Alleviate the Impact of Heterogeneity in Network Alignment From Community View

Qiyao Peng, Yinghui Wang, Pengfei Jiao, *Member, IEEE*, Huaming Wu, *Senior Member, IEEE*, and Lin Pan

***Abstract*— Network alignment is a fundamental problem in various domains since it can establish bridges for the same entity (i.e., anchor nodes) between different networks. Most existing network alignment methods are based on consistency assumption, i.e., anchor nodes exhibit similar local structures or neighbors across different networks. However, many anchor nodes have different local structures or neighbors across different networks, which could be regarded as anchor nodes' heterogeneity. It poses a challenge to methods based on the assumption of consistency, as they lack abundant shared information, such as common neighbors. Fortunately, network communities provide the comprehension of node relationships and group structures within networks, which could alleviate the information insufficient. In this article, we propose to address the challenge of inadequate shared information triggered by nodes' heterogeneity from a community perspective. Our model is based on joint optimization of node representation learning and community discovery, including: 1) a node-level constraint is employed to bring nodes with more anchor pairs as neighbors closer together and 2) a community-level constraint is utilized to bring nodes with higher order similarity closer together. We model the cross-network community alignment relations as asymmetric to mitigate the interference caused by anchor node heterogeneity when measuring community alignment relations. Furthermore, we leverage the learned cross-network community alignment relations to supplement node alignment, which could narrow down the search range of potential anchor nodes by focusing solely on aligning nodes within aligned cross-network communities. We conducted extensive experiments on real-world datasets, and the results show the effectiveness and efficiency of our proposed model on network alignment.**

***Index Terms*— Anchor node heterogeneity, cross-network community alignment relations, network alignment, node representation learning.**

Received 23 October 2022; revised 1 May 2023, 9 January 2024, and 6 August 2024; accepted 26 October 2024. This work was supported in part by the National Natural Science Foundation of China under Grant 62372146 and Grant 62071327 and in part by Zhejiang Provincial Natural Science Foundation of China under Grant LDT23F01015F01. *(Corresponding author: Pengfei Jiao.)*

Qiyao Peng is with the School of New Media and Communication, Tianjin University, Tianjin 300072, China (e-mail: qypeng@tju.edu.cn).

Yinghui Wang is with the Key Laboratory of Information System and Technology, Beijing Institute of Control and Electronic Technology, Beijing 100038, China (e-mail: wangyinghui@tju.edu.cn).

Pengfei Jiao is with the School of Cyberspace and the Data Security Governance Zhejiang Engineering Research Center, Hangzhou Dianzi University, Hangzhou 310018, China (e-mail: pjiao@hdu.edu.cn).

Huaming Wu is with the Center for Applied Mathematics, Tianjin University, Tianjin 300072, China (e-mail: whming@tju.edu.cn).

Lin Pan is with the School of Marine Science and Technology, Tianjin University, Tianjin 300072, China (e-mail: linpan@tju.edu.cn).

Digital Object Identifier 10.1109/TNNLS.2024.3491892

NOMENCLATURE

Symbols	Definition
G^s, G^t	Input networks to be aligned.
E^a	Known anchor pairs set between G^s and G^t.
$\mathbf{A}^s$, $\mathbf{A}^t$	Adjacency matrices of G^s and G^t.
$\tilde{A}^s$, $\tilde{A}^t$	Symmetric normalization by degree matrix of $\mathbf{A}^s$ and $\mathbf{A}^t$.
$\mathbf{D}$	Diagonal degree matrix.
d_i	Degree of node v_i.
C^s, C^t	Set of latent communities of the G^s and G^t.
K_s, K_t	Number of communities in G^s and G^t.
$\mathbf{C}^{s\to t}$, $\mathbf{C}^{t\to s}$	Cross-network community alignment relations from G^s to G^t and from G^t to G^s.
$\mathbf{H}^s$, $\mathbf{H}^t$	Final node embedding matrices of G^s and G^t.
$\mathbf{R}^s$, $\mathbf{R}^t$	Degree to which a node is assigned to a community of nodes in G^s and nodes in G^t.
$\boldsymbol{\mu}^s$, $\boldsymbol{\mu}^t$	Communities embedding matrices of G^s and G^t.
d	Final node embedding dimension.
$\mathbf{W}^{(l)}$	Weight matrix at layer l.
$\mathbf{Y}^{s\to t}$, $\mathbf{Y}^{t\to s}$	Alignment matrix from the view of G^s and G^t.

I. INTRODUCTION

NETWORK alignment refers to the process of identifying identical entities across different networks [1], [2], [3]. For example, social network alignment aims to establish the correspondence between account nodes (referred to as anchor nodes [4]) belonging to the same user across multiple online social platforms [5]. Network alignment can integrate multisource network information, thereby enhancing the performance of downstream applications such as user behavior prediction [6] and detection of malicious entities [7], [8]. It has received increasing attention from both academic and industry.

