Graph Alignment Neural Network Model With Graph to Sequence Learning

Nianwen Ning, Bin Wu, Haoqing Ren, and Qiuyue Li

Abstract—**Network alignment aims at detecting the corresponding entities across multiple networks, which is an essential basis for the fusion and analysis of multiple network information. Moreover, embedding-based network alignment has gradually become one of the promising methods. However, existing methods ignore the confusing selection problem caused by the similarity-orientated principle of network embedding and over-dependence on the hypothesis of structural consistency. In this paper, we propose an end-to-end Graph Alignment Neural Network (GANN) model with graph-to-sequence learning. GANN mainly consists of two modules: Graph encoder and Sequence decoder. In graph encoder module, we present a restricted network embedding method, which can not only capture the local structure and attribute information of nodes but also realize the constraint of node embedding and space reconciliation. In sequence decoder module, we propose a graph-to-sequence learning model to address large graphs' structural consistency hypothesis problem. In this model, an attention-based LSTM mechanism is introduced to infer a node in the source network corresponding to the candidate node sequence in target networks. In this candidate sequence, the correct aligned node is placed at the top. We demonstrate that GANN outperforms the state-of-the-art methods in network alignment tasks on various real-world datasets.**

Index Terms—**Graph neural network, graph to sequence learning, network alignment.**

I. Introduction

NODE alignment (NA) problem aims to identify and infer the corresponding nodes across multiple networks, as shown in Fig. 1. The black dotted line is the corresponding relation between a pair of aligned nodes, and this relation is called an anchor link. Bob is the real-world node corresponding to the aligned nodes Q1 and S1 in the two networks. Network alignment task that predicts these links for bridging multiple networks is also called anchor link prediction. Intuitively, NA is the basis for the knowledge discovery and fusion of multiple networks. Taking social networks as an example, most people can simultaneously enjoy the services provided by multiple online social networks. However, information about multiple accounts belonging to the same user is not unknown due to privacy issues or restricted access. Considering that multi-source, multi-perspective, and multi-dimensional heterogeneous network data can provide richer information, the integration of the heterogeneous relations information of users can achieve more accurate user behavior analysis. Network alignment not only plays a vital role in social network analysis but also helps to improve many other applications, such as cross-domain recommendation [1], biological protein-protein comparison [2], multilingual knowledge discovery [3], blue knowledge graph fusion [4] and psychology analysis [5]. Therefore, NA is a crucial method to address information silo problems and joint mining of multiple network data.

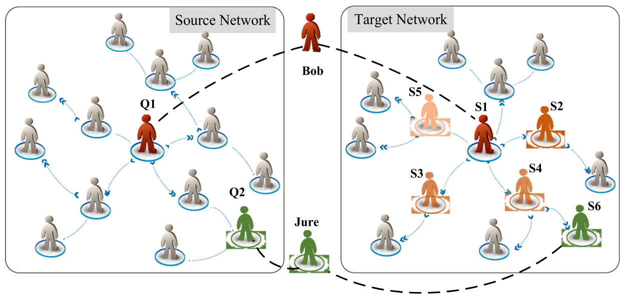


Fig. 1. Alignment illustration of two social networks.

Previous researchers proposed to solve this problem by using available auxiliary information, such as user profile [6] (screen name, gender, age, occupation, etc.), user-generated content [7] (posts, blogs, comments, etc.) and other demographic characteristics [8] (location, trajectory, etc.). However, with the increasing data privacy and information awareness, this information is becoming more difficult to access and collect. How to use known network structure information to infer alignment relation has become a significant problem. Due to the superior performance of network representation learning (also known as network embedding) methods in the extraction of network structure features, network alignment based on network representation learning has gradually become one of the promising directions. According to different embedding methods, network alignment methods

Manuscript received 31 May 2021; revised 17 August 2023; accepted 26 October 2023. Date of publication 15 November 2023; date of current version 7 August 2024. This work was supported in part by the National Key Research and Development Program of China under Grant 2018YFC0831500, in part by the National Natural Science Foundation of China under Grant 61972047, in part by NSFC-General Technology Basic Research Joint Funds under Grant U1936220, and in part by the Key Science and Technology Research Project of Henan Province of China under Grant 222102210067. Recommended for acceptance by P. Bogdanov. *(Corresponding author: Bin Wu.)*

Nianwen Ning is with Henan University, Henan 475001, China, and also with the Beijing University of Posts and Telecommunications, Beijing 100876, China (e-mail: nnw@henu.edu.cn).

Bin Wu is with the School of Computer Science, Beijing University of Posts and Telecommunications, Beijing 100876, China (e-mail: wubin@bupt.edu.cn).

Haoqing Ren and Qiuyue Li are with the Beijing University of Posts and Telecommunications, Beijing 100876, China (e-mail: bycsrhqq@bupt.edu.cn; qiuyueli@bupt.edu.cn).

This article has supplementary downloadable material available at https://doi.org/10.1109/TKDE.2023.3329380, provided by the authors.

Digital Object Identifier 10.1109/TKDE.2023.3329380

can be divided into three types. These alignment methods based on matrix factorization embedding [9], [10], [11], [12] can realize unsupervised and semi-supervised network alignment by low-rank structure recognition, spectral method, and random block model. However, these methods are difficult to be used in large-scale networks. Although these network alignment methods based on random walk embedding [6], [13], [14], [15], [16] are scalable, these methods do not consider linking mode rules and the network structure correlation between known alignment node pairs. In addition, these two types of methods cannot make full use of node attributes or edge attribute information. In order to integrate network structure and node attributes for improving the accuracy of network alignment, network alignment methods based on graph neural networks (GNNs) [17], [18], [19], [20], [21], [22], [23] have become a hot spot of current researches.

Despite network alignment has been focused on by many researchers, there are still many problems that have not been well resolved. The current existing alignment methods based on network embedding mainly have the following two challenges:

1) *Over-dependence problem on the hypothesis of structural consistency:* Most studies are based on the assumption that the alignment nodes have a similar local structure, that is, the structural consistency assumption. However, this assumption may not work in some real-world scenarios. The main reasons are that the scale of real networks is different, and nodes have different linking models in different scenarios. For some online social networks, users have different social preferences for different types of relations. For example, on a LinkedIn network, most people can establish friendships with career and work-related users. On a Facebook network, most users can interact with their real-life friends. In Fig. 1, as far as a pair of aligned nodes Q1 and S1 are concerned, they do not have consistent local structures. Therefore, simply relying on the consistency of local structural information can seriously affect the accuracy of network alignment task.

2) *Confusing selection problem caused by similarity orientation of network embedding methods:* Most of the existing representation learning methods are a similarity-oriented principle, making it challenging to find the best corresponding node more accurately after locking the local structure. For preserving the similarity between a node and its neighbors, GNNs can learn the embedding function of a pair of neighbors in euclidean space. However, due to the difference in embedding semantics and the inconsistency of local structure, this strategy can not accurately identify which node is correct in the selected candidate node. As shown in Fig. 1, the aligned nodes in the target network are found for node Q1 in the source network. After similarity matching, it can be seen that Q1 and the candidate node set {S1, S2, S3, S4, S5} have high similarity. Then, due to similarity-oriented reasons, it is not easy to find the best unique candidate node S1 from this candidate node set.

To solve these challenges, inspired by Graph to Sequence (Graph2seq) Learning [24], [25], [26], [27], we design a novel and flexible Graph Alignment Neural Network (GANN) model for generating the ordering sequence of nodes. GANN first captures the sequence information of the local structure by a breadth-first search algorithm and combines it with an unsupervised graph convolutional neural network to learn latent representations of nodes. Considering a ”one-to-one” constraint of network alignment, we construct an embedding space alignment across networks to select a set of candidate nodes that are most similar to the representation of a target node. This phase mitigates the structural consistency assumptions about edges by selecting the set of locally similar nodes in the potential embedding space as a coarse-grained alignment result, which weakens the effect of edge distribution difference in the two networks to be aligned. Then, we execute the proposed sequence generation model to rank these nodes in the candidate node set so that the most relevant node is placed at the top of the sequence. In this stage, an attention-based LSTM mechanism is used to sequence these nodes for capturing the specificity information among these candidate nodes, and then match with a target node to identify a correctly aligned node. This stage can focus on the difference between nodes by the manner of sequencing nodes, and alleviate the confusing selection problem and achieve accurate matching between the target node and its aligned nodes. In Fig. 1, with respect to node Q1 in the source network, the set of most possible candidate nodes aligned with node Q1 in the target network is {S2, S1, S4, S3, S5}. Combining the embedding representation of node Q1 with the proposed graph-sequence generative model, we sequence nodes in the candidate node set so that the correctly aligned node is placed at the first place of the sequence, i.e., [S1, S2, S3, S4, S5].

The main contributions can be summarized as follows:

- We propose a node embedding-based network alignment model. Unlike most existing methods to learn a mapping function between networks, the encoder-decoder architecture is adopted to generate a candidate node sequence according to alignment probability. To our knowledge, we are the first to use the idea of Graph2Seq to perform network alignment.
- We propose a graph alignment neural network (GANN) model. Considering the confusing selection problem of network alignment based on similarity orientation, GANN utilizes the “One to One” matching principle that the embedding of aligned nodes has strong similarity and weak similarity with other known aligned nodes to realize the embedding constraint and space reconciliation.
- We propose a node sequence generation model based on the attention-based LSTM for the node matching of cross-networks. Considering the difference between nodes in local network structure, the model preserved the different proximity of nodes by ordering neighbors to improve node matching efficiency.
- Extensive evaluations on real-world datasets with different scales and types have been conducted, and the experimental results verify the superiority of the proposed GANN model.

