Collaborative Cross-Network Embedding Framework for Network Alignment

Hai-Feng Zhang, Guojing Ren, Xiao Ding, Li Zhou, and Xingyi Zhang, *Senior Member, IEEE*

***Abstract*—Network alignment aims to identify the corresponding nodes belonging to the same entity across different networks, which is a fundamental task in various applications. Existing embedding-based approaches usually involve two stages, namely embedding and matching. The embedding stage conducts network embedding on each network to capture the primary structural regularity. In the matching stage, a mapping function is built to project the learned embeddings to the same latent space. However, these approaches typically encounter two challenges: (1) the difficulty of unifying the elusive embedding spaces to the same latent space; (2) the difficulty of distinguishing the real anchor nodes from their neighbors, resulting in the confounding matching problem. To address these challenges, we present the Collaborative Cross-Network Embedding (CCNE) framework in this paper. This framework provides a collaborative and straightforward paradigm to better unify the two networks to the same latent space by preserving both intra- and inter-network structural features, without the need for a carefully designed mapping function. Meanwhile, a hard negative sampling strategy is adopted to distinguish anchor nodes from their sampled neighbors. Furthermore, an iterative CCNE is proposed to alleviate the scarcity of observed anchor links. Extensive experiments on real social networks demonstrate that the proposed collaborative framework outperforms current embedding-matching methods in terms of accuracy, robustness as well as compatibility.**

***Index Terms*—Cross-network embedding, graph neural networks, hard negative sampling, network alignment.**

I. Introduction

ONLINE social networks (OSNs) have become more and more popular in recent years, and people usually participate in multiple OSNs to access diverse services because these OSNs often provide different services [1]. For instance, users may register different accounts on both Instagram and Facebook, and they share pictures through Instagram, while chatting with friends through Facebook. The same users across multiple OSNs are termed anchor users and the links connecting pairs of anchor users are called anchor links. These anchor users/links often provide rich useful data for network analysis and the downstream tasks, such as cross-network recommendation [1], influential user identification [2], community detection [3], information propagation [4], and user behavior prediction [5]. However, in the real situation, the known anchor links are extremely few owing to the privacy protection from users themselves or the non-sharing between OSNs, therefore, it is necessary and significant to develop approaches to identify the anchor users across multiple OSNs. And this problem has attracted increasing attention in recent years, which is often defined by different terminologies, including Network Alignment (NA) [6], Anchor Link Prediction (ALP) [7], and User Identity Linkage (UIL) [8].

Early studies utilize user profiles (user names, locations, genders, etc.) to address NA problem [9], [10]. Some papers leverage user-generated content (UGC), such as tweets, posts, blogs, reviews, and ratings [11], [12]. On the one hand, user profiles are often unavailable due to privacy considerations. On the other hand, UGC often contains heterogeneous forms (photos, videos, texts, etc.), resulting in the challenge of representing them. Therefore, attribute-based NA approaches are sometimes disappointing. In many situations, the connection information in social networks is relatively easier to obtain, so many approaches make use of structural information to solve the problem of NA [13], [14]. Inspired by the recent advances in graph representation technology [15], network embedding-based methods have become a mainstream trend to study the NA problem. Existing embedding-based methods usually contain two stages, namely, embedding stage and matching stage [16], [17]. As illustrated in Fig. 1, the first stage is to extract the major structural features of each network via various elaborately designed embedding methods, including different shallow and deep embedding methods. In the matching stage, the mapping function is constructed to project the learned embeddings to the same latent space, in which candidate counterparts are sorted and matched according to pairwise similarity. Nevertheless, the structures and the behaviors of users in different OSNs are significantly different, it is challenging to unify them to the same latent space. Besides, there is a collision between the objectives of the embedding and matching stage. In the embedding stage, proximity nodes (e.g., neighbor nodes) are desired to be as close as possible in the embedding space, e.g., node v_4^β in network $\mathcal{G}^\beta$ should be close to its neighbor node v_5^β (see Fig. 1(a)). In the matching stage, it

Manuscript received 22 September 2023; revised 8 December 2023; accepted 13 January 2024. Date of publication 18 January 2024; date of current version 30 April 2024. This work was supported in part by the National Natural Science Foundation of China under Grant 61973001, in part by the University Synergy Innovation Program of Anhui Province under Grant GXXT-2021-032, and in part by the National Key Research and Development Project, Ministry of Science and Technology, China, under Grant 2018AAA0101300. Recommended for acceptance by Dr. Bo Yang. *(Corresponding author: Hai-Feng Zhang.)*

Hai-Feng Zhang and Xiao Ding are with the Key Laboratory of Intelligent Computing and Signal Processing of Ministry of Education, School of Mathematical Science, Anhui University, Hefei 230601, China (e-mail: haifengzhang1978@gmail.com; xiaoding2021@stu.ahu.edu.cn).

Guojing Ren is with the Institutes of Physical Science and Information Technology, Anhui University, Hefei 230601, China (e-mail: rengj@stu.ahu.edu.cn).

Li Zhou is with the College of Science, Anhui Agricultural University, Hefei 230036, China (e-mail: lizhou@ahau.edu.cn).

Xingyi Zhang is with the School of Computer Science and Technology, Anhui University, Hefei 230601, China (e-mail: xyzhanghust@gmail.com).

Digital Object Identifier 10.1109/TNSE.2024.3355479

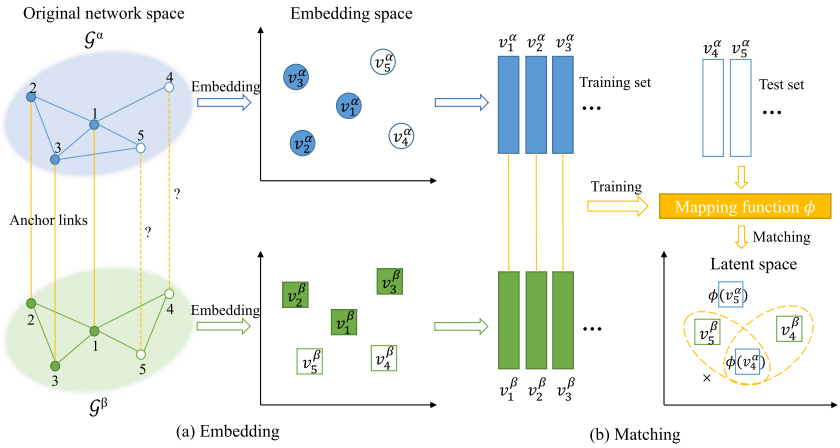


Fig. 1. Architecture of the traditional two-stage methods for NA problem. (a) embedding two networks to representation vectors; (b) training a mapping function to match them in the same latent space.

is desirable to distinguish anchor nodes from their proximities in the latent space. For example, a precise matching mechanism should distinguish the pair $(\phi(v_4^\alpha), v_4^\beta)$ and pair $(\phi(v_4^\alpha), v_5^\beta)$ when one-to-one alignment is considered. However, v_4^β and v_5^β are needed to be as close as possible in the embedding stage, leading to the confounding matching problem (see Fig. 1(b)).

In this paper, we propose a Collaborative Cross-Network Embedding (CCNE) framework to address the above challenges, in which node representations designed for NA are learned by collaboratively optimizing an objective function consisting of intra- and inter-network loss. And the neighbors of each anchor node are sampled as hard negatives to distinguish anchor nodes from their neighbors, which could alleviate the confounding matching problem. Compared with many embedding-matching methods, the proposed framework has no need for a carefully designed mapping function, as the embedding spaces of networks become close gradually during the collaborative training process. In addition, an Iterative Collaborative Cross-Network Embedding (labeled as ICCNE) framework is proposed to solve the problem of scarcity of anchor links. The main contributions are summarized as:

- We propose a novel CCNE framework for NA problem, which builds a collaborative paradigm to learn node representations consisting of intra- and inter-network structural features. Our CCNE framework is actually one stage method, which has no need for a carefully designed mapping function, since the embedding spaces of networks become close automatically during the collaborative training process.
- To alleviate the confounding matching problem, we devise a constraint mechanism based on the fact that the distance between anchor nodes should be shorter than the distance between an anchor node and its neighbors. As a result, we randomly sample a node for each anchor from its neighbors as hard negative.
- We further introduce an ICCNE framework to alleviate the scarcity of observed anchor links. In each iteration, candidate anchor links with higher similarity scores are selected and added to the known anchor links. Performance is enhanced with the increase of known anchor links.
- Extensive experiments on real social networks demonstrate that the proposed collaborative framework outperforms the baselines in terms of effectiveness and robustness. We also verify the compatibility of our framework by using different embedding methods to act as the encoders.

II. Related Work

In this section, we present a brief overview of the NA methods and organize them into three categories: i) attribute-based NA, ii) structure-based NA, and iii) Attribute and structure-based NA.

A. Attribute-Based NA

Early studies utilize user profiles (user names, locations, genders, etc.) to address the NA problem. Zafarani et al. [18] developed a behavioral modeling approach that exploits behavioral patterns exhibited by users when choosing usernames. Liu et al. [10] analyzed the rareness of usernames to help identify anchor users. Since username-only methods face limitations due to the duplication of popular usernames and randomly generated usernames provided by OSNs, researchers pay more attention to multi-attribute methods. Iofciu et al. [9] established a novel metric to assess the similarity of users by integrating user tags and usernames as features. Mu et al. [19] investigated the latent user space and pointed out that accounts of different OSNs can be projected back to the latent user space, where projections of the same user should be close to each other.