Most network alignment methods usually rely on the consistency assumption to predict the anchor pairs across networks [9], [10], [11], [12], [13], [14], [15], [16], [17], [18], [19]. The consistency assumption is that the corresponding anchor nodes are more likely to share similar local structures in their respective networks. For example, IsoRank [9] recognizes two nodes from two networks to be similar if their neighborhoods are similar. PALE [20] preserves the

first-order proximity of nodes to generate the respective embeddings of networks and uses a multilayer perception to reduce the embedding differences of anchor pairs. GAlign [17] utilizes the embeddings from all layers of graph convolutional networks and designs a consistency loss function to enforce consistency constraints.

However, not all real-world scenarios meet the consistency assumption. In fact, anchor nodes exist heterogeneity in low-order neighborhood, i.e., corresponding anchor nodes have different local structures across different networks due to the different neighbors or behaviors of the same nodes. For example, a person may have more friends in one social network (e.g., Douban) than others (e.g., Weibo) or have totally different friends due to different social networks' functionalities. Ignoring such heterogeneity could cause anchor nodes to be misaligned and reduce alignment accuracy. As shown in Fig. 1(a), c^1 and c^2 are anchor pairs to be aligned. If only the local structure and consistency are considered, c^1 will be misaligned to b^2 because b^2 shares the anchor pair (a^1, a^2) in neighbors with c^1, and they have similar degrees compared to c^2. Moreover, as the network size increases, the number of similar candidate nodes within the network also grows, making it difficult to distinguish them.

When aligning networks, the first-order neighbors of nodes and their closely related higher order neighbors provide more effective information. Therefore, greater attention should be given to these nodes. The community structure within networks can reflect some common features of the nodes, therefore, some studies utilize the community information of nodes to address the issues caused by the heterogeneity in low-order neighborhood [21], [22], [23], [24]. Generally, these methods first perform community detection in different networks, then establish "one-to-one" and symmetric correspondences between communities in different networks, and finally align nodes at the community level. As shown in Fig. 1(b), from the community perspective, there is a high-level correlation between the communities in which c^1 and c^2 are located, and it can be inferred that members of such communities, have a high degree of relevance. Thus, c^1 is more likely to align with c^2 rather than b^2.

While previous works have yielded promising results, there remain challenges in leveraging community information for network alignment.

1) *Challenge 1:* How to effectively integrate the community information for achieving better network alignment? Existing methods often perform community detection first and align communities in different networks, then align nodes within each pair of aligned communities [22], [23]. When communities in different networks are misaligned, certain nodes within a community may not be aligned accurately. That is, misalignment of communities hinders the accurate mapping of corresponding nodes across networks.
2) *Challenge 2:* How to correctly measure alignment relationships between cross-network communities to reduce node misalignment? Previous methods usually assume the alignment relationships between cross-network communities are symmetrical and "one to one" [21], ignoring the asymmetric situation caused by the heterogeneity of anchor nodes as shown in Fig. 1(c). Inaccurate measurement can result in erroneous community alignment, thereby impacting the precision of node alignment.

To address the aforementioned challenges, we propose a novel unified framework from the Community view to alleviate the impact of Heterogeneity in Network Alignment (CHNA) in this article. For Challenge 1, unlike previous works that detect communities first and then match nodes within those communities, CHNA simultaneously performs node representation learning and community discovery, making the learned node representations and community information more conducive to network alignment. For Challenge 2, CHNA discards the assumption of symmetric and "one to one" cross-network community alignment used in existing methods. Instead, it establishes asymmetric community alignment relationships to alleviate the impact of node heterogeneity at the community level. Specifically, CHNA first uses a graph convolutional network [25] to encode node neighborhood information and performs community detection by maximizing modularity. This achieves the joint optimization of community discovery and node representation learning tasks within the network. Based on known anchor pairs and community representations, it calculates asymmetric alignment relationships between communities in different networks to capture the differences in higher order neighbors of nodes across networks. Additionally, CHNA uses distance metrics and contrastive learning to constrain the anchor nodes and community representations in different networks, ensuring that the node representations learned from both networks lie in similar embedding spaces. Finally, when aligning nodes in different networks, CHNA uses the alignment relationships between communities to restrict the candidate set for node matching. This effectively reduces the candidate set and improves the network alignment efficiency.

The main contributions of this article could be summarized as follows.

1) We propose an end-to-end framework that jointly optimizes community discovery and network alignment, which can alleviate the impact of anchor nodes' heterogeneity from the view of network community.
2) We consider the consistency and heterogeneity of anchor nodes through the use of node-level and community-level constraints, which can push nodes with more anchor pairs as neighbors closer and nodes with higher order similarity closer. Besides, introducing the community-level constraint could reduce the number of candidate nodes, which could reduce the time required for node alignment effectively.
3) We evaluate CHNA on various real-world datasets and demonstrate its effectiveness and efficiency through extensive experiments compared to existing methods.