The rest of this paper is organized as follows. Section II gives the overview of related researches. Section III elaborates on the definition of the network alignment problem and the alignment matching problem. Section IV introduces in detail the graph alignment neural network model based on graph-sequence generation. Section V specifically analyzes the

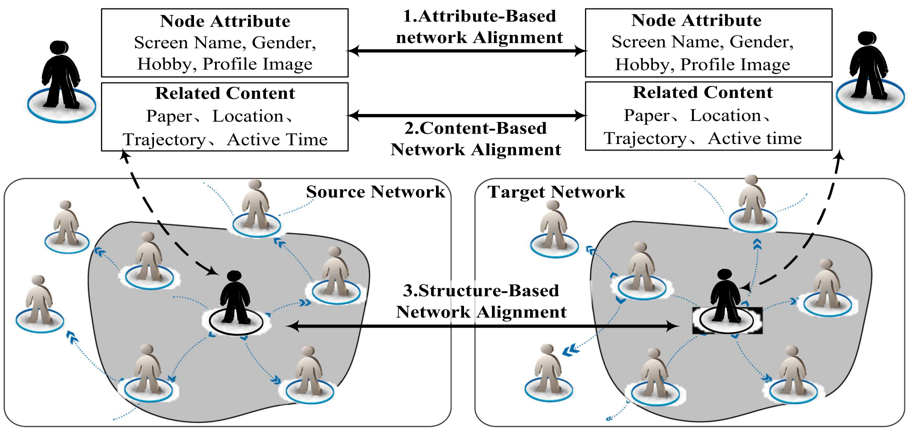

Node Attribute
Screen Name, Gender, Hobby, Profile Image
Related Content
Paper、Location、Trajectory、Active Time
1.Attribute-Based network Alignment
2.Content-Based Network Alignment
3.Structure-Based Network Alignment
Source Network
Target Network

Fig. 2. Classification of network alignment for social network: 1) profile-based network alignment methods; 2) content-based network alignment methods; 3) network structure-based network alignment methods. In these source network and target network, nodes with black represent a pair of aligned nodes.

results of model performance experiments, parameter sensitivity experiments, and ablation verification experiments. Section VI is the conclusion of this paper.

II. Related Work

The essence of the network alignment (NA) problem is network isomorphism, which is to distinguish the same entities in different networks so that the same entities in multiple networks establish a link relationship. NA can solve ”information silo” problem of multiple network data and enable multiple networks to conduct joint analysis and mining. Therefore, the network alignment problem is an important foundation for multiple network analysis and mining. Related researches in recent years are mainly divided into two categories. One is the alignment method of traditional networks, which uses the attributes [6], content [7], structural similarity [28], [29], [30], and interconnection patterns [31] between nodes in multiple homogeneous and heterogeneous networks to align networks. Another is a network alignment method based on representation learning, which studies how to achieve the corresponding relationship reasoning through the representation of nodes and elements between different networks and modals. Take social networks as an example in Fig. 2. If users have different screen names or different register information, it is difficult to align network by simply relying on user profile. Network alignment methods based on representation learning can more comprehensively integrate the structural and incidental information (node attributes) of nodes.

A. *Network Alignment Method Based on Structural Representation Learning*

This type of method is mainly to solve the quality of traditional network alignment algorithms for extracting structural features, that is, from manual setting or linear matrix decomposition method to automatic structural representation learning, which can more effectively preserve the local and global structural information of nodes.

1) *Alignment methods based on matrix factorization network embedding:* Heimann et al. [10] proposed a framework (REGAL) to use a cross-network matrix decomposition model for node embedding and a K-D tree for node matching. For multi-layer networks with only a few nodes aligned or not aligned, Gomes et al. [32] proposed a joint random block model (Joint SBM) for evaluating shared communities across heterogeneous non-aligned networks. Tang et al. [11] developed an iterative degree penalty algorithm (IDP) that includes a penalty principle for inter-layer link prediction in multi-layer networks. Zhang et al. [12] proposed a multi-level network alignment algorithm (MOANA). The algorithm contains three key steps. First, the input network is effectively coarsened into the structural representation, then the coarse-grained representation of the input network is aligned, and finally, interpolation is performed to obtain a multi-level alignment scheme including the most fine-grained node level.

2) *Alignment methods based on random walk network embedding:* Zheng et al. [33] constructed a heterogeneous interaction graph (HIG) by connecting Taobao logs and user identities and transformed the user alignment problem in e-commerce into a node embedding problem. Zhou et al. [34] proposed an end-to-end semi-supervised learning method by learning node representations to capture local and global network structures. Du et al. [15] think that link prediction and network alignment are two basic and interleaved tasks in network analysis, so they proposed a cross-network embedding model by alternately performing link prediction and network alignment. Zhou et al. [35] proposed an efficient and conceptually simple, flexible, and universal framework (Meta-NA), which reformulated the network alignment problem as a simple classification problem through the meta-learning framework of graphs. Zhou et al. [6] presented an unsupervised user identification algorithm (FRUI-P), which first aggregates the friend feature of each user, then calculates the similarity of all candidates in the two networks. Chu et al. [36] proposed a cross-network embedded multi-network alignment method (Cross-MNA) in order to solve the previous method's excessive dependence on structural consistency assumptions and node attributes.

Although the accuracy of network alignment can be improved by simply relying on structure information, it can be affected by the noise of networks. Moreover, different nodes play different roles in different networks, and the local structure formed under different roles can also be different. Therefore, it is better to integrate other information to improve the accuracy of prediction.

B. Network Alignment Methods Based on Structure and Attribute Representation Learning

Tan et al. [37] presented a hypergraph-based manifold alignment algorithm (MAH). Kong et al. [38] proposed a method for entity alignment across multiple heterogeneous data sources using a probabilistic generative model. Zhang et al. [17] proposed a series of network alignment algorithms to effectively align the attribute network (FINAL), which uses node/edge attribute information to guide the (topology-based) alignment process and describes this problem as a convex quadratic optimization problem.

Due to the superior performance of *graph neural network (GNN)*in network representation learning, the research on network alignment based on GNN has attracted more and more attention from scholars in recent years.

1) *Network alignment methods based on graph neural network:* Sun et al. [39] proposed an alignment network model based on the knowledge graph. It aims to alleviate the heterogeneity of the neighborhood structure in an end-to-end manner. Qin et al. [18] proposed a framework (G-CREWE), which embeds nodes as node representations of two resolution levels to achieve effective network alignment. Chen et al. [19] showed a multi-level graph convolution framework (MGCN) that considers both local network structure and hypergraph structure. Ren et al. [20] proposed an iterative network alignment model based on active learning (ActiveIter), which defines a set of network meta-graphs for anchor link feature extraction. Ye et al. [40] proposed a relation vectorized graph convolutional network (VR-GCN) to learn entity representation and relation representation. Trung et al. [41] presented a completely unsupervised network alignment framework (GAlign) based on a multi-level embedding model. Gao et al. [42] explored the consistency of the local structure, and then constructed a matching graph to avoid matching conflicts, and proposed a graph convolutional network (GCN-ALP) with a small batch processing strategy. Since the direct influence of inherent network attribute information on the alignment results is still largely unknown, Zhou et al. [43] proposed an active network alignment method (Attent) to determine the best query node in response to this problem.

2) *Network alignment methods based on attention mechanism:* In order to solve node representations ignoring the impact of local information on the alignment of user nodes, the attention mechanism is introduced into the design of network alignment model. Li et al. [44] demonstrated a cross-heterogeneous network-type perceptual anchor link prediction model (TALP) based on a two-layer graph attention architecture. Ren et al. [21] studied the alignment problems of social networks and user behavior analysis problems and designed an end-to-end network alignment framework based on the attention mechanism (BANANA). They designed a temporal graph attention network component to fuse behavioral information in different social networks.

3) *Network alignment methods based on adversarial neural network:* Pei et al. [3] aimed to formulate and study robust entity alignment problems, and proposed a robust entity alignment method (REA). Hong et al. [22] obtained a domain invariant representation for network alignment through an adversarial domain classifier and proposed a deep unified architecture (DANA), which uses graph convolutional networks to perform network embedding under the principle of domain confrontation. Nguyen et al. [45] proposed a network alignment framework (NAWAL) based on an end-to-end unsupervised embedding. The model uses a generative-adversarial network to align the space without relying on hand-made features or supervised learning in a specific field.

Shu et al. [46] reviewed the research on user identity link inference across online social networks. They divided the existing network alignment methods into two categories: methods based on feature extraction and methods to build predictive models from multiple perspectives. In addition, this paper also summarizes the benchmark data sets and the evaluation indicators of the corresponding algorithms about the alignment direction of social networks. Zhou et al. [47] summarized related user mining methods for social network integration, which mainly include user attributes, user relationships, and integrated information. Trung et al. [48] proposed network alignment benchmark methods and datasets, which provide a comprehensive empirical study for the performance comparison of network alignment methods.

In summary, the existing network alignment methods based on graph neural networks are mainly implemented on the basis of structural consistency assumptions. Moreover, most of the network alignment algorithms based on node embedding overemphasize the similarity between nodes in the embedding stage, which leads to confused selection problems in the subsequent node matching process, which affects the accuracy of the network alignment method. In order to alleviate these problems existing in the above existing work, we proposed a graph alignment neural network model based on graph-to-sequence learning.

III. Preliminaries and Problem Formulation

Definition 1. Network Embedding Problem (also known Network Representation Learning): Give a network $G = \{V, E\}$, in respect of a node $u \in V$, the goal of network embedding to find a function $f : u \rightarrow \mathcal{R}^d$, where $d \ll |V|$.