Some papers leverage user-generated content (UGC), such as blogs, tweets, reviews, posts, and ratings to identify anchor users across multiple OSNs. Goga et al. [11] identified anchor users with geo-locations, timestamps of posts, and writing style captured by language models. Nie et al. [12] built core interests of users from user-generated articles and proposed a dynamic mapping algorithm to identify anchor users. Kong et al. [20] extracted heterogeneous features from multiple OSNs for NA,

in which text features captured from tweets and tips can improve the performance of inferring anchor links. However, with the increasing awareness of privacy, high-quality user profile information is difficult to obtain. At the same time, it is difficult to uniformly represent UGC because these contexts may be in heterogeneous forms, such as photos, videos, and texts.

B. Structure-Based NA

The connection information in social networks is relatively easier to be obtained, so many approaches make use of structural information of networks to predict the anchor users across multiple OSNs. Kollias et al. [13] introduced an uncoupling and decomposing-based method, which can reduce the cost of similarity computation. Korula et al. [21] mathematically formalized the NA problem and postulated that different OSNs are randomly generated from a true underlying graph. Tan et al. [22] used hypergraphs to model high-order relationships and embedded users into a common low-dimensional space by manifold alignment. Zhou et al. [23] assumed that anchor users tend to have similar friendship structures in different OSNs and calculated a matching degree for all candidate anchor links. Zhang et al. [24] proposed a collective link fusion model that integrates intra-network random walks and inter-network random walks. Ding et al. [25] proposed a degree penalty algorithm to preserve second-order common matched neighbors. Feizi et al. [26] proposed a spectral decomposition method for graph alignment. Ding et al. [27] proposed a naive Bayes-based model which can measure the conditional probabilities of matched anchor links and then iteratively predict unmatched anchor links.

Inspired by recent advances in graph representation [15], network embedding-based methods have become a mainstream trend. Man et al. [16] applied network embedding for two OSNs to preserve structural features and learned a mapping function across the two embedding spaces for NA. Liu et al. [28] represented the follower/followee-ship of each user as input and output context vectors and developed a network embedding model to preserve the proximity of users with similar contexts. Wang et al. [29] proposed to embed different OSNs into a Hamming space of binary codes to solve the challenge of scalability. Zhou et al. [8] exploited a dual learning-based paradigm, consisting of an unsupervised pretraining step and a supervised step based on reinforcement learning. Zhou et al. [30] also leveraged the idea of generative adversarial nets (GAN) [31] to optimize the generative mapping function, after representing nodes with the discriminative graph convolutional embedding. Zheng et al. [32] proposed a cycle-consistent adversarial mapping model to optimize the mapping functions with energy-based GAN as well as cycle-consistent training. Zhang et al. [33] proposed a semi-supervised method to strike a balance between alignment consistency and disparity. Du et al. [34], [35] proposed a Skip-gram embedding model based on biased random walks across two networks. Tang et al. [36] developed a cross-graph embedding method to reflect structural proximity and positional proximity simultaneously. Tang et al. [37] also presented a multi-order matched neighborhood consistency-based method to match nodes by aligning the learned node embeddings with only a small number of pseudo alignment seeds.

C. Attribute and Structure-Based NA

Some works utilize both attribute and structure features to better handle the NA Problem. Zhang et al. [14] proposed an energy-based model to capture both the local and global consistency across multiple OSNs and solved it with a subgradient method. Ren et al. [38] studied attributed heterogeneous social networks and defined the inter-network meta diagram for anchor link feature extraction. Zhang et al. [39] leveraged attribute information to instruct the structure-based alignment process, and developed a series of algorithms to efficiently optimize the problem. Huynh et al. [40] leveraged holistic embeddings for NA, including proximity, attribute, and community-based embedding. Zhang et al. [41] designed a novel constrained dual embedding model, which integrates attribute and structure embedding for reconciliation. Saxena et al. [42] proposed a contrastive learning model for unsupervised network alignment. Park et al. [43] augmented node attributes based on nodes' centrality measures and gradually discovered node pairs. Tang et al. [44] proposed a representation-based alignment model to integrate topology consistency and attribute consistency synergistically.

Since attribute features are compatible with structure features, i.e. attribute and structure features can be integrated by concatenating the feature vectors, we mainly focus on network embedding-based methods for universality. Most embedding-based methods contain embedding and matching stages and have achieved promising results. However, embedding-based methods also have some shortcomings. For instance, owing to the structural complexity and non-linear properties in networks with different semantics, it is very challenging to unify the elusive embedding spaces well. Furthermore, distinguishing the anchor nodes from their neighbors lacks consideration in previous works. In view of this, we will introduce our CCNE framework in the next sections to overcome the existing shortcomings.

III. Preliminaries and Problem Formulation

In this section, we will define several important concepts and notations used in this paper, and provide the formulation of the NA problem. For simplicity, we only consider two networks for alignment.

Definition 1 (Graph): A graph (or network) is denoted as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. $\mathcal{V}$ is the set of vertices (or nodes), and each node represents a user. $\mathcal{E}$ is the set of edges, and each link $(v_i, v_j) \in \mathcal{E}$ indicates a relationship between the users v_i and v_j. $\boldsymbol{A}$ is the adjacency matrix of $\mathcal{G}$.

Definition 2 (Anchor link): For two networks $\mathcal{G}^\alpha$ and $\mathcal{G}^\beta$, the corresponding nodes belonging to the same entity are defined as anchor nodes, and the relationships among anchor nodes are called anchor links. As demonstrated in Fig. 1, v_1^α and v_1^β belonging to the same user u_1 are anchor nodes, and pair of nodes (v_1^α, v_1^β) is the anchor link. All anchor links are denoted as $\mathcal{L}$, where $\mathcal{L} = \{(v_i^\alpha, v_j^\beta) | v_i^\alpha \in \mathcal{V}^\alpha, v_j^\beta \in \mathcal{V}^\beta\}$.

Definition 3 (Network alignment): Given two networks ($\mathcal{G}^\alpha$ and $\mathcal{G}^\beta$) and a set of observed anchor links $\mathcal{L}$, the task of network alignment (NA) is to infer the matching regulations between $\mathcal{V}^\alpha$

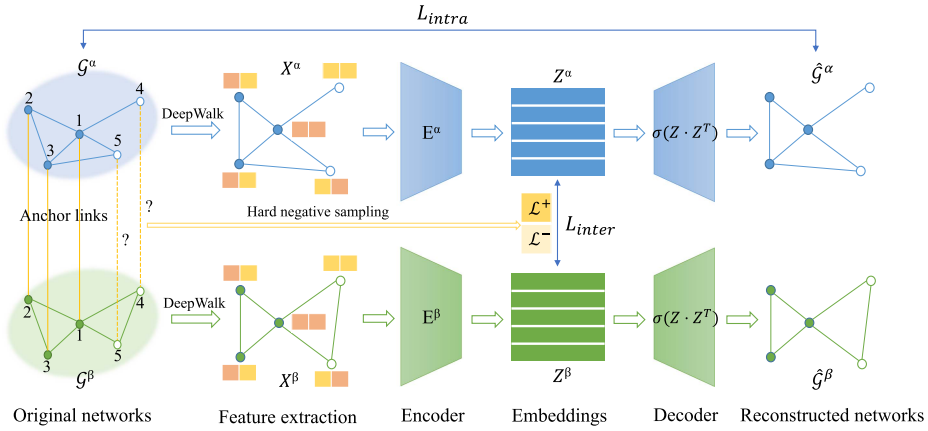


Fig. 2. Proposed collaborative cross-network embedding framework.

and $\mathcal{V}^\beta$:

$$\Psi : \mathcal{V}^\alpha \rightarrow \mathcal{V}^\beta. \tag{1}$$

For each v^α in $\mathcal{V}^\alpha$, Ψ will return the corresponding anchor node v^β in $\mathcal{V}^\beta$. In practice, Ψ would return a candidate matching list for each unmatched node, which is similar to recommendation systems.

IV. The Proposed Method

A. Main Architecture of CCNE Framework

The CCNE framework mainly includes encoder, decoder, and objective function modules (see Fig. 2). In the encoder module, we first use two independent encoders to capture intra-network structural features. In the decoder module, we use the two decoders to reconstruct the networks, such that each reconstructed network is as close as possible to its original network. In the objective function module, several objectives are considered: 1) minimizing the diversity between the reconstructed networks and original networks to preserve intra-network structural information; 2) minimizing the distance between anchor nodes to preserve the inter-network structural features; 3) introducing a hard negative sampling strategy to constrain the distance between anchor nodes and their sampled neighbors, and help to alleviate the confounding matching problem. Within this framework, both the intra- and inter-network structural features are preserved by optimizing the total objective function consisting of intra- and inter-network loss, prompting that the two networks are better embedded in the same latent space. The detailed descriptions of each module are given as follows:

1) Encoder Module: The encoder module includes two encoders E^α and E^β for $\mathcal{G}^\alpha$ and $\mathcal{G}^\beta$, respectively. We use Graph Convolutional Network (GCN) as the encoder since it has shown promising results in network representation. We employ DeepWalk [45] to extract low dimensional embedding of networks as the input features of GCN because we assume that attribute features of nodes are not available. Previous studies [46] have shown that GCN achieves the best performance with 2-3 layers, and it suffers over-smoothing when the number of layers is larger than 3. Therefore, in this work, the depth of both E^α and E^β is set to 2, and the output vectors of E^α and E^β can be formulated as:

$$\begin{aligned} \boldsymbol{Z}^\alpha &= E^\alpha(\boldsymbol{X}^\alpha, \boldsymbol{A}^\alpha; \boldsymbol{W}^\alpha) \\ &= \tilde{\boldsymbol{A}}^\alpha \mathrm{ReLu}(\tilde{\boldsymbol{A}}^\alpha \boldsymbol{X}^\alpha \boldsymbol{W}_1^\alpha) \boldsymbol{W}_2^\alpha, \end{aligned} \tag{2}$$

$$\begin{aligned} \boldsymbol{Z}^\beta &= E^\beta(\boldsymbol{X}^\beta, \boldsymbol{A}^\beta; \boldsymbol{W}^\beta) \\ &= \tilde{\boldsymbol{A}}^\beta \mathrm{ReLu}(\tilde{\boldsymbol{A}}^\beta \boldsymbol{X}^\beta \boldsymbol{W}_1^\beta) \boldsymbol{W}_2^\beta, \end{aligned} \tag{3}$$

where $\boldsymbol{X}^\alpha$ and $\boldsymbol{X}^\beta$ are the input features of E^α and E^β, which are obtained by the Deepwalk embedding method. $\tilde{\boldsymbol{A}} = \tilde{\boldsymbol{D}}^{-\frac{1}{2}} \boldsymbol{A}' \tilde{\boldsymbol{D}}^{-\frac{1}{2}}$ is the symmetrically normalized adjacency matrix, in which $\boldsymbol{A}' = \boldsymbol{A} + \boldsymbol{I}_N$ is the adjacency matrix of the undirected graph $\mathcal{G}$ with added self-connections, $\boldsymbol{I}_N$ is the identity matrix, and the diagonal elements of matrix $\tilde{\boldsymbol{D}}$ are $\boldsymbol{D}_{ii} = \sum_j \boldsymbol{A}'_{ij}$. $\boldsymbol{W}^\alpha$ and $\boldsymbol{W}^\beta$ are the parameter matrices. The activation function $\mathrm{ReLu}(\cdot) = max(0, \cdot)$. $\boldsymbol{Z}^\alpha$ and $\boldsymbol{Z}^\beta$ are the final output embeddings of two networks, each row of which denotes the embedding vector of the corresponding node. Weight-sharing can be used to bridge E^α and E^β, where $\boldsymbol{W}_2^\alpha = \boldsymbol{W}_2^\beta$.

2) Decoder Module: Inspired by [47], a simple inner-product decoder is employed to reconstruct the original network $\mathcal{G}$ by calculating the pairwise similarity of their embedding vectors:

$$\hat{\boldsymbol{A}} = \sigma(\boldsymbol{Z} \cdot \boldsymbol{Z}^T). \tag{4}$$

Here, $\sigma(\cdot)$ is the Sigmoid function, and its value is between 0 and 1. $\hat{\boldsymbol{A}}_{i,j}$ denotes the similarity between v_i and v_j, indicating the probability of connection between them. By minimizing the difference between the reconstructed matrix $\hat{\boldsymbol{A}}$ and the adjacency matrix $\boldsymbol{A}$, intra-network structural features are preserved.

3) Objective Function Module: The objective function includes intra- and inter-network loss, in which intra-network loss mainly retains intra-network structural features, and inter-network loss retains inter-network structural features. Then the total loss of CCNE is collaboratively optimized to better unify the two networks to the same latent space.

Intra-network loss: The reconstruction loss of the above auto-encoder, which preserves the second-order proximity [48] can

be simply represented as:

$$L_{recon} = ||\boldsymbol{A} - \hat{\boldsymbol{A}}||_2. \quad (5)$$

Nevertheless, such a reconstruction loss may be not suitable for our problem owing to the sparsity of networks, in which the number of non-zero elements is far less than that of zero elements. To avoid elements in $\hat{\boldsymbol{A}}$ being reconstructed to be zero, and reduce the computational cost, we adopt the random negative sampling strategy. For each edge (v_i, v_j) in $\mathcal{E}$, we randomly sample a node pair (v_i, v_k), in which v_k has no connection with v_i. The set of these sampled node pairs is called $\mathcal{E}_n$, and the reconstruction loss can be rewritten as:

$$L_{recon} = -\frac{1}{|\mathcal{E}|}\sum_{(v_i,v_j)\in\mathcal{E}} \log(\sigma(\boldsymbol{z}_i^T \cdot \boldsymbol{z}_j)) - \frac{1}{|\mathcal{E}_n|}\sum_{(v_i,v_k)\in\mathcal{E}_n} \log(1-\sigma(\boldsymbol{z}_i^T \cdot \boldsymbol{z}_k)). \quad (6)$$

The intra-network loss is the sum of the reconstruction loss of $\mathcal{G}^\alpha$ and $\mathcal{G}^\beta$:

$$L_{intra} = L_{recon}^\alpha + L_{recon}^\beta. \quad (7)$$

By minimizing the intra-network loss, intra-network structural features of two networks can be preserved.

Inter-network loss: Besides intra-network structural features, inter-network structural features are crucial to the NA problem. We consider anchor nodes to be close in the latent space, i.e., they share similar embeddings. Therefore, we can build an anchor-aware loss to minimize the distance of anchor nodes.

Obviously, the number of non-anchor links is far more than anchor links. To overcome the sample imbalance problem, we adopt undersampling, which selects the several nearest non-anchor links for each anchor link. These non-anchor links termed hard negatives are more informative and helpful for NA. In detail, for each anchor link (v_i^α, v_j^β), we randomly sample k non-anchor links (v_i^α, v_k^β) for v_i^α, where $v_k^\beta \in \mathcal{N}(v_j^\beta)$, indicating that v_k^β is chosen from the neighbors of v_j^β. The same selection is performed for v_j^β. The set of anchor links is called $\mathcal{L}^+$ with the label 1, while the set of these sampled non-anchor links is called $\mathcal{L}^-$ (i.e., $|\mathcal{L}^-| = 2k|\mathcal{L}^+|$) with the label -1. The inter-network loss is defined as follows:

$$L_{inter} = \frac{1}{|\mathcal{L}^+|}\sum_{(v_i^\alpha,v_j^\beta)\in\mathcal{L}^+} (1-\cos(\boldsymbol{z}_i^\alpha, \boldsymbol{z}_j^\beta)) + \frac{1}{|\mathcal{L}^-|}\sum_{(v_k^\alpha,v_l^\beta)\in\mathcal{L}^-} max(\cos(\boldsymbol{z}_k^\alpha, \boldsymbol{z}_l^\beta) - \epsilon, 0), \quad (8)$$

where ϵ is a margin parameter: if $\cos(\boldsymbol{z}_k^\alpha, \boldsymbol{z}_l^\beta)$ is greater than ϵ, non-anchor link $(\boldsymbol{z}_k^\alpha, \boldsymbol{z}_l^\beta)$ is hard to be distinguished from anchor link $(\boldsymbol{z}_i^\alpha, \boldsymbol{z}_j^\beta)$, therefore $\cos(\boldsymbol{z}_k^\alpha, \boldsymbol{z}_l^\beta)$ is encouraged to decrease; otherwise, it is not taken into account. By minimizing inter-network loss, we can preserve inter-network structural information.

Total loss of CCNE: The total objective function of CCNE is defined as a linear combination of both intra- and inter-network loss, so that we can preserve intra- and inter-network structural information by collaboratively optimizing this:

$$L = \alpha \cdot L_{intra} + (1-\alpha) \cdot L_{inter}. \quad (9)$$

Here, α is the weight parameter to tradeoff the two parts of the objective function.

For the existing embedding-matching methods, the embedding stage and the matching stage are two relatively isolated processes. In the embedding stage, intra-network structural features are captured by various embedding techniques, while the two networks are mapped to the common latent space with known anchor links in the matching stage. Evidently, the embedding stage can affect the matching stage. However, the latter stage cannot affect the former stage, which may result in anchor nodes and their neighbors being too close to distinguish them. In our proposed framework, collaborative training and hard negative sampling are used to embed anchor nodes as close as possible while retaining intra-network structural features, so as to alleviate the confounding matching problem.

The pseudocode for CCNE is shown as Algorithm 1.

Algorithm 1: CCNE.

Input: Networks $\mathcal{G}^\alpha$, $\mathcal{G}^\beta$, anchor links $\mathcal{L}^+$, learning rate η, margin ϵ, and weight parameter α
Output: Embedding vectors $\boldsymbol{Z}^\alpha$ for $\mathcal{G}^\alpha$, $\boldsymbol{Z}^\beta$ for $\mathcal{G}^\beta$

```
Obtain structure feature vectors X^α for G^α, X^β for G^β based on DeepWalk
Initialize parameter set W = {W^α, W^β, W^s}
while not converged do
    Sample non-anchor links L^- from neighbors of anchor nodes
    Generate embedding vectors Z^α, Z^β by (2) and (3)
    Calculate the total loss by (9)
    Update W with Adam optimizer
end while
return Z^α, Z^β
```

B. Network Alignment

The embeddings obtained by the proposed framework are in the same latent space, so that we can compute their similarity directly without the mapping function. We use cosine similarity to build the alignment matrix:

$$sim_{align}(v_i^\alpha, v_j^\beta) = \frac{(\boldsymbol{z}_i^\alpha)^T \cdot \boldsymbol{z}_j^\beta}{\|\boldsymbol{z}_i^\alpha\|_2 \cdot \|\boldsymbol{z}_j^\beta\|_2}. \quad (10)$$

For any unmatched node v_i^α in $\mathcal{G}^\alpha$, we compute its similarity with each unmatched node v_j^β in $\mathcal{G}^\beta$, from which we select top-k most likely nodes as candidate matching list for v_i^α.

C. Iterative CCNE

The NA problem often suffers from the scarcity of observed anchor links. An intuitive way of solving this problem is that a small portion of node pairs from unmatched anchor links $\mathcal{U}$ is selected for labeling. In each iteration, bs node pairs from $\mathcal{U}$ are

Algorithm 2: ICCNE.