II. Related Work

The concepts of "anchor node," "aligned networks", and the research of link prediction across aligned networks were first proposed by Kong et al. [4]. Network alignment has attracted

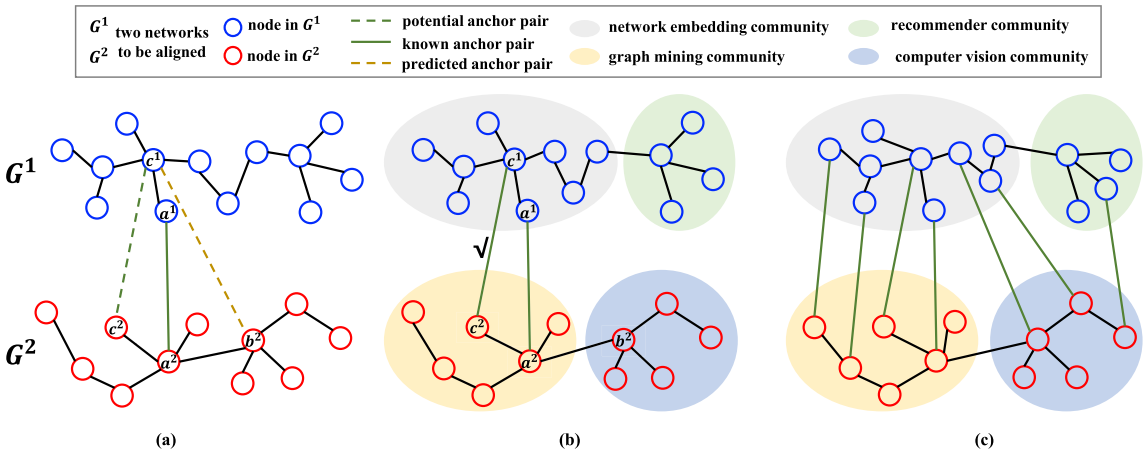

Fig. 1. (a) c^1 will be misaligned with b^2 under the consistency assumption because they have similar local structures compared to c^1 and c^2. (b) Community information helps network alignment. When c^1 and c^2 belong to communities with a high correlation, c^1 is more likely to be aligned with c^2 rather than b^2. (c) Without additional information, the alignment relations between cross-network communities are determined by their anchor node members. The number of anchor pairs between two communities indicates their level of association. Due to the heterogeneity of anchor nodes, corresponding nodes in one network can be distributed across multiple communities in the other network. As a result, the alignment relations between communities no longer follow the assumptions of "one-to-one" and symmetry when viewed from different networks.

active research in the last few years [3], [17], [26], [27], [28], [29], [30], [31], [32], [33], [34].

The significant research idea of early methods follows the consistency assumption, where anchor nodes have similar structures and/or attribute features. For example, UMA [26] optimizes by maximizing consistent relationships among nodes. MAGNA [27] is a genetic algorithm that evolves network populations to maximize topological consistency. FINAL [28] leverages the node/edge attribute information to guide the topology-based alignment process. REGAL [29] preserves structural similarities and attribute agreement by an implicit matrix factorization.

With the development of network embedding in various fields [35], [36], [37], [38], representation learning has been used to improve the effectiveness of alignment methods. Embedding-based methods preserve the structure and/or attribute information by embedding nodes into a low-dimensional feature space, then finding potential anchor nodes through the embedding-based nearest-neighbor search [3], [17], [20], [30], [39], [40], [41], [42], [43], [44], [45]. For example, CrossUGA [39] uses a shared encoder to learn cross-network node representations based on attribute and structure reconstruction. Then, it uses a discriminator to mitigate the distribution differences between different networks via adversarial training. PARROT [42] captures topology information by random walk with restart, with three carefully designed consistency regularization terms. NSVUIL [43] learns node representations as a Gaussian distribution in the Wasserstein space and designs a noise-aware self-learning module augment the annotations. DegUIL [44] complements missing neighborhoods for tail nodes and discards redundant structural information for super head nodes in embeddings, respectively, to obtain ideal neighborhoods for meaningful aggregation in GNNs. DualNA [45] designs dual graph convolutional networks as feature extractors to encode the local and global structural information, respectively, and employ attentional feature fusion to obtain holistic representations to describe users comprehensively. Although embedding-based network alignment methods no longer prioritize strict consistency optimization, the implicit assumption of consistency persists in the objective of achieving similar embeddings for anchor pairs.

When dealing with a large number of anchor nodes that exhibit heterogeneity between networks, achieving satisfactory results becomes challenging. Some methods explicitly incorporate heterogeneity constraints during the model design process. For example, DHNA [46] defines heterogeneity based on the node degrees of anchor pairs, and employs a heterogeneity constraint to ensure that nodes with significant degree differences across networks have similar embeddings. DHNA may yield satisfactory results when dealing with small network sizes. However, it is constrained by time complexity due to the requirement of calculating the node degree difference between nodes in two networks, and cannot be extended to large-scale network alignment tasks.