This function f can capture the structural properties of nodes, edges, or networks and embeds these features to a low-dimensional and dense vector. If two nodes are similar in the network G, this similarity should also be preserved in the embedded space. Current network embedding methods are mainly divided into three categories: (1) network embedding methods based on matrix factorization, (2) network embedding methods

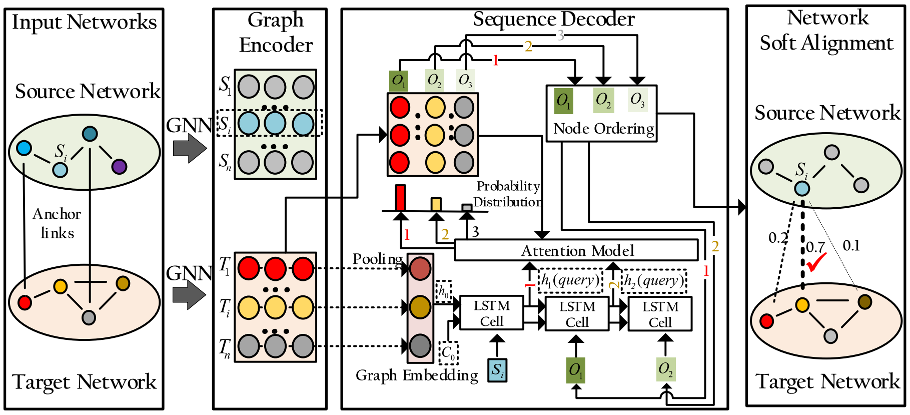


Fig. 3. Neural network model of topological sorting alignment graph with alignment constraints. The model is mainly divided into three stages: 1) preprocessing stage; 2) network embedding for graph encoder module; 3) node sequence generation for sequence decoder module.

based on random walk, (3) network embedding methods based on deep learning.

Definition 2. Network Alignment Problem (also known anchor link prediction): Given a pair of networks $G_s = \{V_s, E_s\}$ and $G_t = \{V_t, E_t\}$, the goal of the network alignment problem is to refer the aligned node set $S_{anchor} = \{(u, v) | u \in V_s, v \in V_t\}$. Generally, this problem is treated as a binary classification problem. Given a pair of nodes (u, v) where $u \in V_s$, $v \in V_t$, to predict whether there is a link between nodes u and v.

This definition is mainly for a pair of networks. The transitivity of alignment relations can be simply extended to the alignment problem between multiple pairs of networks. In addition, the problem can also be modeled as a regression problem, that is, to predict the probability of anchor link existence between node pairs.

Definition 3. Network Matching Problem: *One-to-one network alignment (also known Hard alignment)* maps one node of a given network G_s to at most one node of another network G_t: $f(u, G_s, G_t) \rightarrow (v, G_t)$, where node $u \in V_s$ and node $v \in V_t$.

One-to-many network alignment (also known Soft alignment) maps one node of a given network G_s to multiple nodes of another network G_t: $f(u, G_s, G_t) \rightarrow (\mathbf{V}, G_t)$, where $\mathbf{V}$ is a set of several nodes in network G_t, $\mathbf{V} \subset V_t$.

Many-to-many network alignment maps a group of nodes in a given network G_s to a group of nodes in another network G_t: $f(\mathbf{U}, G_s, G_t) \rightarrow (\mathbf{V}, G_t)$, where $\mathbf{U}$ is a set of several nodes in network G_s, $\mathbf{U} \subset V_s$.

One-to-one and one-to-many may have better results because of using the correct correspondence between nodes or the number of saved edges as the standard method to evaluate the network alignment output. However, many protein gene duplications, mutations, and interaction rewiring events have occurred during evolution in biological networks. In addition, proteins/genes usually function in the form of complexes or modules and manifest themselves in the form of communities in biological networks. Therefore, many-to-many node mapping can be considered more reasonable and closer to the actual situation because it can align complexes/modules with similar functions between different networks (species). In other words, it is difficult to find a perfect matching neighborhood topology between nodes in different biological networks. It is difficult to evaluate the topological quality of many-to-many node mapping compared with one-to-many and one-to-many mapping. The latter two have been more widely studied in the existing researches. Although these three types of node mapping are suitable for different purposes, we mainly study one-to-one network alignment and one-to-many network alignment problems in this paper.

IV. The Proposed Model

The model proposed in this paper mainly consists of three stages: 1) preprocessing stage; 2) network embedding for Graph encoder module; 3) node sequence generation for Sequence decoder module. The framework of the model is shown in Fig. 3.

A. Preprocessing Stage

To train an unsupervised graph neural network model, we need to sample the neighboring nodes around each node to capture its local structure, similar to DeepWalk and Node2Vec methods. Besides, our model is a semi-supervised learning model, so we need to construct a label set for the training of aligned anchor node pairs in this paper. For one-to-many network node alignment tasks, to evaluate the performance of the algorithm, the best alignment model should be to sequence nodes in the candidate node set by alignment probabilities so that the correct node is placed first in the sequence, and other candidate nodes should be some neighbors of the correct node. Therefore, to solve these two key problems simultaneously, we use a breadth-first search method to sample nodes and obtain node sequences.

For a pair of nodes (v, u), a node sequence constructed by breadth first search starting from a node u is treated as a ground truth sequence of node v for training GANN in this paper instead of considering only a single aligned node. Most of the existing network alignment methods based on network embedding usually use the “one-to-one” node matching manner. This manner causes a similarity-oriented confusing selection problem, so we utilize the ”one-to-many” manner to capture the difference between nodes for alleviating the problem.

Suppose the anchor link (v, u) is known, where v also is expressed as v^s, denotes this node in source network s. u also be expressed as u^t, denotes this node in target network t. A training label $N^v_{Sequence}$ for v is obtained by treated u in network t as a start node:

$$N^v_{Sequence} = \left[u^t, u^t_1, u^t_2, \ldots, u^t_{len}\right], \tag{1}$$

which len is the length of the sequence. The purpose of (1) design is to capture the local adjacency information of the initial node. The characteristic of this sequence is that it is closely connected to the target node u at the front part of the sequence, and the higher-order neighbors of u are at the end of the sequence. In order to introduce the algorithm described in this subsection more clearly, we label a pair of networks as source network s and target network t, respectively.

B. Network Embedding for Graph Encoder Module

For taking the structural information and the attribute information of nodes into account, a graph convolutional neural network is used to realize the graph encoder module. The initial input matrix $\mathbf{H}^{(0)} = \mathbf{H}$ is a node attribute matrix or node adjacency matrix. Assuming that the output of node i in k-th convolution layer is $\mathbf{x}^{(k)}_i$, we denote the embedding result output of node i in previous layer $k-1$-th as $\mathbf{H}^{(k-1)}_i$. The k-th neural layer of graph convolutional is given as,

$$\mathbf{x}^{(k)}_i = \mathbf{W}^{(k)}_1 \mathbf{H}^{(k-1)}_i + \mathbf{W}^{(k)}_2 \cdot \frac{1}{|N(i)|} \sum_{j \in N(i) \cup (i)} \mathbf{H}^{(k-1)}_j, \tag{2}$$

where $N(i)$ is neighbors of node i, $|\cdot|$ is a function that counts the number of elements, $\mathbf{W}^{(k)}_1$ and $\mathbf{W}^{(k)}_2$ are the parameters that need to be learned for the first layer of graph convolution. Considering the complexity of the model and the efficiency of learning parameters, the results of graph convolutional networks with two layers are regarded as the graph encoder module of GANN. Assuming that a graph convolutional network with k layers is used to GANN, we use the negative sampling strategy in the Node2Vec method to construct a loss function of node embedding. Corresponding to one example edge $e(i, j)$ of all edges in the source network s, the global negative sampling strategy is to preserve the embedding $\mathbf{x}^{(k)}_i$ of node i similar to the embedding $\mathbf{x}^{(k)}_j$ of node j corresponding to one edge of all edges in the source network s (or target network t). The objective function of node embedding in intra-layer is:

$$\begin{aligned} O_{\text{embedding}} = & \sum_{(i,j) \in E} \log \left[sigmod \left({\mathbf{x}^{(k)}_i}^{\top} \mathbf{x}^{(k)}_j \right) \right] \\ & + \sum_{n_e=1}^{M_i} E_{v_k \propto P(i)} \left[\log \left(1 - \sigma \left({\mathbf{x}^{(k)}_i}^{\top} \mathbf{x}^{(k)}_{n^i_e} \right) \right) \right] \\ & + \sum_{n_e=1}^{M_j} E_{v_k \propto P(j)} \left[\log \left(1 - \sigma \left({\mathbf{x}^{(k)}_j}^{\top} \mathbf{x}^{(k)}_{n^j_e} \right) \right) \right], \end{aligned} \tag{3}$$

where k denotes the number of graph convolutional layer, M_i is the number of negative sample nodes in respect of i, E is the edge set in each network, $\mathbf{x}^{(k)}_{n_e}$ denotes the embedding of the n^i_e negative sample node of node i, and σ is a activation ReLU function. This loss function aims to preserve the structure information and attributes information of each network. In order to clearly introduce the other modules of GANN proposed in this paper, we define $\mathbf{X}^{(k)}_s$ denotes the embedding result of source network s, $\mathbf{X}^{(k)}_t$ denotes the embedding result of the target network t. Then, the embedding of node v in the source network s is represented as $\mathbf{X}^{(k)}_s[v, :]$, and the embedding of node u in the target network t is represented as $\mathbf{X}^{(k)}_t[u, :]$.