Input: Networks $\mathcal{G}^\alpha$, $\mathcal{G}^\beta$, anchor links $\mathcal{L}^+$, learning rate η, margin ϵ, weight parameter α, batch size bs, and maximum size M
Output: Embedding vectors $\boldsymbol{Z}^\alpha$ for $\mathcal{G}^\alpha$, $\boldsymbol{Z}^\beta$ for $\mathcal{G}^\beta$
1: **for** $i = 1 : M/bs$ **do**
2: Generate embedding vectors $\boldsymbol{Z}^\alpha$, $\boldsymbol{Z}^\beta$ based on Algorithm 1
3: Select bs candidate anchor links based on (11)
4: Update anchor links $\mathcal{L}^+$ by adding the selected bs candidate anchor links
5: **end for**
6: **return** $\boldsymbol{Z}^\alpha$, $\boldsymbol{Z}^\beta$

added to the known anchor links $\mathcal{L}^+$ based on query strategy ϕ. Then embeddings are retrained until the number of extra anchor links reaches the maximum size M. We adopt the cosine similarity-based query strategy to select candidate anchor links from $\mathcal{U}$:

$$\phi(v_i^\alpha, v_j^\beta) = \cos(v_i^\alpha, v_j^\beta). \tag{11}$$

The larger value of $\phi(v_i^\alpha, v_j^\beta)$ is, the more possibility of (v_i^α, v_j^β) being an anchor link. We add this strategy to our CCNE framework, namely ICCNE (Iterative CCNE). We expect that the performance of NA should be better with the recursion of CCNE and anchor links expansion. The pseudocode for ICCNE is shown as Algorithm 2.

D. Time Complexity Analysis

The depth of the GCN encoder is set to 2, where the dimension of the input features is c, the dimension of the hidden layer is h, and the dimension of the output embeddings is d. $\mathcal{E} = max\{|\mathcal{E}^\alpha|, |\mathcal{E}^\beta|\}$. Accordingly, we know that the time complexity of graph convolution is $O(\mathcal{E}h(c+d))$, the time complexity of intra-network loss is $O(\mathcal{E})$, and the time complexity of inter-network loss is $O(|\mathcal{L}^+|)$. In most OSNs, $|\mathcal{L}^+| << \mathcal{E}$, so the time complexity of CCNE is $O(\mathcal{E}h(c+d))$. That is to say, the time complexity of CCNE is mainly determined by the scale of edges $\mathcal{E}$ when hyperparameters of the model are fixed.

V. Experiments

In this section, we compare CCNE with existing baseline methods on real-world social networks to address the effectiveness of CCNE. We also analyze the impacts of different hyperparameters and settings.

A. Datasets

We employ three real-world cross-network datasets, and their structural statistics are listed in Table I, where interoperability [35] can reflect the similarity between different networks to some extent. Interoperability can be calculated as (12).

$$Interoperability(\mathcal{G}^\alpha, \mathcal{G}^\beta) = \frac{2 \times |\mathcal{E}^\alpha \cap \mathcal{E}^\beta|}{|\mathcal{E}^\alpha| + |\mathcal{E}^\beta|}. \tag{12}$$

TABLE I
Structural Statistics of the Datasets

Datasets	Nodes#	Edges#	Anchors#	Interoperability#
Douban-onine	3906	8164	1118	0.3124
Douban-offline	1118	1511		
Twitter	4543	15941	3057	0.1242
YouTube	4914	21068		
Twitter	5120	130575	1609	0.0439
Foursquare	5313	54233		

Note that we only adopt the information of network structure and anchor links for the following datasets, namely, the attribute information of users is not included.

Douban-online/offline: This Douban dataset was collected in 2010 and consists of 50 k users and 5 M edges [49]. Each user has information including offline event participation. The offline network contains 1118 users according to users' co-occurrence in social gatherings, and a subnetwork with 3,906 nodes from the original network is extracted as the online network, which contains all these offline users [39].

Twitter/YouTube: This anonymized dataset has been obtained starting from Friendfeed, a social media aggregator [50]. In this system, users can directly post messages and comments on other messages much like in Twitter and other similar OSNs, where they can also register their accounts on other systems. From these, a multilayer network was retrieved, including the Twitter layer and YouTube layer. We select subnetworks of these two layers, with 3057 anchor links.

Twitter/Foursquare: This dataset was collected from Twitter and Foursquare and published in [24]. The anchor link between Foursquare and Twitter is obtained by crawling users' Twitter accounts from their Foursquare homepages. We employ part of this dataset, which contains 1609 anchor links [28].

B. Experiment Settings

1) Comparison Methods: Since the CCNE framework only uses the structural information of intra- and inter-networks, the baseline methods should have the same scenarios. Moreover, our CCNE framework mainly focuses on overcoming the shortcomings of the two-stage embedding-based methods. Therefore, in this paper, the following baseline methods are considered:

- *DNN [51]:* This method simply trains an MLP classifier to determine whether the input node pair vector is an anchor link. For each node pair across networks, the node embeddings are concatenated into the node pair vector. The labels of anchor links are set to 1, and 0 for non-anchor links.
- *DeepWalk [45]:* This method constructs an expanded graph containing all nodes and edges from two networks, and all training anchor links between two networks. Then DeepWalk is applied to the expanded graph to learn node embeddings of two networks.
- *CLF [24]:* This method is a collective link fusion model predicting both intralayer and interlayer links. It can be seen as random walks with restart [52] across two networks.
- *PALE [16]:* This method contains embedding stage and matching stage. It first embeds the networks into two

TABLE II
PERFORMANCE COMPARISON OF DIFFERENT METHODS WITH 20% TRAINING DATA

	Douban-online/offline			Twitter/YouTube			Twitter/Foursquare		
	Precision@1	Precision@10	MRR	Precision@1	Precision@10	MRR	Precision@1	Precision@10	MRR
DNN	0.0648	0.3374	0.1494	0.0045	0.0335	0.0167	0.0054	0.0505	0.0244
DeepWalk	0.0301	0.5684	0.1914	0.0018	0.0558	0.0222	0.0009	0.0535	0.0198
CLF	0.0000	0.3777	0.1167	0.0000	0.0871	0.0297	0.0000	0.0854	0.0284
PALE	0.1751	0.6264	0.3209	0.0150	0.1190	0.0505	0.0287	0.1823	0.0783
IONE	0.2056	0.4838	0.3025	0.0200	0.0458	0.0304	0.0163	0.1119	0.0547
DeepLink	0.1532	0.5555	0.2826	0.0170	0.1137	0.0504	0.0312	0.1732	0.0767
dNAME	0.1644	0.6027	0.3000	0.0179	0.1313	0.0566	0.0370	0.2043	0.0921
CAMU	0.2000	0.6381	0.3382	0.0154	0.0895	0.0420	0.0311	0.1725	0.0789
NeXtAlign	**0.2857**	0.6368	**0.4037**	0.0304	0.1025	0.0558	0.0233	0.0970	0.0470
CENALP	0.1285	0.4123	0.2134	0.0339	0.0720	0.0469	0.0629	0.1297	0.0870
Grad-Align+	0.1877	0.4335	0.2614	**0.0380**	0.0715	0.0509	**0.0730**	0.1064	0.0851
CCNE	0.2230	**0.6598**	0.3648	0.0245	**0.1471**	**0.0664**	0.0540	**0.2554**	**0.1210**

The bold values are the best results.

low-dimensional spaces independently by preserving intra-structure features and then learns the mapping function based on observed anchor links.

- *IONE [28]:* This method represents the follower-ship and followee-ship of each node as input and output context vectors, and then develops a network embedding model to preserve the proximity of nodes with similar contexts.
- *DeepLink [8]:* Compared to PALE, DeepLink mainly improves the matching process. It exploits a dual learning-based paradigm, consisting of an unsupervised pretraining step and a supervised step based on reinforcement learning.
- *dNAME [30]:* This method presents an adversarial learning paradigm by leveraging the idea of GAN [31]. This adversarial matching process is to learn the distribution of anchor users, which can be considered as a minimax game between the mapping function and discriminator.
- *CAMU [32]:* This method proposes a cycle-consistent adversarial mapping model to learn two mapping functions across different embedding spaces. The adversarial training based on energy-based GAN aims to reduce the distribution discrepancy between the mapped users and the corresponding anchors. Then the cycle-consistency training prevents the conflict of the two learned mapping functions.
- *NeXtAlign [33]:* This method uses RelGCN to extract features, and designs different sampling distributions to achieve the trade-off between the alignment consistency and disparity.
- *CENALP [34], [35]:* This method is a Skip-gram embedding model based on biased random walks across two networks, which alternately performs network alignment and link prediction.
- *GradAlign+ [43]:* This method feeds node attributes and augmented node attributes into two GCNs respectively, and gradually discovers node pairs. Note that node attributes are not available in our experiment, so we use 1-hop centrality as node attributes and Katz centrality as augmented node attributes.

Note that CENALP and GradAlign+ are both iterative methods, so they will be compared not only with CCNE when the training ratio is 0.2 (see Table II), but also with ICCNE under different training ratios in the experiments (see Fig. 4).

2) Settings: In order to equally compare the performance of different methods, all the above baselines employ the same embedding model, except for the inter-network loss and the hard negative sampling strategy. The input features of a 2-layer GCN encoder are obtained by using DeepWalk method on respective networks. For the DeepWalk, the dimension is set to 128, the window size is 10, the walk length is 80, the number of walks per node is 10, and the iteration number is 5. For the GCN encoder, the input dimension c is 128, the hidden dimension h is 256, and the output dimension d is 128.

CCNE employs the same parameter settings as the above embedding model. Further, the margin ϵ of inter-network loss is set to 0.8, the weight parameter α is set to 0.5, and the learning rate η is 0.001. We adopt Adam optimizer for better convergence.