Therefore, some studies utilize community information to alleviate the impact of heterogeneity of anchor nodes, while using community partitioning to transform two larger scale network alignments into smaller scale network alignments between two communities [21], [22], [23], [47]. These methods most adopt a "divide-and-conquer" strategy, i.e., first match two communities in different networks, then nodes can be aligned within each pair of matched communities. For example, C-Align [21] puts community discovery, community alignment, and node alignment in one framework and uses community information as guidance to reduce computation complexity. AlignNemo [47] builds a weighted alignment graph from the input networks and then extracts all connected subgraphs of a given size from the alignment graph and ranks them according to weights on nodes and edges. CAPER [23] coarsens a network into multiple levels of varying coarseness, aligns at the coarsest level, and then projects back to finer levels while refining the solution at each level. The divide-and-conquer strategy greatly relies on the results of matching

communities in different networks, and improper community matching can significantly diminish the alignment accuracy of nodes. Therefore, some methods disregard the relationships between cross-network communities and solely focus on learning community representations that offer semantically richer information for nodes within each network. For example, NAME [24] encodes community information in the network as node representations to capture global network information. Additionally, it utilizes first-order consistency and GCN to obtain node local information and attribute information. These three representations are then used to compute similarity between nodes across networks. However, relying solely on communities as additional information will lead to increased computational complexity and may not be feasible when dealing with large-scale networks.

Compared to existing methods, the method proposed in this article uses community partitioning to align only the nodes within aligned communities in two networks, making it applicable for large-scale network alignment. It also employs joint optimization and considers the asymmetric correspondences between communities to avoid misalignment at the community level, which could effectively alleviate the impact of anchor node heterogeneity by incorporating community information.

III. Problem Statement

In this section, we provide the necessary definitions, and we summarize the main symbols and notations used in this article in Nomenclature. We use bold uppercase letters (e.g., $\mathbf{X}$) to represent matrices, bold lowercase letters (e.g., $\mathbf{x}$) to represent vectors, and unbolded letters (e.g., x) for scalars, as per convention. The operator $|\cdot|$ denotes the size of a set. We will explain each notation as it is first encountered in our article.

In this article, we focus on the case of two networks for experiments in this article. We suppose the networks are unweighted, and all the edges are undirected in this work. A network can be represented as $G = (V, E)$, where $V := \{v_i\}$ is the set of nodes and $E := \{(v_i, v_j)|v_i \in V, v_j \in V\}$ is the set of edges representing node relationships. Notation $\mathbf{A}$ represents a $|V| \times |V|$ adjacency matrix with an element a_{ij} indicating there is an edge between v_i and v_j, i.e., $\mathbf{A}_{ij} = 1$ if $(v_i, v_j) \in E$, and $\mathbf{A}_{ij} = 0$ otherwise.

As mentioned in Section I, we alleviate the impact of heterogeneity from the community view. We assume each network has some nonoverlapping latent communities, and the community partition can be denoted as $C = \{C_1, C_2, \ldots, C_K\}$, where K is the number of nonoverlapping latent communities in G. In the following, we use superscripts to denote which network the variable is associated with, and subscripts to denote the indexes of the nodes or communities, such as C_i^s represents the ith community in G^s.

A. Anchor Pairs

Given two networks $G^s = (V^s, E^s)$ and $G^t = (V^t, E^t)$, anchor pairs set can be denoted as $E^a = \{(v_i^s, v_i^t)|v_i^s \in V^s, v_i^t \in V^t\}$, where anchor nodes v_i^s and v_i^t represent the same entity across G^s and G^t.

B. Cross-Network Community Alignment Relations

For the community partitions C^s and C^t, we use the alignment relations matrix to represent the correspondence between communities in C^s and C^t. We measure the cross-community alignment relationships separately from the perspectives of the G^s and G^t networks. By considering different network viewpoints, the corresponding relationships between the same pair of communities are different, thus satisfying the asymmetric alignment relationships between cross-network communities as shown in Fig. 1(c). From the view of G^s, the cross-community alignment relations can be denoted as $\mathbf{C}^{s\to t} \in \{0, 1\}^{K_s \times K_t}$, $\mathbf{C}_{ij}^{s\to t} = 1$ means C_i^s and C_j^t are alignment communities from the view of G^s. And from the view of G^t, it can be denoted as $\mathbf{C}^{t\to s} \in \{0, 1\}^{K_t \times K_s}$.

C. Joint Network and Community Alignment

Given two partially aligned networks G^s, G^t and a few known anchor pairs E^a, the goal of joint network and community alignment is to identify all anchor pairs under the cross-network community alignment relations. That is, we aim to learn a predictive function: $f : (G^s, G^t, E^a) \to \mathbf{Y}$, where $\mathbf{Y} \in \{0, 1\}^{|V^s|\times|V^t|}$, $\mathbf{Y}(v_i^s, v_j^t) = 1$ if v_i^s and v_j^t are predicted to be a pair of anchor nodes across communities C_i^s and C_j^t, and 0 otherwise.

IV. Methodology

In this section, we describe the proposed CHNA and the mechanism that integrates community discovery and network alignment into one framework. The overview process of CHNA is shown in Fig. 2, which consists of four main parts.