To ensure that each network in the cross-network alignment problem can be embedded in the same space, we need to adjust the separate embedding of the multiple networks. Considering the labeling cost, sparsity, and reduction of the model's learning parameters, we use direct constraints to achieve embedding space reconciliation instead of through linear mapping of embedding again. In the graph encoder module, to consider the "one to one" constraint principle of network alignment task, in respect of a pair aligned node (v, u), the embedding $\mathbf{X}^{(k)}_s[v, :]$ of node v and the embedding $\mathbf{X}^{(k)}_s[u, :]$ of node u should be similar. Their embedding vectors should be dissimilarity to other aligned anchor nodes. Therefore, we utilize a constraint function to realize the embedding nodes in inter-layers by:

$$\begin{aligned} O_{\text{anchor}} = & \sum_{(v,u) \in S_{\text{anchor}}} \left\| \mathrm{X}^{(k)}_1[v, :] - \mathrm{X}^{(k)}_2[u, :] \right\|^2 \\ & - \sum_{(v,u),(p,q) \in S_{\text{anchor}}} \begin{pmatrix} \left\| \mathrm{X}^{(k)}_1[v, :] - \mathrm{X}^{(k)}_2[q, :] \right\|^2 \\ + \left\| \mathrm{X}^{(k)}_1[p, :] - \mathrm{X}^{(k)}_2[u, :] \right\|^2 \end{pmatrix}, \end{aligned} \tag{4}$$

where $\| * \|$ denotes the euclidean distance of the matrix, and S_{anchor} denotes the set of known aligned nodes. (v, u) and (p, q) are two anchor links. In order to realize the "one-to-one" constraint principle of nodes, three embedding constraints are designed in (4). First, the first term on the right of the equal sign indicates that two embedding vectors of the anchor link pair should be more similar in the embedding space. The second term on the right of the equal sign indicates that a node v (or u) embedding in the anchor link (v, u) should be dissimilar as a node q (or p) embedding in another anchor link (p, q). This second item is also different from existing network alignment networks based on representation learning. This objective function realizes the constraints of network embedding learning separately. The goal is to make an aligned node pair more similar and dissimilar to other aligned nodes. The network embedding process has certain specificity information, which is beneficial to the node's accurate matching.

C. Node Sequence Generation for Sequence Decoder Module

We propose a node sequence generation model based on a graph to sequence learning for sequence decoder. This model can improve the efficiency of calculation in node matching processes and alleviate the over-dependence problem on the hypothesis of structural consistency. First, we adopt a k-means method to cluster nodes for coarse-grained alignment. Each node in the source network is used as the clustering center for clustering nodes of the target network. After this node clustering process, the local structure where the corresponding anchor node is located in the target network is locked. For example, in large-scale social networks, we first find the circle of friends that is most likely to align nodes. Use sorting mode to sort a small number of nodes in this circle of friends, so that the most likely aligned nodes are placed at the top of the sequence. Then, in the fine-grained alignment process, we transform the node matching process of network alignment into a combinatorial optimization problem where the output dictionary size depends on the number of elements in the input sequence [49], [50]. The idea of hierarchical computing can introduce differentiated local information, which not only accurately locks the local positions of the most likely aligned nodes, but also improves the efficiency of the alignment process. Based on the process, node sequence generation is conducted to sequence these nodes in the local structure as the result of network alignment. For fine-grained process, the attention-based LSTM is used to predict the correct aligned candidate node sequence. Assuming a pair of nodes (v, u) of anchor link, in respect of node v in source network, the conditional probability p of the prediction sequence $N_{pre}^{v} = [u_1, u_2.., u_{len}]$ given the input target network t can be formalized as:

$$P\left(N_{pre}^{t}\right) = \prod_{j=1}^{len} p\left(u_j \mid [u_1, \ldots, u_{j-1}], \mathbf{x}_t^{(k)}, \mathbf{x}_s^{(k)}[v,:]; \theta\right), \tag{5}$$

which θ is a parameter of sequence generation model, len is the length of sequence. Eq(5) provides a formulaic definition for node sequence generation in respect of node embedding.

(1) *Sequence Decoder Based on LSTM:* We use Long Short-Term Memory (LSTM) method to model the learning process of (5) with parameters. For each node v in the source network s, the gated state transition operation for the hidden state $\mathbf{h}_r$ of a node at the r-th position in the candidate sequence can defined as:

$$\begin{aligned} i_r &= \sigma\left(W_i \mathbf{X}_s^{(k)}[v,:] + U_i \mathbf{h}_{r-1}\right), \\ o_r &= \sigma\left(W_o \mathbf{X}_s^{(k)}[v,:] + U_o \mathbf{h}_{r-1}\right), \\ f_r &= \sigma\left(W_f \mathbf{X}_s^{(k)}[v,:] + U_f \mathbf{h}_{r-1}\right), \\ \widetilde{c}_r &= \tanh\left(W_u \mathbf{X}_s^{(k)}[v,:] + U_u \mathbf{h}_{r-1}\right), \end{aligned} \tag{6}$$

which i_r, o_r, f_r are the input, output and forget gates, respectively. $W_i, W_o, W_f, W_u, U_i, U_o, U_f, U_u$ are LSTM model parameters, $\sigma(\cdot)$ is Sigmoid activation function. (6) aims to use LSTM capture to generate the positional characteristics of node sequences in respect of target network embedding. The internal state $\mathbf{c}_r$ of LSTM is updated by:

$$\mathbf{c}_r = i_r \odot \widetilde{c}_r + f_r \odot c_{r-1}. \tag{7}$$

Notice that $\mathbf{c}_0$ is initialized as a zero matrix, $\odot$ is an element vector product operation.$\mathbf{c}_r$ is used to store historical information up to the current moment. For r-th node of generated sequence, the hidden state $\mathbf{h}_r$ determines the output at the current moment, which is updated according to $\mathbf{c}_r$ and o_r:

$$\mathbf{h}_r = \sigma\left(\frac{\mathbf{h}_{r-1} + \mathbf{h}_r o_r \odot \tanh(\mathbf{c}_r)}{2}\right). \tag{8}$$

(2) *Next Node Selection based on Attention:* The r-th node in the generated sequence in target network t corresponding to node v in source network s is formulated as:

$$v^r = \mathbf{W}_1 \mathbf{h}_r + \mathbf{U}\mathbf{C}_t^{(k)} + \mathbf{W}_2 \mathbf{X}_s^{(k)}[v,:], \tag{9}$$

$$p\left(\mathbf{C}_r \mid \mathbf{X}_t^{(k)}\right) = \text{softmax}(\mathbf{v}^T \tanh(v^r)), \tag{10}$$

which $\mathbf{v}, \mathbf{W}_1, \mathbf{U}, \mathbf{W}_2$ is the parameters that need to be learned in this module, $\mathbf{C}_t^{(k)}$ denotes the embedding of candidate node set by k-means clustering about the target network t. Our specific calculation method is to concatenate the embedding of node v into the embedding matrix $\mathbf{V}_t^{(k)}$ of the target network t, then perform node clustering and find the most similar len nodes in the cluster where node v is located. We regard the locked local area as candidate node sets. $\mathbf{X}_s^{(k)}[v,:]$ is the node v to be aligned in the source network s. (10) calculates attention weights by combining the embedding of each node in the target network with the embedding of nodes in the source network, which serves as an indicator for sorting. The predicted next node r-th in respect of v is the one with the highest coherence probability:

$$\mathbf{o_r} = \underset{N_{len<r}}{\text{argmax}}\ p\left(\mathbf{C}_r \mid \mathbf{X}_t^{(k)}\right), \tag{11}$$

which $N_{len<r}$ denotes the generated node sequence with length less than r. $\mathbf{o_r}$ is the embedding of a node u_r (denotes row or column vector) in matrix $\mathbf{X}_t^{(k)}$, its row index and column index are the indexes corresponding to the elements of the maximum value in u_r. Then, we treat $\mathbf{o_r}$ as the input of next cell in LSTM.

(3) *Initial hidden state of sequence decoder module:* $\mathbf{h}_r$ is the hidden state of LSTM. In this paper, the global embedding of target network is treated as the hidden state $\mathbf{h}_0$ of LSTM. In order to calculate the hidden state, three types of methods can be used to perform pooling on $\mathbf{X}_t^{(k)}$, which are:

- *Mean Pooling:*

$$\mathbf{h}_0 = \sum_{i \in V_t} \mathbf{X}_t^{(k)}[i,:], \tag{12}$$

- *Maximize Pooling:*

$$\mathbf{h}_0 = \max_{i \in V_t} X_t^{(k)}[i,:], \tag{13}$$

- *Minimize Pooling:*

$$\mathbf{h}_0 = \min_{i \in V_t} X_t^{(k)}[i,:]. \tag{14}$$

In the training of GANN phase, the correct node order is known $N_{Sequence}^{v} = [u_t, u_1^t, \ldots, u_{len}^t]$, the initial sequence input of the sequence decoder module of GANN is the embedding of node v of source network s. According to (5), we can finally get the fixed-length node sequence $N^v_{pre} = [u, u_1, \ldots, u_{len}]$ in target network t corresponding to the node v in source network s. The loss function of node sequence generation is formalized as:

$$O_{order}(N^v_{Sequence}, N^v_{pre}) = -\sum_{r=1}^{len} p(\mathbf{C}_r) \log(q(\mathbf{C}_r)). \quad (15)$$

The goal of (15) is to ensure the accuracy of the local structure found by generating a sequence of nodes and calculating the similarity of surrounding nodes obtained through preprocessing steps. Considering the loss function of the node embedding, the space reconciliation and the sequence generation for nodes as the overall loss of the GANN model:

$$O = O_{embedding} + O_{anchor} + O_{order}. \quad (16)$$

Then, we utilize the minimized cross-entropy loss function to optimize and update the parameters of our model through the Adam optimizer.

D. GANN Pseudocode

We present a pseudocode of graph alignment neural network with a network embedding constraint method and Graph-to-Sequence, called GANN. The pseudocode of GANN model is shown in Algorithm 1.