All experiments are run on a CentOS Linux 7.5 server with an NVIDIA A100-40 G GPU and 16 GB of allocation memory. All codes are implemented in Python with PyTorch and PyTorch Geometric library. All experiments in this paper are carried out 10 times and averaged.

3) Evaluation Metrics: To evaluate the performance of NA methods, Precision@K and Mean Reciprocal Rank (MRR) are employed as the evaluation metrics [17]. Precision@K is to measure the accuracy of matching, which is defined as:

$$Precision@K = \frac{1}{n}\sum_{i=1}^{n}\mathbb{I}(r_i <= K), \tag{13}$$

where r_i is the rank of the corresponding anchor node in the candidate list, $\mathbb{I}(r_i <= K)$ indicates whether the corresponding anchor node of user i occurs in the $top-K(K <= n)$ candidate list, and n is the number of testing anchor nodes. K is often set to 10 for stable results. Besides, MRR is to measure the average precision performance over all testing anchor nodes, which has good stability:

$$MRR = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{r_i}. \tag{14}$$

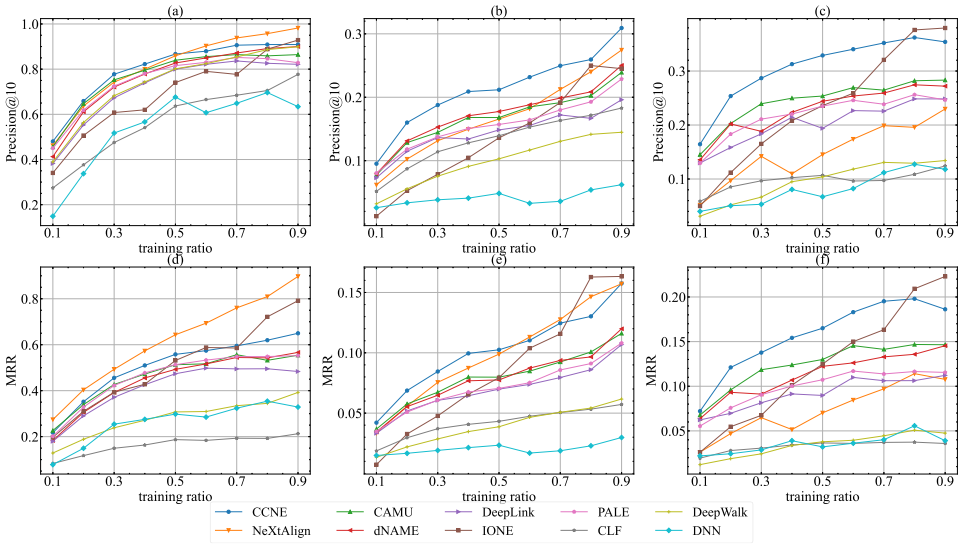


Fig. 3. Effectiveness of CCNE under different training ratios. (a) and (d) Precision@10/MRR on Douban-online/offline dataset; (b) and (e) Precision@10/MRR on Twitter/YouTube dataset; (c) and (f) Precision@10/MRR on Twitter/Foursquare dataset.

Obviously, the higher values of these metrics indicate the better performance of NA.

C. Experimental Results

1) Effectiveness Analysis: In this section, we compare the performance of CCNE with other baselines on three real-world datasets. The results with training ratio $\theta = 0.2$ are summarized in Table II, where the best results are shown in bold, and the sub-optimal results are underlined. In addition, The results under different training ratios are demonstrated in Fig. 3 (the comparison with iterative methods is not included in Fig. 3, but demonstrated in Fig. 4 separately). We can draw the following conclusions:

- For different training ratios θ and on all datasets, the DNN method often has the worst performance among these methods, because it works on finding the difference between anchor links and non-anchor links, but ignores the similarity between anchor users;
- In some ways, DeepWalk and CLF can be considered as probabilistic methods that estimate the transfer probability of a node walking across two networks. They perform poorly with a small training ratio, as the probability of a node walking from one network to the other is very small;
- The dNAME and CAMU usually perform better than PALE and DeepLink, demonstrating that adversarial training could learn better mapping functions to unify the latent spaces of two networks;
- IONE performs poorly when the training ratio is small, but sometimes achieves optimal results when the training ratio is large;
- CENALP and GradAlign+ are iterative methods. From Table II, one can observe that they have great Precision@1 on Twitter/YouTube and Twitter/Foursquare datasets when the training ratio is 0.2. However, their Precision@10 and MRR do not perform as well as CCNE (non-iterative method). It might be because error accumulates when added training anchors increase during the iterative process, especially when the initial training ratio is small.
- NeXtAlign performs best on the Douban-online/offline dataset, while it does not perform well on the other two datasets. It might be because NeXtAlign is more suitable for networks with higher similarity (i.e., interoperability).
- CCNE significantly outperforms most of the baselines. It validates the effectiveness of our proposed CCNE framework. In particular, CCNE works well with a small portion of anchor links, indicating its high capability of preserving inter-network features.

In general, the accuracy of NA increases when the training ratio becomes larger. However, in real situations, anchor links are usually difficult to obtain. In Section IV-C, we have proposed the ICCNE method to alleviate the scarcity of anchor links. To analyze the effectiveness of ICCNE, we compare it with two iterative methods (i.e., CENALP and GradAlign+) on three real-world datasets under different training ratios θ, and we add $bs = 10$ most possible anchor links in each iteration. The experimental results are shown in Fig. 4. One can observe that ICCNE generally performs better than CCNE, indicating the effectiveness of the iterative process. ICCNE generally outperforms CENALP and GradAlign+, especially when θ is small, i.e., the number of anchor links is scarce. For instance, ICCNE improves MRR by 36.2%, and Precision@10 by 14.5% respectively over CCNE when $\theta = 0.2$ on Douban-online/offline dataset (see Fig. 4(a) and (d)). That is to say, the scarcity of observed anchor links for NA can be partially solved by the ICCNE method.

2) Ablation Study: To investigate the importance of the different components in our proposed model, we conduct an ablation study on the Twitter/YouTube dataset when training ratio $\theta = 0.9$. We compare our model CCNE with 5 ablated variants: (i) CCNE without negative sampling in intra-network loss, where $\epsilon = 1$ (w/o negative sampling); (ii) sample negatives from all nodes randomly rather than neighbors of anchor nodes (with

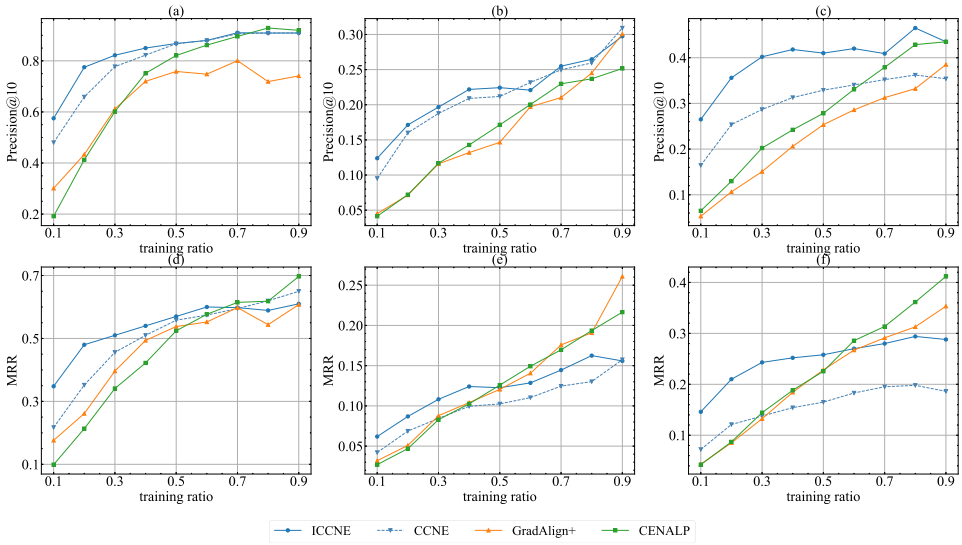


Fig. 4. Effectiveness of ICCNE under different training ratios. (a) and (d) Precision@10/MRR on Douban-online/offline dataset; (b) and (e) Precision@10/MRR on Twitter/YouTube dataset; (c) and (f) Precision@10/MRR on Twitter/Foursquare dataset.

TABLE III
ABLATION STUDY ON TWITTER/YOUTUBE DATASET

Method	Precision@10	MRR
CCNE	**0.3092**	**0.1575**
W/o negative sampling	0.2801	0.1529
With random negative sampling	0.2961	0.1551
W/o weight-sharing	0.2912	0.1448
W/o intra-network loss	0.0134	0.0072
W/o inter-network loss	0.0069	0.0038

The bold values are the best results.

TABLE IV
ABLATION STUDY WITH/WITHOUT MAPPING PROCESS

Dataset	Metric	**CCNE**	With mapping
Douban-online/offline	MRR	**0.3648**	0.3444
	Precision@1	**0.2230**	0.2053
	Precision@10	**0.6598**	0.6371
Twitter/YouTube	MRR	**0.0664**	0.0633
	Precision@1	**0.0245**	0.0211
	Precision@10	**0.1471**	0.1428
Twitter/Foursquare	MRR	**0.1210**	0.1042
	Precision@1	**0.0540**	0.0420
	Precision@10	**0.2554**	0.2352

The bold values are the best results.

random negative sampling); (iii) CCNE without weight-sharing in the 2-nd layer (w/o weight-sharing); (iv) CCNE without the intra-network loss, where $\alpha = 0$ (w/o intra-network loss); (v) CCNE without the inter-network loss, where $\alpha = 1$ (w/o inter-network loss). The results are shown in Table III, where black means the optimal results. We can observe that CCNE outperforms most variants, indicating the importance of these components.