1) *Node Representation Learning and Community Discovery:* We design an encoder for learning node representations and detecting communities in the network. We merge the above two tasks into a trainable end-to-end system. The encoder has two layers. The first layer is a network embedding layer that uses the adjacency matrix of the network to embed nodes as d-dimensional vectors. The second layer performs differentiable optimization. It takes the continuous-space embeddings as input and produces a soft assignment of nodes to communities and the community centers in the embedding space. We provide a detailed description of this process in Section IV-A.
2) *Cross-Network Community Alignment Relations Learning:* In this section, we utilize known anchor pairs and community embeddings to guide the cross-network community alignment relations learning. We assume that the known anchor pairs contribute more to inferring cross-network community alignment relations. And the distance between embedding representations of two communities should be smaller when they have more cross-network relations. Based on the aforementioned assumptions, we update the cross-network community alignment relations matrix $\mathbf{C}^{s\to t}$ and $\mathbf{C}^{t\to s}$ during the process of node representation learning and community discovery. We provide a detailed description of this process in Section IV-B.

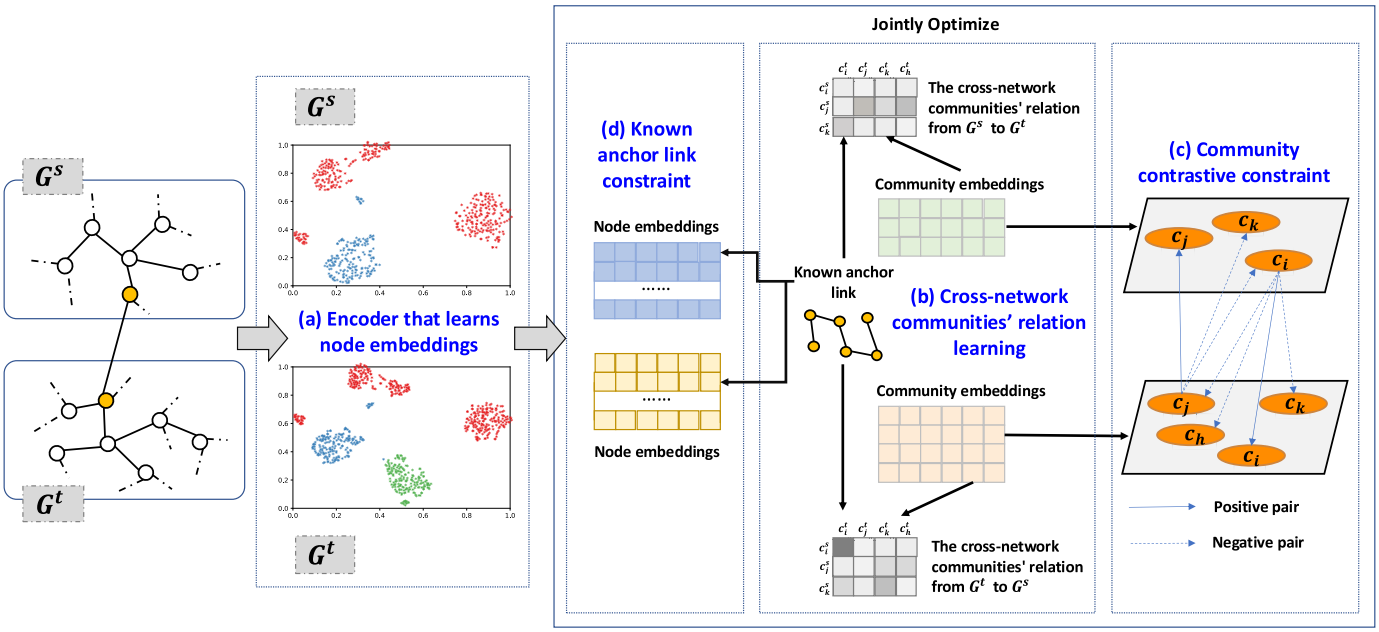

Fig. 2. Overall process of the proposed CHNA includes. (a) Encoder that learns node embeddings and detects communities in G^s and G^t. (b) Cross-network community alignment relations learning, which measures the relationships between cross-network communities. (c) Community-level contrastive constraint that pulls representations of similar communities together. (d) Node-level constraint that ensures the representations of known anchor pairs in different networks are similar.

3) *Community-Level Constraint:* As communities in different networks consist of not only anchor nodes but also numerous nonanchor nodes, the representations of aligned communities may exhibit differences. To address this, we enforce proximity in the embedding space for the community representations with alignment relationships from different networks, aiming to minimize their semantic disparities. The detailed process is described in Section IV-C.
4) *Node-Level Constraint:* During the joint optimization of node representations and community representations, there is a mutual influence between them. Consequently, the representations learned for anchor pairs in different networks may exhibit disparities, necessitating constraints on the known anchor nodes to ensure similarity in their representations across networks. Moreover, this constraint also enables nodes with a significant number of shared known anchor pairs as common neighbors in different networks to have similar representations, thereby satisfying the general consistency assumption. The detailed process is described in Section IV-C.