E. Complexity Analysis

The main flow of GANN model is as follows: In lines 2-3, we initialize these parameters of GANN with K layers graph neural network, the number of iterations, the initialization of some parameters of LSTM, and others. In lines 4-5, we sample nodes and conduct ground truth training node sequence of target network t. The time complexity of this step is $O(|V_s| + |V_t| + 2len)$. During the iteration process, in lines 7-12, we conduct the representation learning of the source network s and target network t, and get the embedding of each node in the target network. The per-batch space and time complexity is fixed at $O(\prod_{i=1}^{K} S_i)$, where $S_i, i \in \{1, \ldots, K\}$, which K is the layer number of graph neural network. The expected runtime of a single batch is unpredictable and in the worst case $O(d|V|)$, which d denotes the embedding dimension. In line 14, we conduct k-means clustering method in respect of target network t by regarding v as a central node in source network s with the time complexity $O(len|V_t|)$. In line 15, we update node embedding $\mathbf{X}_s^{(k)}$ and $\mathbf{X}_t^{(k)}$ and calculate loss $O_{\text{embedding}}$ with the time complexity $O(|V_s| + |V_t|)$. In line 16, we update node embeddings of anchor node set by (4) with the time complexity $O(|\mathcal{S}_{anchor}| \cdot len)$. In line 17, we calculate h_i^d by (12) with the time complexity $O(|V_t|)$. In line 18, we calculate the hidden state of the LSTM for node sequence generation with the time complexity $O(|V_t| \cdot d)$ and predict node sequence N^v_{pre} is generated with the time complexity $O(|V_t|)$. In line 19, we calculate the cross entropy loss O_{order} according to $N^v_{sequence}$ and N^v_{pre}. In line 21, we update the number of iterations T. Iterate the above process and update the parameters of GANN until the result is stable or reaches the maximum number of iterations. Based on the parameters in the experimental setup and

Algorithm 1: Sample Output of -Snes_Monitor -Ksp_Monitor.

```
Input: A pair of graph $G_s = \langle V_s, E_s, L_s, X_s \rangle$;
       $G_t = \langle V_t, E_t, L_t, X_t \rangle$; $S_{anchor}$ is a set of
       anchor nodes; $T$ is a maximize iteration
       parameter; $K$ is the layer number of graph
       neural network; $len$ is the size of sample
       nodes.
1  begin
2      Initialize all parameters of GANN, respectively.
3      $t = 1$
4      Conduct the preprocess and obtain Eq.(1).
5      Conduct the preprocess and obtain Eq.(1).
6      while $t \geq T$ or not converge do
7          for $v$ in $V_s$ do
8              Calculate $\mathbf{X}_s^{(k)}$ by Eq.(2) .
9          end
10         for $v$ in $V_t$ do
11             Calculate $\mathbf{X}_t^{(k)}$ by Eq.(2).
12         end
13         for $v$ in $\mathcal{S}_{anchor}$ do
14             Conduct k-means clustering to the target
                 network by regarding $v$ as a central node.
15             Update node embedding $\mathbf{X}_s^{(k)}$ and $\mathbf{X}_t^{(k)}$
                 by minimizing Eq.(3).
16             Update parameters by minimizing Eq.(4).
17             Calculate $h_i^d$ by Eq.(12), Eq.(13) or Eq.(14).
18             Generate the sequence of nodes $N^v_{pre}$ by
                 Eq.(6) and Eq.(10).
19             Calculate loss $O$ by Eq.(15) and Eq.(16).
20         end
21     end
22     $t$ += 1
23 end
```

the assumption of similar number $|V|$ of nodes in two networks, the total time complexity of the core calculation process of this algorithm is $O(T \cdot d \cdot len \cdot |\mathcal{S}_{anchor}| \cdot |V|)$, which $|\mathcal{S}_{anchor}|$ denotes the number of node pairs in the training data set, T denotes the number of iterations. GANN uses inductive calculation mode during the alignment process, and its complexity mainly depends on the size of the network nodes and the size of the training set. Detailed experimental results can refer to the comparative experiments in various datasets in the supplementary materials. These results demonstrate that GANN has achieved excellent performance in various large-scale datasets with the fastest runtime.

V. Experiment and Analysis

A. Experiment Setup

In this section, different real datasets are used to verify GANN and other latest comparison methods. These experiments are all conducted on a computer with Intel CoreTM CPU i5-3470, 3.20 GHz, and 12.0 GB RAM. We utilize uniform parameter settings for training. The length of the walk: $walk\_length$=5,

TABLE I
STATISTICS OF MULTI-LAYER NETWORK ALIGNMENT DATA SET

Dataset	Nodes	Edges information	Attributes	Alignment
CKM	246/246/246	Advice:480 Discussion:565 Friend:506	True	Full
Douban	1118/3906	Online:1512 Offline:8164	True	Partial
Tw-Fa	17359/20024	Twitter:114999 Facebook:112381	False	Partial
Tw-Fo	20417/17355	Twitter:236882 Foursquare:132208	False	Partial

the embedding dimension of the vector: d=100, the number of negative sampled: $|n_e|$=5, the length of the alignment sequence output of GANN: s_{len}=10, the learning rate: lr=0.001, the weight decay parameter: $weight\_deca$=0.0001.

1) Dataset: We use different types, different attributes, and different scale data sets to validate GANN. The statistical information of these data sets is shown in Table I. In this paper, four real-world data sets are used in this section. The specific information is as follows:

CKM Dataset: is multiple networks of physicians on medical innovation collected by Coleman, Katz, and Menzel, which considers physicians in four towns in Illinois: Peoria, Bloomington, Quincy, and Gallesberg. This dataset is used to study the impact of network connections on the adoption of the new drug tetracycline by these doctors.

Douban Dataset: contains two networks collected from the Douban social network. One network is an online social network, which shows who is following whom on Douban website; the other network is an offline activities network, which uses the co-appearance of users in social activities, and people make physical contact in social activities. The online network has 1118 users, and the offline network has 3906 users, including all online users and their locations for constructing node attributes.

Twitter and Facebook Dataset (Tw-Fa): comes from Facebook and Twitter accounts in Singapore. The sub-network of Facebook has 17,359 users, and the sub-network of Twitter has 20024 users. Part of the authenticity of the 1998 user identity pair was determined by the user's short biographical description on their Twitter account, which stated their Facebook account. No node function is provided for this dataset.

Twitter and Foursquare Dataset (Tw-Fo): is also collected from the Foursquare and Twitter accounts of users in Singapore. The sub-network of Twitter contains 20,417 users, and the sub-network of Foursquare contains 17,355 users. There are 3602 pairs of user identities for these two sub-networks.

According to the different characteristics of these datasets, they are mainly divided into three comparison groups to evaluate the performance of GANN in different aspects: (1) Multiple Network Dataset. The performance of the model in the multiple networks alignment task is verified on the CKM data set, and the alignment performance between a pair of networks is verified on other data sets. (2) Attribute Network Datasets. CKM and Douban are used to verify the performance of the information fusion of node attributes and network structure. Also, we also verify the sensitivity of our model to node attribute information in these networks. (3) Large Datasets Without Attribute. Our model is verified that the alignment performance only depends on structure information for large-scale Tw-Fa and Tw-Fo datasets without node attributes.

This paper compares different network alignment algorithms based on representation learning to verify the effectiveness of GANN. In order to compare the performance of these models and algorithms on training sets of different proportions and the dependence on the amount of label data, all data sets are divided into different proportions for verification in this paper. The specific division method is: take known pairs of aligned anchor links according to different ratios (20%, 50%, 80%) as the training set, and the remaining nodes with known anchor links as the test set. For example, in the Douban dataset, 80% of the nodes are selected as the training set, and the remaining 20% of the nodes are used as the test set to verify the performance of six methods. In the process of testing, GANN uses a 2-layer graph convolutional neural network and a sequence generation module based on LSTM. The generated sequence length is 10, and the pooling operation is average pooling. Since the REGAL algorithm is unsupervised data, only the test set is used to verify the algorithm. In order to compare the fairness of the algorithm, other experimental parameters are that the embedding dimension of a node is 100 dimensions.

B. Comparison Algorithms

This paper compares GANN with PALE, DeepLink, IONE, REGAL and GAlign. The specific introduction is as follows:

PALE [14]: maximizes the log-likelihood and potential space matching of the observed edge and predicts anchor links by network embedding.

DeepLink [34]: is an end-to-end semi-supervised learning method. It samples the network and learns to encode network nodes as vector representations for capturing local and global network structures, which in turn can be used to align anchor nodes through deep neural networks.

IONE [13]: adds known and potential anchor users to the network to facilitate the transmission of context information in different networks. Under the unified optimization framework, the network embedding problem and the user alignment problem are solved at the same time. Stochastic gradient descent and negative sampling algorithms are used to solve the scalable problem.

REGAL [10]: uses the ability of automatically learned node representation to match nodes in different networks. In REGAL, they designed xNetMF, which is an elegant and principled node embedding formula, which is uniquely extended to multiple network alignment problems.

GAlign [41]: is a completely unsupervised network alignment framework based on a multiple-level embedding model. The model uses graph convolutional neural networks to represent the embedding of each node to prove that it satisfies the consistency constraint and further designs a data expansion method and refinement mechanism to make the model adapt to consistency conflict and noise.

We compare GANN with PALE, DeepLink, IONE, REGAL and GAlign to verify the performance of our model. PALE, IONE and Galing have a strong dependence on the assumption of structural consistency. DeepLink and REGAL algorithms use greedy algorithm and KD tree for similarity calculation in the node matching stage to verify the performance of this model in solving fuzzy selection problems. In addition, REGAL can directly conduct multiple network alignment task. Therefore, we use REGAL and GANN to compare the performance of multiple network alignment.

C. Evaluation Metric

To evaluate the performance of GANN and other comparison algorithms, we use three evaluation indexes: Accuracy, Top-k accuracy (Precision@k), and Mean Reciprocal Rank (MRR), in which Accuracy is used to measure the hard alignment results, and Precision@k and MRR are used to measure the soft alignment results.