Would it be better to add a matching stage after CCNE? To address this possible concern, we compare CCNE with its variant with a mapping process. In detail, we first obtain node embeddings Z^α and Z^β for each network via CCNE and then use an MLP $\phi : \mathcal{G}^\alpha \to \mathcal{G}^\beta$ to project Z^α to $\phi(Z^\alpha)$, which is in the same space as Z^β. Experimental results on all datasets when training ratio $\theta = 0.2$ are shown in Table IV. One can find that CCNE itself performs better than its variant with a mapping process. It indicates that CCNE has projected different networks into the same space via jointly preserving intra- and inter-network consistency, and it does not need an extra mapping function after CCNE.

3) Parameter Sensitivity Analysis: To analyze the parameter sensitivity of the CCNE method, the impacts of hyperparameters, including the dimension d, the margin of the inter-network loss ϵ, and the weight parameter α on the performance of CCNE method are investigated. We conduct the experiments of CCNE on Twitter/YouTube dataset while fixing the training ratio $\theta = 0.9$. Fig. 5(a) illustrates the effect of the dimension d by varying its values as $\{16, 32, 64, 128, 256, 400, 512\}$. The alignment performance presents an apparent enhancement with the increase of the embedding dimension d and tends to remain stable when d exceeds 128. This may benefit from more information carried by larger embedding dimensions. Fig. 5(b) presents the effect of the margin ϵ by varying its values from 0 to 1, and $\epsilon = 1$ means negative sampling is not employed. It can be seen that Precision@10 reaches its maximum when $\epsilon = 0.7$. This finding indicates that hard negative sampling is effective, and proximity between the anchor nodes and their neighbors is partially preserved whereas we aim to separate the anchor nodes from their neighbors. Fig. 5(c) presents the effect of the weight parameter α by varying its values from 0 to 1. Note that $\alpha = 0$ means only inter-network loss is taken into account, while $\alpha = 1$ implies only intra-network loss is employed. One can find that Precision@10 is extremely low for both $\alpha = 0$ and $\alpha = 1$, indicating both intra- and inter-network losses are crucial. It achieves the best performance when $\alpha = 0.4$, as both intra- and inter-network structural features are considered.

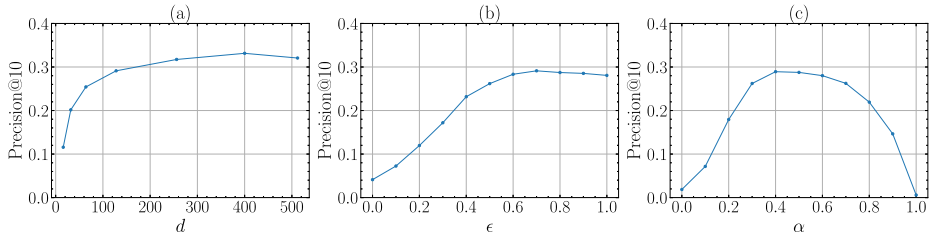


Fig. 5. Parameter sensitivity analysis of CCNE. All experiments are conducted on Twitter/YouTube dataset with training ratio $\theta = 0.9$. (a) Precision@10 versus the embedding dimensionality d. (b) Precision@10 versus the margin of the inter-network loss ϵ. (c) Precision@10 versus the weight parameter α.

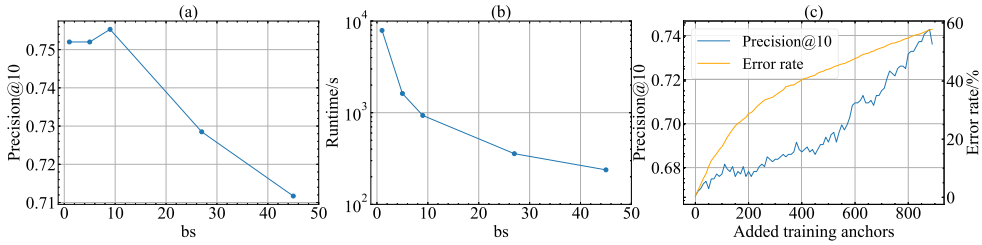


Fig. 6. Parameter sensitivity analysis of ICCNE. All experiments are conducted on Douban-online/offline dataset with training ratio $\theta = 0.2$. (a) Precision@10 versus bs. (b) Runtime versus bs. (c) Precision@10 & error rate versus total number of added training anchors.

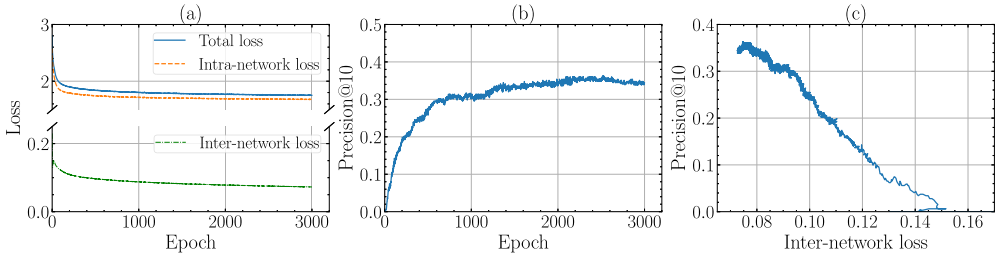


Fig. 7. Convergence analysis. All experiments are conducted on Twitter/Foursquare dataset with training ratio $\theta = 0.8$. (a) Different losses versus the number of epochs. (b)Precision@10 versus the number of epochs. (c)Precision@10 versus the inter-network loss.

To analyze the parameter sensitivity of the ICCNE method, we investigate the impact of the number of added training anchors per iteration bs on Douban-online/offline dataset with training ratio $\theta = 0.2$. Fig. 6(a) and (b) present Precision@10 and runtime of ICCNE when $bs = 1, 5, 9, 27, 45$ (i.e., 0.1%, 0.5%, 1%, 3%, 5% of test anchors). It seems that ICCNE achieves optimal performance and relatively low runtime when $bs = 9$. We also investigate the impact of the total number of added training anchors. Results shown in Fig. 6(c) demonstrate that Precision@10 increases with the increasing number of added training anchors. The error rate of all training anchors accumulates when added training anchors increase, which might have side effects on network alignment.

4) Convergence Analysis: In Fig. 7, we conduct the experiment on Twitter/Foursquare dataset to analyze the convergence of CCNE. As demonstrated in Fig. 7(a), the intra-network loss, inter-network loss, and the total loss all decrease with the number of epochs. The intra-network loss converges around epoch=100, while inter-network loss and total loss still decrease with the number of epochs. That is to say, the total loss is mainly determined by the inter-network loss. Fig. 7(b) expresses that the Precision@10 increases with the number of epochs, while its value does not converge until epoch>1000, indicating that the training process is mainly dominated by the inter-network loss at this time. The result in Fig. 7(c) also suggests that Precision@10 has an inverse relationship with the inter-network loss in general, implying the positive role of the inter-network loss in improving the performance of CCNE method.

5) Robustness Analysis: To analyze the robustness of CCNE, we compare the performance of CCNE with the baselines on a set of synthetic datasets. We adopt the following strategy to sample two sub-networks, where nodes are all inherited from the original network [16]. First, p is randomly generated with the uniform distribution in $[0, 1]$ for each edge. Then we build two sub-networks with p: (1) If $p \leq 1 - 2\alpha_s + \alpha_s\alpha_o$, the edge is dropped in both sub-netorks; (2) If $1 - 2\alpha_s + \alpha_s\alpha_o < p \leq 1 - \alpha_s$, the edge is preserved only in the first sub-network; (3) If $1 - \alpha_s < p \leq 1 - \alpha_s\alpha_o$, the edge is preserved only in the other sub-network; (4) Otherwise, the edge is preserved in both sub-networks. Generally, the sample ratio α_s can reflect the sparsity level of networks, and the expected fraction α_o can reflect the overlapping level (i.e., the number of anchor links).

We use Twitter dataset [28] as the original network to study the robustness of models when facing different sparsity levels.

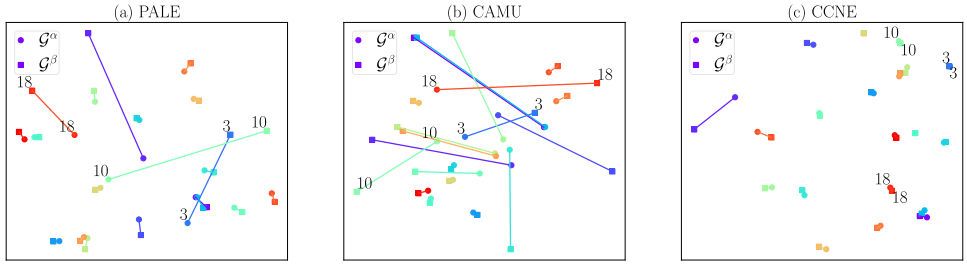


Fig. 8. t-SNE performance of different NA methods. Nodes from $\mathcal{G}^\alpha$ and $\mathcal{G}^\beta$ are represented by circles and squares, respectively. Nodes with the same color are the corresponding anchor nodes.

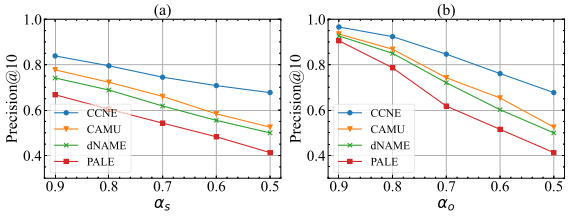


Fig. 9. Robustness analysis. We take Twitter as the original network with training ratio $\theta = 0.2$. (a) Precision@10 versus the sparsity level α_s. (b)Precision@10 versus the overlapping level α_o.