A. Node Representation Learning and Community Discovery

1) GCN-Based Encoder: For node representation learning, we focus on the neighbors of nodes than the whole network, because nearby nodes can provide more helpful information in predicting node relations. To better use the network's local structure while preserving the whole network structure into a low-dimension embedding space, we take a simple model parameterized by a convolutional network as an encoder. In our model, we use GCNs [25]. GCN employs a neighborhood aggregation scheme across all layers, where each convolutional layer only processes first-order neighborhood information. For each network, the propagation rules of each convolutional layer are as follows:

$$\mathbf{H}^{(l+1)} = \sigma\left(\tilde{\mathbf{A}}\mathbf{H}^{(l)}\mathbf{W}^{(l)}\right) \tag{1}$$

where $\sigma(\cdot)$ is an activation function, $\mathbf{W}^{(l)}$ is the parameter matrix of each layer, $\tilde{\mathbf{A}} = \mathbf{D}^{-(1/2)}(\mathbf{A}+\mathbf{I})\mathbf{D}^{-(1/2)}$ is the symmetrically normalized adjacency matrix and $\mathbf{D} \in \mathbb{R}^{|V|\times|V|}$ is a diagonal matrix containing each node's degree in G. $\mathbf{H}^{(l+1)}$ is the $|V| \times d$ embedding matrix. Since we only consider the alignment of networks whose nodes are without attributes in this article, we use the identity matrix $\mathbf{I}$, i.e., $\mathbf{H}^{(0)} = \mathbf{I}$, as the model input.

For community discovery, we assume that each network can be divided into K nonoverlapping communities. We use $\boldsymbol{\mu}_k$ to denote the center of community C_k and assume each node is assigned to any community with a certain probability. $\mathbf{R}_{ik}$ is used to denote the probability to which node v_i is assigned to community C_k, and $\sum_k \mathbf{R}_{ik} = 1$ for all v_i. We optimize the community centers via iterative process updates

$$\boldsymbol{\mu}_k = \frac{\sum_i \mathbf{R}_{ik}\mathbf{h}_i}{\sum_i \mathbf{R}_{ik}} \tag{2}$$

$$\mathbf{R}_{ik} = \frac{\exp\left(-\beta\left\|\mathbf{h}_i - \boldsymbol{\mu}_k\right\|\right)}{\sum_j \exp\left(-\beta\left\|\mathbf{h}_i - \boldsymbol{\mu}_j\right\|\right)} \tag{3}$$

where $\mathbf{h}_i$ is the embedding of node v_i, $k = 1, 2, \ldots, K$ and $i = 1, 2, \ldots, |V|$. We take $\mathbf{R}_{ik}$ as a soft-min assignment of each point to the community centers based on distance. The equation can be used with any norm $\|\cdot\|$, and we use the negative cosine similarity in this article because of its good performance. β is a hyperparameter that controls the effect of community division. These iterates [(2) and (3)] converge to

a fixed point where $\boldsymbol{\mu}$ remains the same between successive updates [48]. Through the output of the forward pass, we can get the community representations $\boldsymbol{\mu}$ and affiliation $\mathbf{R}$ between nodes and communities.

2) Optimizing: For community discovery, the objective is to partition the nodes of the network into K distinct communities that are internally dense but have limited edges connecting them. Formally, the objective is to find a partition maximizing the modularity [49], defined as follows:

$$Q(\mathbf{R}) = \frac{1}{2|E|} \sum_{v_i, v_j \in V} \sum_{k=1}^{K} \left[\mathbf{A}_{ij} - \frac{d_i d_j}{2|E|} \right] \mathbf{R}_{ik} \mathbf{R}_{jk} \tag{4}$$

where d_i is the degree of node v_i, and $\mathbf{R}_{ik} = 1$ if node v_i is assigned to community C_k and 0 otherwise. $\mathbf{A}$ is the adjacency matrix. $|E|$ is the total number of edges in the network. Equation (4) measures the number of edges within communities compared to the expected number if edges were placed randomly. And for each community among the K communities, there should be one partition obtained using (4). Define $\mathbf{B}$ as the modularity matrix with entries: $\mathbf{B}_{ij} = \mathbf{A}_{ij} - (d_i d_j / (2|E|))$, the training objective of a network (the expected value of a partition sampled according to $\mathbf{R}$) can be written as follows:

$$l_{\text{modularity}} = -\frac{1}{2|E|} Tr\left[\mathbf{R}^T \mathbf{B} \mathbf{R}\right]. \tag{5}$$

For networks G^s and G^t to be aligned, we let them share the last-layer parameters $\mathbf{W}^l$ of the encoder for transferring information across GCNs. Additionally, we ensure that the final learned node embedding dimension d is the same in both G^s and G^t. Through training them together, we can obtain the final node representations $\mathbf{H}^s$ and $\mathbf{H}^t$ in the approximate embedding space, community representations $\boldsymbol{\mu}^s$, $\boldsymbol{\mu}^t$, and node assignment $\mathbf{R}^s$, $\mathbf{R}^t$ for G^s and G^t, respectively.

