu C, Zeng X, Wang H et al (2018) A unified framework for community detection and network representation learning[J]. IEEE Trans Knowl Data Eng 31(6):1051–1065
21. Yang C, Xiao Y, Zhang Y, et al. (2020) Heterogeneous Network Representation Learning: Survey, Benchmark, Evaluation, and Beyond[J]. arXiv preprint arXiv:2004.00216
22. Wang Y, Feng C, Chen L, Yin H, Guo C, Chu Y (2019) User identity linkage across social networks via linked heterogeneous network embedding[J]. World Wide Web 22(6):2611–2632
23. Liu L, Cheung W K, Li X, et al. (2016) Aligning Users across Social Networks Using Network Embedding[C]//IJCAI. 1774–1780
24. Heimann M, Shen H, Safavi T, et al. (2018) Regal: Representation learning-based graph alignment[C]//Proceedings of the 27th ACM international conference on information and knowledge management. 117–126
25. Li C, Wang S, Wang Y, et al. (2019) Adversarial learning for weakly-supervised social network alignment[C]//Proceedings of the AAAI Conference on Artificial Intelligence. 33(01): 996–1003
26. Yang C, Liu Z, Zhao D, Sun M and Chang E Y (2015) Network representation learning with rich text information. In IJCAI, 2111–2117
27. Wang L, Hu K, Zhang Y, Cao S (2019) Factor graph model based user profile matching across social networks[J]. IEEE Access 7: 152429–152442
28. Wang D, Cui P, Zhu W (2016) Structural deep network embedding[C]//Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining. 1225–1234
29. Zhang J, Xia C, Zhang C, et al. (2017) BL-MNE: emerging heterogeneous social network embedding through broad learning with aligned AutoEncoder[C]//2017 IEEE international conference on data mining (ICDM). IEEE, 605–614
30. Jin D, Li B, Jiao P, et al. (2019)Network-Specific Variational Auto-Encoder for Embedding in Attribute Networks[C]//IJCAI. 2663–2669
31. Liao L, He X, Zhang H, Chua TS (2018) Attributed social network embedding[J]. IEEE Trans Knowl Data Eng 30(12):2257–2270
32. Balikas G, Amini M R (2016) An empirical study on large scale text classification with skip-gram embeddings[J]. arXiv preprint arXiv: 1606.06623
33. Dunbar RIM, Arnaboldi V, Conti M, Passarella A (2015) The structure of online social networks mirrors those in the offline world[J]. Soc Networks 43:39–47
34. Liang J, Jacobs P, Sun J et al (2019)Semi-supervised embedding in attributed networks with outliers[C]//proceedings of the 2018 SIAM international conference on data mining. Soc Ind Appl Math:153–161
35. Mikolov T, Sutskever I, Chen K, et al. (2013) Distributed representations of words and phrases and their compositionality[C]// Advances in neural information processing systems. 3111–3119
36. Zhang Z, Yang H, Bu J, et al. (2019) ANRL: Attributed Network Representation Learning via Deep Neural Networks[C]//IJCAI. 18: 3155–3161
37. Dai Q, Shen X, Zhang L, et al. (2019) Adversarial training methods for network embedding[C]//The World Wide Web Conference. 329–339
38. Z. Yang, W. Cohen, and R. Salakhutdinov. Revisiting semi-supervised learning with graph embeddings. ICML'16
39. He X, Liao L, Zhang H, et al. (2017) Neural collaborative filtering[C]//Proceedings of the 26th international conference on world wide web. 173–182
40. Glorot X, Bengio Y (2010) Understanding the difficulty of training deep feedforward neural networks[C]//Proceedings of the thirteenth international conference on artificial intelligence and statistics. 249–256
41. Van der Maaten L, Hinton G (2008) Visualizing data using t-SNE[J]. J Mach Learn Res, 9(11)

Lidong Wang was born on December, 4, 1982. She received her Ph.D. degree from the College of Computer Science and Technology, Zhejiang University. She is currently an Associate Professor at Hangzhou Normal University.Her current research interests include text mining, image processing, and machine learning.