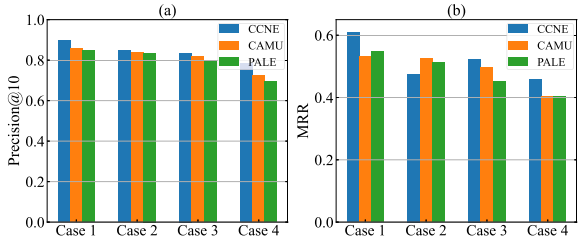


Fig. 10. Effect of different embedding methods on the performance of NA in terms of (a) Precision@10 and (b) MRR.

To do so, we fix the overlapping level α_o to 0.5, and change α_s as $\{0.5, 0.6, 0.7, 0.8, 0.9\}$ (see Fig. 9(a)). Similarly, the sparsity level α_s is fixed to 0.5 while α_o is varied from 0.5 to 0.9, which demonstrates the robustness when facing different overlapping levels (see Fig. 9(b)). We can find that: (1) Both the sparsity and overlapping levels of networks do influence the performance of models. All models perform better with larger value of α_s or α_o; (2) Compared with the sparsity level, the overlapping level seems to play a more significant role in NA performance; (3) As shown in Fig. 9(a), when the sparsity level drops from 0.9 to 0.5, the performance of CAMU, dNAME, and PALE declines about 32%, 33%, and 38%, while CCNE only decreases less than 20%. Also, as shown in Fig. 9(b), the CCNE is still more robust when facing smaller overlapping level.

D. Case Study

To visualize the latent space of the two networks, the dimension of the embeddings on Douban-online/offline dataset obtained by different methods is reduced to 2 by using popular t-SNE [53], and 20 pairs of anchor nodes are randomly selected from the test set. The results are shown in Fig. 8. Nodes from $\mathcal{G}^\alpha$ and $\mathcal{G}^\beta$ are represented by circles and squares, respectively. And nodes with the same color are the corresponding anchor nodes. It can be observed that, for PALE (see Fig. 8(a)) and CAMU (see Fig. 8(b)), anchor nodes 3, 10, and 18 are too far away, while they are close in CCNE (see Fig. 8(c)). This indicates that CCNE better preserves the structural features among the networks. Overall, nodes of the same color (corresponding anchor nodes) are closer in CCNE than that in PALE and CAMU, and nodes of different colors are more separated. The observations indicate that CCNE retains more similarity of anchor nodes across the networks and can better distinguish anchor nodes from their neighbors.

E. Compatibility of CCNE Framework

Network embedding plays an important role in CCNE and other baselines. There are many well-known network embedding methods, such as, GAT [54] and GraphSAGE [55]. To investigate the effect of different embedding methods for NA, we design some cases as follows:

- **Case 1**: The original settings are shown in Section V-B2.
- **Case 2**: The GCN encoder in Case 1 is replaced by GAT.
- **Case 3**: The GCN encoder in Case 1 is replaced by GraphSAGE.
- **Case 4**: DeepWalk that extracts the input features of GCN encoder is replaced by LINE [56].

These experiments are all conducted on Douban-online/ offline dataset with a training ratio of $\theta = 0.8$. The results are shown in Fig. 10. One can find that CCNE outperforms PALE and CAMU in most cases except for MRR in Case 2. This proves the compatibility of our CCNE framework. Besides, Case 1 slightly outperforms Case 2 and Case 3, suggesting that GCN is better than GAT and GraphSAGE under this framework. Case 4 is weaker than Case 1, which might result from the high capability of LINE in preserving the intra-network proximity, weakening the requirement in preserving the inter-network proximity.

VI. Conclusion and Future Work

In this paper, we built a collaborative paradigm, i.e., collaborative cross-network embedding (CCNE) framework, to learn node representations to the same latent space by preserving both intra- and inter-network features. It first leverages two independent encoders to capture intra-network structural features, and then learns inter-network structural features based on the similarity between anchor nodes. Further, the neighbors of anchor nodes are sampled as hard negatives to alleviate the

confounding matching problem. Compared with the existing embedding-matching methods, the framework does not require the mapping function since intra- and inter-network structural features are both preserved during the collaborative optimization process. A series of experiments on real-world datasets show that CCNE outperforms many embedding-matching methods in terms of effectiveness, robustness and compatibility. To alleviate the scarcity of observed anchor links, we also proposed an extension framework ICCNE to increase anchor links in each iteration, which was experimentally proven to be effective.

Although CCNE is based on structural features, attribute features can help learn more discriminative representations, e.g., in scenarios where user profile information is available, attribute features can be concatenated with structural features. Many works have demonstrated the effectiveness of adversarial training in improving the performance of NA problem, so we can incorporate such a mechanism in our framework. Further, we also note that ICCNE suffers from significant time overhead, due to re-training in each iteration. We should find an online training method that could accelerate training. These issues mentioned above, but not limited to them, need to be considered in future work.

REFERENCES

[1] L. Qi, X. Wang, X. Xu, W. Dou, and S. Li, "Privacy-aware cross-platform service recommendation based on enhanced locality-sensitive hashing," *IEEE Trans. Netw. Sci. Eng.*, vol. 8, no. 2, pp. 1145–1153, Apr.–Jun. 2021.

[2] P. Basaras, G. Iosifidis, D. Katsaros, and L. Tassiulas, "Identifying influential spreaders in complex multilayer networks: A centrality perspective," *IEEE Trans. Netw. Sci. Eng.*, vol. 6, no. 1, pp. 31–45, Jan.–Mar. 2019.

[3] Y. Huang, A. Panahi, H. Krim, and L. Dai, "Community detection and improved detectability in multiplex networks," *IEEE Trans. Netw. Sci. Eng.*, vol. 7, no. 3, pp. 1697–1709, Jul.–Sep. 2020.

[4] Y. Huang and H. Dai, "Multiplex conductance and gossip based information spreading in multiplex networks," *IEEE Trans. Netw. Sci. Eng.*, vol. 6, no. 3, pp. 391–401, Jul.–Sep. 2019.

[5] Z. Zhang et al., "When behavior analysis meets social network alignment," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 7, pp. 7590–7607, Jul. 2023.

[6] H. Hong, X. Li, Y. Pan, and I. W. Tsang, "Domain-adversarial network alignment," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 7, pp. 3211–3224, Jul. 2022.

[7] A. Cheng et al., "Deep active learning for anchor user prediction," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, 2019, pp. 2151–2157.

[8] F. Zhou, L. Liu, K. Zhang, G. Trajcevski, J. Wu, and T. Zhong, "DeepLink: A deep learning approach for user identity linkage," in *Proc. IEEE Conf. Comput. Commun.*, 2018, pp. 1313–1321.

[9] T. Iofciu, P. Fankhauser, F. Abel, and K. Bischoff, "Identifying users across social tagging systems," in *Proc. 5th Int. AAAI Conf. Web Social Media*, 2011, pp. 522–525.

[10] J. Liu, F. Zhang, X. Song, Y.-I. Song, C.-Y. Lin, and H.-W. Hon, "What's in a name? An unsupervised approach to link users across communities," in *Proc. 6th ACM Int. Conf. Web Search Data Mining*, 2013, pp. 495–504.

[11] O. Goga, H. Lei, S. H. K. Parthasarathi, G. Friedland, R. Sommer, and R. Teixeira, "Exploiting innocuous activity for correlating users across sites," in *Proc. 22nd Int. Conf. World Wide Web*, 2013, pp. 447–458.

[12] Y. Nie, Y. Jia, S. Li, X. Zhu, A. Li, and B. Zhou, "Identifying users across social networks based on dynamic core interests," *Neurocomputing*, vol. 210, pp. 107–115, 2016.

[13] G. Kollias, S. Mohammadi, and A. Grama, "Network similarity decomposition (NSD): A fast and scalable approach to network alignment," *IEEE Trans. Knowl. Data Eng.*, vol. 24, no. 12, pp. 2232–2243, Dec. 2012.

[14] Y. Zhang, J. Tang, Z. Yang, J. Pei, and P. S. Yu, "COSNET: Connecting heterogeneous social networks with local and global consistency," in *Proc. 21th ACM SIGKDD Conf. Knowl. Discov. Data Mining*, 2015, pp. 1485–1494.

[15] H. Cai, V. W. Zheng, and K. C.-C. Chang, "A comprehensive survey of graph embedding: Problems, techniques, and applications," *IEEE Trans. Knowl. Data Eng.*, vol. 30, no. 9, pp. 1616–1637, Sep. 2018.

[16] T. Man, H. Shen, S. Liu, X. Jin, and X. Cheng, "Predict anchor links across social networks via an embedding approach," in *Proc. 25th Int. Joint Conf. Artif. Intell.*, 2016, pp. 1823–1829.

[17] H. T. Trung et al., "A comparative study on network alignment techniques," *Expert Syst. Appl.*, vol. 140, 2020, Art. no. 112883.

[18] R. Zafarani and H. Liu, "Connecting users across social media sites: A behavioral-modeling approach," in *Proc. 19th ACM SIGKDD Conf. Knowl. Discov. Data Mining*, 2013, pp. 41–49.

[19] X. Mu, F. Zhu, E.-P. Lim, J. Xiao, J. Wang, and Z.-H. Zhou, "User identity linkage by latent user space modelling," in *Proc. 22nd ACM SIGKDD Conf. Knowl. Discov. Data Mining*, 2016, pp. 1775–1784.

[20] X. Kong, J. Zhang, and P. S. Yu, "Inferring anchor links across multiple heterogeneous social networks," in *Proc. 22nd ACM Int. Conf. Inf. Knowl. Manage.*, 2013, pp. 179–188.