Accuracy: It is one of the most critical aspects of the hard alignment manner. For the obtained similarity matrix S, the heuristic greedy matching algorithm is used as a post-processing step for the matrix to obtain one-to-one alignment accuracy. The heuristic algorithm iteratively finds the highest score $s_{i,j}$ on the alignment matrix and records the node pair (i, j), and then deletes all the scores involving the node i or the node from the matrix (is replaced by zero); the process stops when all nodes in one of the graphs are paired.

$$Accuracy = \frac{P_{positive}}{G_{anchor}}. \tag{17}$$

which $P_{positive}$ is correctly identified node pairs, G_{anchor} is ground-truth node pairs. In this paper, we use the reciprocal of the position number of the element in the obtained sequence divided by the length of the sequence as the similarity obtained by the positioning node. Therefore, the similarity matrix S is constructed based on this method.

In addition to accuracy, we verify the performance of our GANN by a soft network alignment task. The detailed flow of the experimental design is: (1) in the ground truth construction phase: we construct the ground truth node sequence of each node belonging to the target network by BFS, where the initial node is guaranteed to be in the first place. We treat the result sequence as the ground truth node sequences. (2) In the prediction phase: for a source node in the source network, we treat the generated node sequences with length K as Top-K nodes that should be most aligned with the target nodes, and form the predicted node sequences in decreasing order of the probability of alignment.

Precision@k: The metric measures whether the positive matching will occur in top-k candidates corresponding to k nodes having the highest similarity with the source node. Generally, Precision@k is defined as:

$$\text{Precision@k} = \frac{P^k_{candiates}}{G_{anchor}}, \tag{18}$$

which $P^k_{candiates}$ denotes is the times that the target node in top k similarity candidates nodes, G_{anchor} denotes the length of ground-truth node sequences.

Mean Reciprocal Rank (MRR): Some applications may need to generate a ranking list of potential matches to rank correct matches as high as possible. The average Reciprocal Rank is used as a soft evaluation index to study whether the alignment technology can produce meaningful results. The calculation formula of MRR is as follows:

$$\text{MRR} = \frac{1}{|Q|}\sum_{i=1}^{Q}\left(\frac{1}{ra}\right), \tag{19}$$

which Q is the sample query set, $|Q|$ is the number of samples in set Q, ra is the index of the correct matching node in the candidate sequence. In this paper, we use the node sequence obtained from the pre-processing stage of GANN as the ground-truth node ranking.

According to the evaluation metrics Precision@k and Mean Reciprocal Rank (MRR), the correlation and similarity between the predicted node sequences and the ground truth node sequences are measured to indicate the performance of the node sequences generated.

D. Performance Analysis Experiment

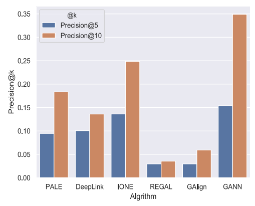
The experimental results of all algorithms are shown in the Table II and Fig. 4. In training datasets of different division proportions, GANN generally shows superior performance. Specially, GANN can achieve more than 10% of the latest GAlign in smaller-scale data sets; on the larger-scale data sets with an imbalance in the number of nodes between networks, GANN also shows better adaptability and competitive performance than other comparison methods.

According to the Table II, it can be seen that the performance of the result of MRR and Accuracy of different algorithms in training data sets with different division proportions is quite different. For the network alignment task of three networks in CKM dataset, our model is better than other algorithms on the whole. For the network alignment task of a pair of networks, Deeplink shows better performance in data sets with a small scale but has poor experimental results in a large-scale and unbalanced data set. The scale imbalance between the source network and the target network can adversely affect the learning process of the mapping function. PALE and IONE have an over-dependence on the data of the training datasets. As the proportion of the training dataset increases, the performance of these methods also shows more excellent performance. GAlign is one of the most eye-catching network alignment algorithms

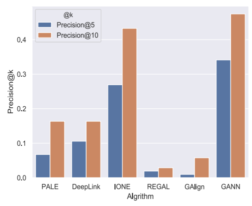
TABLE II
EXPERIMENTAL RESULTS OF ALL ALGORITHMS ON 4 DATASETS

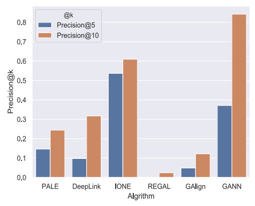
Training	Metric	Dataset	PALE	DeepLink	IONE	REGAL	GAlign	GANN
20%	MRR/Accuracy	CKM	0.0736/0.0414	0.0816/**0.0592**	0.0943/0.0118	0.0255/0.0059	0.0308/0.0059	**0.1395**/0.0533
		Douban	0.2741/0.2741	0.1408/0.0670	0.2467/0.1061	0.0768/0.0078	**0.5395**/0.2514	0.5195/**0.3528**
		Tw-Fa	0.0198/0.0119	0.0069/0.0044	0.0258/0.0162	0.0021/0.0010	0.0010/0	**0.0318/0.0188**
		Tw-Fo	0.0978/0.0902	0.0146/0.0076	0.0627/0.0440	0.0033/0	0.0010/0	**0.1037/0.0922**
50%	MRR/Accuracy	CKM	0.0660/0.0577	0.0878/0.0385	0.1535/0.0096	0.0244/0	0.0234/0	**0.3095/0.11825**
		Douban	0.4797/0.4150	0.2213/0.1753	0.4186/0.0751	0.0637/0.0018	0.5453/0.2504	**0.5589/0.3665**
		Tw-Fa	0.0484/0.0370	0.0127/0.0090	0.0436/0.0420	0.0021/0.0010	0.0012/0	**0.0626/0.0517**
		Tw-Fo	0.165/0.1677	0.0339/0.0272	0.147/0.0650	0.0033/0	0.0009/0	**0.1792/0.1609**
80%	MRR/Accuracy	CKM	0.0893/0.0488	0.0858/0	0.2383/0.0011	0.0082/0.0153	0.0389/0	**0.336/0.1285**
		Douban	0.5298/0.4554	0.2463/0.2054	0.6107/**0.6070**	0.0766/0.0045	0.5083/0.2679	**0.6134**/0.4309
		Tw-Fa	0.0899/**0.0532**	0.013/0.0075	0.0821/0.0326	0.0021/0.0010	0.0010/0	**0.0912**/0.0521
		Tw-Fo	0.1911/**0.1942**	0.0573/0.0583	0.2445/0.0457	0.0033/0	0.0008/0	**0.2508**/0.1903

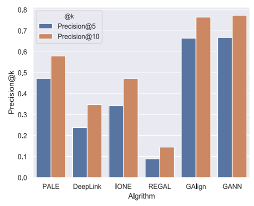
The bold entities highlight the results of the proposed method.

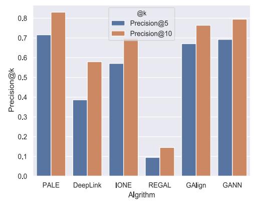


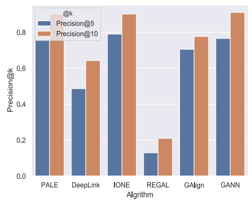
(a) 20% training set of CKM

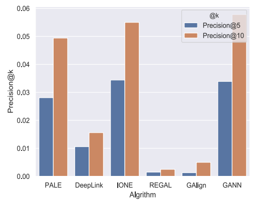

(b) 50% training set of CKM

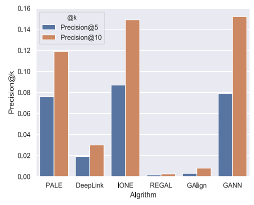

(c) 80% training set of CKM

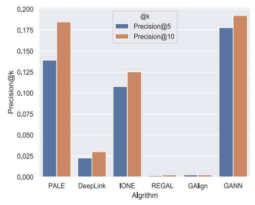

(d) 20% training set of Douban

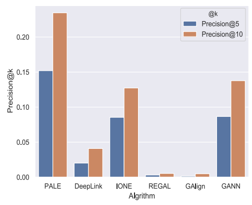

(e) 50% training set of Douban

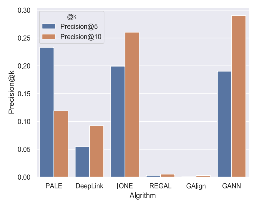

(f) 80% training set of Douban


(g) 20% training set of Tw-Fa


(h) 50% training set of Tw-Fa


(i) 80% training set of Tw-Fa


(j) 20% training set of Tw-Fo


(k) 50% training set of Tw-Fo

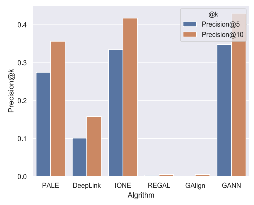

(l) 80% training set of Tw-Fo

Fig. 4. Experiment result of all comparison algorithm about precision.

and shows significantly different performances in these four datasets. Specifically, the reason why GAlign exhibits poor performance is that the algorithm uses different levels of information of the GNNs to align two networks (Tw-Fa and Tw-Fo) without node attributes. The different level information is highly sensitive to the quality of node attributes and the different network scales. Moreover, GAlign overly depends on the assumption of structural consistency. This is one of the main reasons why GAlign shows different performance in these datasets. IONE and GANN can show competitive performance on each dataset. The main reason is that the two methods are relatively similar in the embedding stage so that adjacent nodes can be embedded in the same local space. However, it is different in the matching stage. GAlign, PALE, DeepLink, and IONE dependent on the proportion of training set to learn the mapping function. Similar nodes are gathered in the same local space, so it is difficult to distinguish different nodes in the matching stage. The hierarchical structure information obtained by GANN effectively alleviates this problem.