B. Cross-Network Community Alignment Relations Learning

When no additional prior information is available, the alignment relationships between cross-network communities can be reflected by the proportion of anchor pairs in the respective communities. Specifically, by calculating the ratio of common anchor pairs between community C_i^s in network G^s and communities in network G^t to the total number of members in C_i^s, we can identify the communities in G^t that have alignment relationships with C_i^s. Therefore, we assume that the more anchor pairs two cross-network communities share, the more significant their correlation. Additionally, the more similar the embeddings of the two cross-network communities, the stronger their correlation. By combining known anchor pairs and community embeddings, we define the cross-network community correlation from network G^s to G^t as follows:

$$\mathbf{P}_{ij}^{s \to t} = \frac{n_{\left(C_i^s, C_j^t\right)}}{n_{C_i^s}} \cdot e^{\theta\left(\boldsymbol{\mu}_i^s, \boldsymbol{\mu}_j^t\right)} \cdot e^{\mathbf{C}_{ij}^{s \to t}} / \xi \tag{6}$$

where $n_{(C_i^s, C_j^t)}$ is the number of known anchor pairs between the community C_i^s and the community C_j^t, $n_{C_i^s}$ is the number of anchor nodes in C_i of G^s, $\theta(\cdot, \cdot)$ is the cosine similarity. Let $\mathbf{C}^{s \to t}$ be the cross-network community alignment relations matrix from the G^s view, and we set initial values to 1 at the beginning. ξ is a positive integer hyperparameter. We define the $\mathbf{P}_{ij}^{t \to s}$ in the same way.

Then, we update $\mathbf{C}^{s \to t}$ and $\mathbf{C}^{t \to s}$ under the guidance of $\mathbf{P}^{s \to t}$ and $\mathbf{P}^{t \to s}$ as follows:

$$\mathbf{C}_{ij}^{s \to t} = \left\lceil \mathbf{P}_{ij}^{s \to t} + \varepsilon \right\rceil - 1 \tag{7}$$

where ε is a hyperparameter that denotes C_j^t are C_i^s matched among G^s and G^t if $\mathbf{P}_{ij}^{s \to t} > 1 - \varepsilon$, i.e., $\mathbf{C}_{ij}^{s \to t} = 1$ when $\mathbf{P}_{ij}^{s \to t} > 1 - \varepsilon$. The same goes for $\mathbf{C}^{t \to s}$.

C. Community-Level and Node-Level Constraint

After initially computing the alignment relationships between cross-network communities based on known anchor pairs, there still exist differences in the representations of aligned communities. Therefore, we utilize community-level constraints to enforce similarity between the representations of communities that have alignment relationships. Specifically, we define the following pairwise objectives for aligned community pairs between the two networks:

$$l(C_i^s, C_j^t) = \log \frac{e^{\theta\left(\boldsymbol{\mu}_i^s, \boldsymbol{\mu}_j^t\right)}}{e^{\theta\left(\boldsymbol{\mu}_i^s, \boldsymbol{\mu}_j^t\right)} + \sum_k e^{\theta\left(\boldsymbol{\mu}_i^s, \boldsymbol{\mu}_k^t\right)}} \tag{8}$$

$$l(C_i^t, C_j^s) = \log \frac{e^{\theta\left(\boldsymbol{\mu}_i^t, \boldsymbol{\mu}_j^s\right)}}{e^{\theta\left(\boldsymbol{\mu}_i^t, \boldsymbol{\mu}_j^s\right)} + \sum_k e^{\theta\left(\boldsymbol{\mu}_i^t, \boldsymbol{\mu}_k^s\right)}} \tag{9}$$

where the community pair (C_i^s, C_j^t) satisfies $\mathbf{C}_{ij}^{s \to t} = 1$ and (C_i^s, C_k^t) satisfies $\mathbf{C}_{ik}^{s \to t} = 0$. Community pair (C_i^t, C_j^s) satisfies $\mathbf{C}_{ij}^{t \to s} = 1$ and (C_i^t, C_k^s) satisfies $\mathbf{C}_{ik}^{t \to s} = 0$.

In fact, the general consistency assumption is expected that anchor pairs in different networks exhibit similar structures or share common neighbors. Specifically, the corresponding anchor nodes are likely to have overlapping neighbors, which in turn can be considered as corresponding anchor node pairs. We leverage the known anchor pairs as bridges to minimize the dissimilarity between the representations of potential anchor pairs in the embedding space. For each known anchor pair (v_i^s, v_i^t), the embeddings of v_i^s and v_i^t should be very similar even the same in the embedding space. And we fuse this prior knowledge to constrain the embedding of known anchor nodes learned by the encoder

$$l_{\text{anchor}} = \sum_{(v_i^s, v_i^t) \in E^a} \left\| \mathbf{h}_i^s - \mathbf{h}_i^t \right\| \tag{10}$$

where $\mathbf{h}_i^s$ and $\mathbf{h}_i^t$ are the embeddings of anchor nodes v_i^s and v_i^t in G^s and G^t, respectively. E^a is the anchor pairs set. We could reduce the difference between known anchor pairs by making the value of this constraint as small as possible. When two nodes from different networks have the same anchor pairs as neighbors, there is a high probability that their representations will be the same, thus becoming potential anchor pairs.