[21] N. Korula and S. Lattanzi, "An efficient reconciliation algorithm for social networks," in *Proc. Int. Conf. Very Large Data Bases Endowment*, 2014, pp. 377–388.

[22] S. Tan, Z. Guan, D. Cai, X. Qin, J. Bu, and C. Chen, "Mapping users across networks by manifold alignment on hypergraph," in *Proc. AAAI Conf. Artif. Intell.*, 2014, pp. 159–165.

[23] X. Zhou, X. Liang, H. Zhang, and Y. Ma, "Cross-platform identification of anonymous identical users in multiple social media networks," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 2, pp. 411–424, Feb. 2016.

[24] J. Zhang and P. S. Yu, "Integrated anchor and social link predictions across social networks," in *Proc. 24th Int. Joint Conf. Artif. Intell.*, 2015, pp. 2125–2132.

[25] X. Ding, C. Ma, X. Zhang, H.-S. Chen, and H.-F. Zhang, "SOIDP: Predicting interlayer links in multiplex networks," *IEEE Trans. Comput. Social Syst.*, vol. 9, no. 4, pp. 997–1007, Aug. 2022.

[26] S. Feizi, G. Quon, M. Recamonde-Mendoza, M. Médard, M. Kellis, and A. Jadbabaie, "Spectral alignment of graphs," *IEEE Trans. Netw. Sci. Eng.*, vol. 7, no. 3, pp. 1182–1197, Jul.–Sep. 2020.

[27] X. Ding, H. Zhang, C. Ma, X. Zhang, and K. Zhong, "User identification across multiple social networks based on naive Bayes model," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Sep. 14, 2022, doi: 10.1109/TNNLS.2022.3202709.

[28] L. Liu, W. K. Cheung, X. Li, and L. Liao, "Aligning users across social networks using network embedding," in *Proc. 25th Int. Joint Conf. Artif. Intell.*, 2016, pp. 1774–1780.

[29] Y. Wang, H. Shen, J. Gao, and X. Cheng, "Learning binary hash codes for fast anchor link retrieval across networks," in *Proc. World Wide Web Conf.*, 2019, pp. 3335–3341.

[30] F. Zhou, Z. Wen, G. Trajcevski, K. Zhang, T. Zhong, and F. Liu, "Disentangled network alignment with matching explainability," in *Proc. IEEE Conf. Comput. Commun.*, 2019, pp. 1360–1368.

[31] I. Goodfellow et al., "Generative adversarial nets," in *Proc. Int. Conf. Adv. Neural Inf. Process. Syst.*, 2014, pp. 2672–2680.

[32] C. Zheng, L. Pan, and P. Wu, "CAMU: Cycle-consistent adversarial mapping model for user alignment across social networks," *IEEE Trans. Cybern.*, vol. 52, no. 10, pp. 10709–10720, Oct. 2022.

[33] S. Zhang, H. Tong, L. Jin, Y. Xia, and Y. Guo, "Balancing consistency and disparity in network alignment," in *Proc. 27th ACM SIGKDD Conf. Knowl. Discov. Data Mining*, 2021, pp. 2212–2222.

[34] X. Du, J. Yan, and H. Zha, "Joint link prediction and network alignment via cross-graph embedding," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, 2019, pp. 2251–2257.

[35] X. Du, J. Yan, R. Zhang, and H. Zha, "Cross-network skip-gram embedding for joint network alignment and link prediction," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 03, pp. 1080–1095, Mar. 2022.

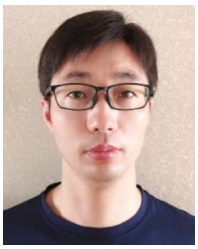
[36] W. Tang, H. Sun, J. Wang, Q. Qi, H. Chen, and L. Chen, "Cross-graph embedding with trainable proximity for graph alignment," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 12, pp. 12556–12570, Dec. 2023.

[37] W. Tang et al., "Multi-order matched neighborhood consistent graph alignment in a union vector space," in *Proc. 46th Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2023, pp. 963–972.

[38] Y. Ren, C. C. Aggarwal, and J. Zhang, "ActiveIter: Meta diagram based active learning in social networks alignment," *IEEE Trans. Knowl. Data Eng.*, vol. 33, no. 5, pp. 1848–1860, May 2021.

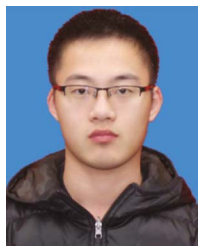
[39] S. Zhang and H. Tong, "FINAL: Fast attributed network alignment," in *Proc. 22th ACM SIGKDD Conf. Knowl. Discov. Data Mining*, 2016, pp. 1345–1354.

[40] T. T. Huynh et al., "Network alignment with holistic embeddings," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 2, pp. 1881–1894, Feb. 2023.
[41] Z. Zhang, L. Sun, S. Su, J. Qu, and G. Li, "Reconciling multiple social networks effectively and efficiently: An embedding approach," *IEEE Trans. Knowl. Data Eng.*, vol. 33, no. 1, pp. 224–238, Jan. 2021.
[42] S. Saxena, R. Chakraborty, and J. Chandra, "HCNA: Hyperbolic contrastive learning framework for self-supervised network alignment," *Inf. Process. Manage.*, vol. 59, no. 5, 2022, Art. no. 103021.
[43] J.-D. Park, C. Tran, W.-Y. Shin, and X. Cao, "GradAlign+: Empowering gradual network alignment using attribute augmentation," in *Proc. 31st ACM Int. Conf. Inf. Knowl. Manage.*, 2022, pp. 4374–4378.
[44] W. Tang et al., "Identifying users across social media networks for interpretable fine-grained neighborhood matching by adaptive gat," *IEEE Trans. Serv. Comput.*, vol. 16, no. 5, pp. 3453–3466, Sep./Oct. 2023.
[45] B. Perozzi, R. Al-Rfou, and S. Skiena, "DeepWalk: Online learning of social representations," in *Proc. 20th ACM SIGKDD Conf. Knowl. Discov. Data Mining*, 2014, pp. 701–710.
[46] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. 5th Int. Conf. Learn. Representations*, 2017.
[47] T. N. Kipf and M. Welling, "Variational graph auto-encoders," 2016, *arXiv:1611.07308*.
[48] D. Wang, P. Cui, and W. Zhu, "Structural deep network embedding," in *Proc. 22nd ACM SIGKDD Conf. Knowl. Discov. Data Mining*, 2016, pp. 1225–1234.
[49] E. Zhong, W. Fan, J. Wang, L. Xiao, and Y. Li, "COMSOC: Adaptive transfer of user behaviors over composite social network," in *Proc. 18th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2012, pp. 696–704.
[50] M. E. Dickison, M. Magnani, and L. Rossi, *Multilayer Social Networks*. Cambridge, U.K.: Cambridge Univ. Press, 2016.
[51] D. W. Ruck, S. K. Rogers, M. Kabrisky, M. E. Oxley, and B. W. Suter, "The multilayer perceptron as an approximation to a Bayes optimal discriminant function," *IEEE Trans. Neural Netw.*, vol. 1, no. 4, pp. 296–298, Dec. 1990.
[52] H. Tong, C. Faloutsos, and J.-Y. Pan, "Fast random walk with restart and its applications," in *Proc. IEEE 6th Int. Conf. Data Mining*, 2006, pp. 613–622.
[53] L. Van der Maaten and G. Hinton, "Visualizing data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, no. 11, pp. 2579–2605, 2008.
[54] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," in *Proc. 6th. Int. Conf. Learn. Representations*, 2018.
[55] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Proc. Int. Conf. Adv. Neural Inf. Process. Syst.*, 2017, pp. 1025–1035.
[56] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, "LINE: Large-scale information network embedding," in *Proc. 24th Int. Conf. World Wide Web*, 2015, pp. 1067–1077.



Hai-Feng Zhang received the B.Sc. degree from Fuyang Normal University, Fuyang, China, in 2003, the M.Sc. degree from Shanghai University, Shanghai, China, in 2006, and the Ph.D. degree from the University of Science and Technology of China, Hefei, China, in 2011. He is currently a Professor with the School of Mathematics Science, Anhui University, Hefei. His main research interests include theories and applications of complex networks and network science.

Guojing Ren received the B.Eng. degree from Southeast University, Nanjing, China, in 2018. He is currently working toward the M.Eng. degree with the Institutes of Physical Science and Information Technology, Anhui University, Hefei, China. His research interests include complex networks and machine learning.



Xiao Ding received the B.Sc. degree from Fuyang Normal University, Fuyang, China, in 2019. He is currently working toward the Ph.D. degree with the School of Mathematics Science, Anhui University, Hefei, China. His research interests include complex networks and machine learning.

Li Zhou received the B.Sc. degree from Fuyang Normal University, Fuyang, China, in 2004, the M.Sc. degree in from Southeast University, Nanjing, China, in 2007. She is currently a Lecturer with the College of Science, Anhui Agricultural University, Hefei, China. Her research interests include complex networks and privacy protection.

Xingyi Zhang (Senior Member, IEEE) received the B.Sc. degree from Fuyang Normal University, Fuyang, China, in 2003, and the M.Sc. and Ph.D. degrees from the Huazhong University of Science and Technology, Wuhan, China, in 2006 and 2009, respectively. He is currently a Professor with the School of Computer Science and Technology, Anhui University, Hefei, China. His research interests include unconventional models and algorithms of computation, multiobjective optimization, and network science. Prof. Zhang was the recipient of the 2018 IEEE Transactions on Evolutionary Computation Outstanding Paper Award and the 2020 IEEE Computational Intelligence Magazine Outstanding Paper Award.