In order to verify the performance of all algorithms in soft alignment tasks, we utilize Precision@5 and Precision@10 indicators to verify. These experimental results are shown in Fig. 4. As can be seen from the figure, we can see that GANN shows

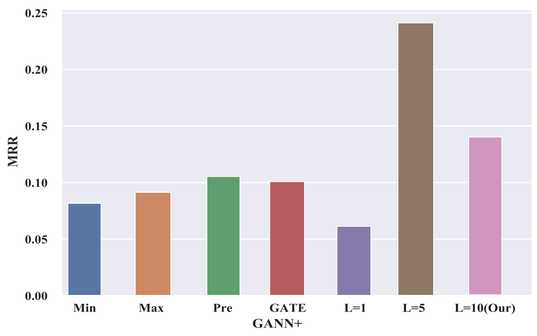


Fig. 5. Multiple parameter analysis experiment of GANN.

excellent performance in general. For the soft alignment task, GANN, PALE, and IONE show stable performance in different training sets and test sets. GANN is based on the node sequence generation model, and the breadth-first search algorithm is used to construct the labeled dataset so that the learned sequence is distinguished in the local structure. Unlike PALE and IONE, GANN can embed the learned nodes with hierarchical information. This hierarchical information enables the correct node to be placed in the first place of the sequence, and the subsequent nodes in the sequence are the higher-order neighbors of the correct node in turn. At the same time, in addition to the impact on performance due to attribute dependence and the hypothesis of structural consistency, DeepLink, and Galign cannot guarantee that a node set in the candidate sequence includes the correct nodes and its surrounding nodes. It can introduce noise to affect the quality of network alignment.

1) Parametric Analysis Experiment: We analyze the parameter sensitivity of GANN on the 20% training set, which only uses of “Advice” network and “Discussion” network of CKM dataset. The specific analysis is shown in Fig. 5. This experiment verifies the influence of three parameters, including pooling calculation on the result, the effect of training model on the result and the effect of training node sequence length on the experiment. For verifying the effect of pooling calculation on the results, we use the end-to-end network alignment model based on LSTM and the generation sequence of length 10 to train compared models. For verifying the effect of training mode calculations on the results, we use an end-to-end network alignment model based on average pooling, and the generation sequence of length 10 to train compared models. For verifying the effect of the length of the generation sequence on the result, we use an end-to-end network alignment model based on average pooling and LSTM. The detailed analysis results are as follows:

(1) Under the same other parameters, “Min” represents the minimum pooling operation (the results are shown in the first column), “Max” represents the maximum pooling operation (the results are shown in the second column), and the average pooling operation (the results are shown in the seventh column). These results are used to verify the effect of the hidden state on the entire model in the sequence decoder module of GANN. Through the comparison of the experimental results, it can be seen that the average pooling operation of GANN has superior performance, and the average pooling operation can capture more general graph structure information. The “Max” and “Min” operations largely depend on the node embedding and the initial feature of the node. For example, suppose the attributes of nodes in the source network have the same attribute values or abnormal values. In that case, it can have an obvious impact on the pooling quality of “Max” and “Min” operations.

(2) Under the same other parameters, “Pre” indicates that the node embedding stage is a pre-training process (the result is shown in the third column), and the embedding process and the sequence decoder module under the same parameters are an end-to-end mode (the result is shown in the seventh column). These results are used to verify whether GANN is sensitive to the embedding method. By comparing the experimental results, we can know that GANN based on an end-to-end embedding strategy, can effectively capture the node structure and attribute information that is more relevant to the alignment task.

(3) Under the same other parameters, “GATE” means that the gated neural network (GRU) is used in the sequence decoder module (the result is shown in the fourth column) to replace LSTM used in this paper (the result is shown in the seventh column), which are used to verify whether GANN has a strong dependency on the sequence generation method. Comparing the experimental results shows that GANN based on LSTM is better than GANN based on GATE. LSTM has a more complex structure and more parameters. Therefore it can more accurately fit the mapping function in the essentially complex network alignment task.

(4) Under the same other parameters, in the sequence decoder module, ”L=1” means that the length of the generated sequence is 1 (the result is shown in the fifth column), that is, GANN degenerates into a similar IONE and PALE. ”L=5” means that the length of the generated sequence is 5, and ”L=10” means that the length of the generated sequence is 10, which is used to verify the sensitivity of GANN to the length parameters of the generated sequence. Through the comparison of the experimental results, it can be seen that when the sequence length is 1, the performance of GANN is significantly reduced. However, with the increase of the parameter “L”, it shows a trend of first increasing and then decreasing. We believe that the longer generation sequence cannot accurately capture the hierarchical information in the node embedding. When the sequence of the generated sequence is very long, more noise information will be introduced. It can seriously affect the efficiency of GANN model. In addition, it should be noted that although GANN can achieve the best results in “L=5”, but we set it to 10 in the experimental setup stage. The main reason is that the dataset used in this experiment is relatively small. In performance experiments, we set the generation sequence to 10 because our model can perform better performance in most data sets.

E. Ablation Experiment

In this experiment, we aim to use different embedding methods to verify the effectiveness of the node embedding module of GANN, that is, the effectiveness of the proposed embedding method and the effectiveness of the embedding space alignment

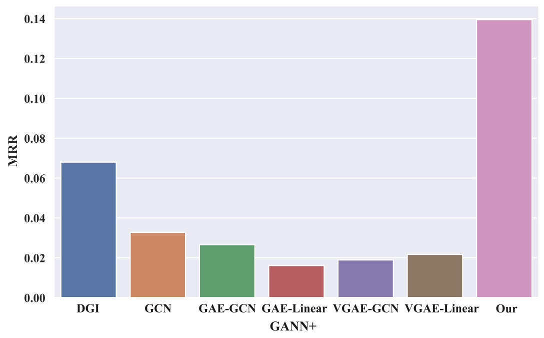


Fig. 6. Ablation analysis experiment of GANN.

method. In this experiment, Deep Graph Infomax (DGI) [51] is currently one of the latest and best methods. It is generally agreed in many related types of research that DGI outperforms GCN and Our embedding methods. About GANN based on DGI, we do not realize the embedding space reconciliation process of different networks. From the comparison of the results in the first column and the seventh column in Fig. 6, it can be known that although we use a better node embedding method, the performance of alignment result cannot outperform our model due to the lack of embedding space reconciliation process. About GANN based on GCN (the specific experimental results are shown in the second column), although the node embedding performance is similar to our embedding method, the breadth-first search (BFS) method is used in our model. Due to the introduction of BFS algorithm, the embedding of nodes can be updated by the loss function in the sequence decoder module to preserve the hierarchical information of a network. Considering the embedding space reconciliation problem, we compare GANN with a series of algorithms of the graph autoencoder (the specific experimental results are shown in the third to sixth columns) such as GAE-GCN, GAE-Linear, VGAE-GCN, VGAE-Linear. This comparative experiment verifies that the embedding quality of nodes also has an obvious impact on the alignment performance of GANN.

VI. Conclusion

This paper mainly addresses the problem of network alignment and realizes the detection of the same node in different networks to build an essential bridge between multiple networks to avoid the information silo problem of network data. For alleviating the confusing selection problem and the over-dependence on structural consistency assumptions, we propose a graph alignment neural network model for node sequence generation with graph-to-sequence learning (GANN). This model first utilizes breadth-first search to generate node sequences in preprocessing stage. Based on these node sequences, we design node representation learning with embedding constraint so that the known aligned nodes have similar representation. In order to lock the local structure where the correct node is located, we use the K-means algorithm to cluster the embedding of nodes. Finally, we propose a sequence decoder module based on a graph to sequence learning to align multiple networks. The method utilizes attention-based LSTM to learn the embedding sequence corresponding to the node sequence generated in the preprocessing stage. We can train the node sequence generation method to align networks. The experimental results show that the GANN model proposed in this paper can show excellent performance on a different scale, different attribute states, and different proportions of training sets.

References

[1] Y. Li, J. Ren, J. Liu, and Y. Chang, "Deep sparse autoencoder prediction model based on adversarial learning for cross-domain recommendations," *Knowl.-Based Syst.*, vol. 220, 2021, Art. no. 106948.

[2] R. Singh, J. Xu, and B. Berger, "Global alignment of multiple protein interaction networks with application to functional orthology detection," in *Proc. Nat. Acad. Sci.*, vol. 105, no. 35, pp. 12763–12768, 2008.

[3] S. Pei, L. Yu, G. Yu, and X. Zhang, "REA: Robust cross-lingual entity alignment between knowledge graphs," in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2020, pp. 2175–2184.

[4] H.L. Nguyen, D. ThinhVu, and J. J. Jung, "Knowledge graph fusion for smart systems: A survey," *Inf. Fusion*, vol. 61, pp. 56–70, 2020.

[5] B. J. West et al., "Relating size and functionality in human social networks through complexity," in *Proc. Nat. Acad. Sci.*, vol. 117, no. 31, pp. 18355–18358, 2020.

[6] X. Zhou, X. Liang, X. Du, and J. Zhao, "Structure based user identification across social networks," *IEEE Trans. Knowl. Data Eng.*, vol. 30, no. 6, pp. 1178–1191, Jun. 2018.

[7] 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.*, San Francisco, 2013, pp. 179–188.

[8] J. Feng et al., "User identity linkage via co-attentional neural network from heterogeneous mobility data," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 2, pp. 954–968, Feb. 2022.

[9] H. Nassar, N. Veldt, S. Mohammadi, A. Grama, and D. F. Gleich, "Low rank spectral network alignment," in *Proc. World Wide Web Conf.*, Lyon, 2018, pp. 619–628.

[10] M. Heimann, H. Shen, T. Safavi, and D. Koutra, "REGAL: Representation learning-based graph alignment," in *Proc. 27th ACM Int. Conf. Inf. Knowl. Manage.*, Torino, 2018, pp. 117–126.

[11] R. Tang, S. Jiang, X. Chen, H. Wang, W. Wang, and W. Wang, "Interlayer link prediction in multiplex social networks: An iterative degree penalty algorithm," *Knowl.-Based Syst.*, vol. 194, pp. 105598(1)–105612(14), 2020.

[12] S. Zhang, H. Tong, R. Maciejewski, and T. Eliassi-Rad, "Multilevel network alignment," in *Proc. World Wide Web Conf.*, San Francisco, 2019, pp. 2344–2354.

[13] 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.*, New York, 2016, pp. 1774–1780.

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

[15] 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. 3, pp. 1080–1095, Mar. 2020.

[16] X. Chen, M. Heimann, F. Vahedian, and D. Koutra, "Cone-align: Consistent network alignment with proximity-preserving node embedding," in *Proc. 29th ACM Int. Conf. Inf. Knowl. Manage.*, 2020, pp. 1985–1988.

[17] S. Zhang and H. Tong, "Attributed network alignment: Problem definitions and fast solutions," *IEEE Trans. Knowl. Data Eng.*, vol. 31, no. 9, pp. 1680–1692, Sep. 2019.

[18] K. K. Qin, F. D. Salim, Y. Ren, W. Shao, M. Heimann, and D. Koutra, "G-CREWE: Graph compression with embedding for network alignment," in *Proc. 29th ACM Int. Conf. Inf. Knowl. Manage.*, 2020, pp. 1255–1264.

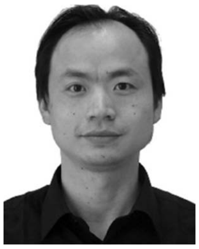
[19] H. Chen, H. Yin, X. Sun, T. Chen, B. Gabrys, and K. Musial, "Multi-level graph convolutional networks for cross-platform anchor link prediction," in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, Virtual Event, 2020, pp. 1503–1511.

[20] 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.

[21] F. Ren, Z.J. Zhang, S. Z. Su, and C. Guo, "Banana: When behavior analysis meets social network alignment," in *Proc. 29th Int. Joint Conf. Artif. Intell. 17th Pacific Rim Int. Conf. Artif. Intell.*, Yokohama, 2020, pp. 1438–1444.

[22] H. Hong, X. Li, Y. Pan, and I. Tsang, "Domain-adversarial network alignment," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 7, pp. 3211–3224, Jul. 2022.
[23] H.T. Trung et al., "Adaptive network alignment with unsupervised and multi-order convolutional networks," in *Proc. IEEE 36th Int. Conf. Data Eng.*, Dallas, 2020, pp. 85–96.
[24] K. Xu, L. Wu, Z. Wang, Y. Feng, M. Witbrock, and V. Sheinin, "Graph2Seq: Graph to sequence learning with attention-based neural networks," 2018, *arXiv: 1804.00823*.
[25] D. Beck, G. Haffari, and T. Cohn, "Graph-to-sequence learning using gated graph neural networks," in *Proc. 56th Annu. Meeting Assoc. Comput. Linguistics*, Melbourne, Australia, 2018, pp. 273–283.
[26] L. Song, Y. Zhang, Z. Wang, and D. Gildea, "A graph-to-sequence model for AMR-to-text generation," in *Proc. 56th Annu. Meeting Assoc. Comput. Linguistics*, Melbourne, Australia, 2018, pp. 1616–1626.
[27] Z. Guo, Y. Zhang, Z. Teng, and W. Lu, "Densely connected graph convolutional networks for graph-to-sequence learning," *Trans. Assoc. Comput. Linguistics*, vol. 7, pp. 297–312, 2019.
[28] S. Gu, J. Johnson, F. E. Faisal, and T. Milenković, "From homogeneous to heterogeneous network alignment via colored graphlets," *Sci. Rep.*, vol. 8, no. 1, pp. 1–16, 2018.
[29] Q. Wang, D. Shen, S. Feng, Y. Kou, T.-Z. Nie, and G. Yu, "Identifying users across social networks based on global view features with crowdsourcing," *J. Softw.*, vol. 29, no. 3, pp. 811–823, 2018.
[30] Y. Liu, H. Ding, D. Chen, and J. Xu, "Novel geometric approach for global alignment of PPI networks," in *Proc. AAAI Conf. Artif. Intell.*, San Francisco, 2017, pp. 31–37.
[31] V. Vijayan and T. Milenković, "Multiple network alignment via multimagna," *IEEE/ACM Trans. Comput. Biol. Bioinf.*, vol. 15, no. 5, pp. 1669–1682, Sep./Oct. 2018.
[32] G. Gomes, V. Rao, and J. Neville, "Community detection over a heterogeneous population of non-aligned networks. 2019, *arXiv:1904.05332*.
[33] W. Vincent et al., "Heterogeneous embedding propagation for large-scale e-commerce user alignment," in *Proc. IEEE Int. Conf. Data Mining*, Singapore, 2018, pp. 1434–1439.
[34] 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.*, Honolulu, 2018, pp. 1313–1321.
[35] F. Zhou, C. Cao, G. Trajcevski, K. Zhang, T. Zhong, and J. Geng, "Fast network alignment via graph meta-learning," in *Proc. IEEE Conf. Comput. Commun.*, Canada, 2020, pp. 686–695.
[36] X. Chu, X. Fan, D. Yao, Z. Zhu, J. Huang, and J. Bi, "Cross-network embedding for multi-network alignment," in *Proc. World Wide Web Conf.*, San Francisco, 2019, pp. 273–284.
[37] 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.*, Canada, 2014, pp. 159–165.
[38] C. Kong, M. Gao, C. Xu, Y. Fu, W. Qian, and A. Zhou, "EnAli: Entity alignment across multiple heterogeneous data sources," *Front. Comput. Sci.*, vol. 13, no. 1, pp. 157–169, 2019.
[39] Z. Sun et al., "Knowledge graph alignment network with gated multi-hop neighborhood aggregation," in *Proc. 34th AAAI Conf. Artif. Intell., 32nd Innov. Appl. Artif. Intell. Conf., 10th AAAI Symp. Educ. Adv. Artif. Intell.*, 2020, pp. 222–229.
[40] R. Ye, X. Li, Y. Fang, H. Zang, and M. Wang, "A. vectorized relational graph convolutional network for multi-relational network alignment," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, Macao, 2019, pp. 4135–4141.
[41] T. T. Huynh, V. V. Tong, T. T. Nguyen, H. Yin, M. Weidlich, and N. Q. V. Hung, "Adaptive network alignment with unsupervised and multi-order convolutional networks," in *Proc. 36th IEEE Int. Conf. Data Eng.*, Dallas, Dallas, 2020, pp. 85–96.
[42] H. Gao, Y. Wang, S. Lyu, H. Shen, and X. Cheng, "Gcn-alp: Addressing matching collisions in anchor link prediction," in *Proc. IEEE Int. Conf. Knowl. Graph*, Nanjing, 2020, pp. 412–419.
[43] Q. Zhou, L. Li, X. Wu, N. Cao, L. Ying, and H. Tong, "Attent: Active attributed network alignment," in *Proc. World Wide Web Conf.*, Ljubljana, 2021, pp. 3896–3906.
[44] X. Li, Y. Shang, Y. Cao, Y. Li, J. Tan, and Y. Liu, "Type-aware anchor link prediction across heterogeneous networks based on graph attention network," in *Proc. AAAI Conf. Artif. Intell., 34th AAAI Conf. Artif. Intell., 32nd Innov. Appl. Artif. Intell. Conf., 10th AAAI Symp. Educ. Adv. Artif. Intell.*, 2020, pp. 147–155.
[45] T.T. Nguyen et al., "Structural representation learning for network alignment with self-supervised anchor links," *Expert Syst. Appl.*, vol. 165, pp. 113857–113871, 2021.
[46] K. Shu, S. Wang, J. Tang, R. Zafarani, and H. Liu, "User identity linkage across online social networks: A review," *ACM SIGKDD Explorations Newslett.*, vol. 18, no. 2, pp. 5–17, 2017.
[47] X. P. Zhou, X. Liang, J. C. Zhao, Z. Y. Li, and Y. F. Ma, "Correlating user mining methods for social network integration: A survey," *J. Softw.*, vol. 28, no. 6, pp. 1565–1583, 2017.
[48] H. T. Trung et al., "A comparative study on network alignment techniques," *Expert Syst. Appl.*, vol. 140, no. 31, pp. 1823–1829, 2020.
[49] J. Li, D. Ye, and S. Shang, "Adversarial transfer for named entity boundary detection with pointer networks," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, Macao, China, 2019, pp. 5053–5059.
[50] R. Wang, J. Yan, and X. Yang, "Learning combinatorial embedding networks for deep graph matching," in *Proc. IEEE Int. Conf. Comput. Vis.*, Seoul, 2019, pp. 3056–3065.
[51] P. Velickovic, W. Fedus, W. L. Hamilton, P. Liò, Y. Bengio, and R. D. Hjelm, "Deep graph infomax," in *Proc. 7th Int. Conf. Learn. Representations*, 2019.

Nianwen Ning received the PhD degree in computer science and technology from the Beijing University of Posts and Communications, in 2021. He is currently a lecturer in the school of Artificial Intelligence at Henan University. His research interests include graph data mining and intelligent transportation.



Bin Wu received the PhD degree from the ICT of Chinese Academic of Sciences, in 2002. He is currently a professor in Beijing University of Posts and Telecommunications. He has published more than 100 papers in refereed journals and conferences. His current research interests include social computing, data mining, and complex network.

Haoqing Ren received the master's degree from the Beijing University of Posts and Communications, in 2023. He is currently working at the Agricultural Bank of China Research and Development Center located in Tianjin, since 2023. His research interests include graph data mining, social computing, data mining, and complex network.

Qiuyue Li received the master's degree from Beijing University of Posts and Communications, in 2022. She is currently working at China's Alibaba Group, since 2022. Her research interests include graph data mining and distributed computing.