Finally, we combine the GCN-based encoder objective and the above constraints, and train the following loss function to

learn node representations:

$$\mathbf{L} = l^s_{\text{modularity}} + l^t_{\text{modularity}} + l_{\text{anchor}} - \frac{1}{|K_s|} l\left(C^s_i, C^t_j\right) - \frac{1}{|K_t|} l\left(C^t_i, C^s_j\right) \tag{11}$$

where K_s and K_t represent the number of communities in networks G^s and G^t, respectively.

D. Network Alignment Based on Node Representations

Instead of finding corresponding anchor pairs in the entire target network, we match nodes in the restricted search space. Based on the learned alignment relationships between cross-network communities, we utilize a similarity measurement for nodes in matched communities to perform network alignment.

After acquiring the final node representations and community assignments, we match the potential anchor pairs. For a given node $v^s_i \in G^s$, $i \in \{1, 2, \ldots, |V^s|\}$, we need to find the most similar node $v^t_j \in G^t$, $j \in \{1, 2, \ldots, |V^t|\}$ for it. The communities $C^s_{k_i}$, $C^t_{k_j}$ that v^s_i, v^s_j belong to are matched from the view of G^s, i.e., $\mathbf{C}^{s\to t}_{k_i k_j} = 1$. Therefore, for each node $v^s_i \in G^s$ finding the corresponding node in G^t, we can use a mapping function to predict whether a pair of nodes (v^s_i, v^t_j) are anchor pairs in the restricted search space $v^t_j \in C^t_{k_j}$

$$\mathbf{Y}^{s\to t}\left(v^s_i, v^t_j\right) = \begin{cases} 1, & \text{if } \theta\left(\mathbf{h}^s_i, \mathbf{h}^t_j\right) > \theta\left(\mathbf{h}^s_i, \mathbf{h}^t_k\right), \quad k \neq j \\ 0, & \text{otherwise.} \end{cases} \tag{12}$$

We find the corresponding node in G^s for $v^t_i \in G^t$ in the same way, and generate result $\mathbf{Y}^{t\to s}$.

More generally, we let the matrix $\mathbf{Y}^{s\to t}$ represent the probability that the node and the candidate node are matching nodes

$$\mathbf{Y}^{s\to t}\left(v^s_i, v^t_j\right) = \begin{cases} \theta\left(\mathbf{h}^s_i, \mathbf{h}^t_j\right), & \text{if } \mathbf{C}^{s\to t}_{k_i k_j} = 1 \\ 0, & \text{otherwise.} \end{cases} \tag{13}$$

In this way, we can sort the probability from high to low and return the candidate nodes list.

Algorithm 1 illustrates the whole process of the proposed model.

E. Analysis

To further explain why we can alleviate the impact of heterogeneity from a community view, we analyze the learning process of our CHNA.

As introduced in Section IV-A, in G^s and G^t, we embed a node and its adjacent nodes in the local structure to be close in the embedding space, and generate K communities through clustering. The optimization is to maximize $l_{\text{modularity}}(v, \mathbf{A})$, $v \in V$, i.e., the quality of community partition evaluated on the input network. According to (2) and (3), we need to obtain embeddings for $(\partial \boldsymbol{\mu}/\partial \mathbf{H})$ and $(\partial \mathbf{R}/\partial \mathbf{H})$, which allows us to backpropagate gradients from the loss $l_{\text{modularity}}(v, \mathbf{A})$ to the component that produced the embeddings $\mathbf{H}$. Define a function f as follows:

$$f_{i,l}(\boldsymbol{\mu}, v) = \boldsymbol{\mu}^l_i - \frac{\sum_j \mathbf{R}_{jk} \mathbf{h}^l_j}{\sum_j \mathbf{R}_{jk}} \tag{14}$$

Algorithm 1 Network Computation With Representations Learned by CHNA

Input: Networks G^s and G^t
Output: The alignment matrix $\mathbf{Y}^{s\to t}$ for G^s and $\mathbf{Y}^{t\to s}$ for G^t

```
Construct a GCN-based encoder
for G^s = (V^s, E^s) and G^t = (V^t, E^t) do
    Initialize the first layer embeddings H^s = H^t = I, cross-network community alignment relations matrix C^{s→t} = C^{t→s} = I, community numbers K_s and K_t
    // the initial value of K_s and K_t is the average community number obtained by the Leuven algorithm.
    for epochs do
        Compute node embeddings H^s and H^t for each layer by Eq. (1)
        Compute community embeddings μ^s and μ^t by Eq. (2)
        Compute nodes about each community's allocation R^s and R^t by Eq. (3)
        Compute cross-network community alignment relations matrix C^{s→t} and C^{t→s} by Eq. (7), respectively
        Compute the loss function L by Eq. (11)
        Update parameters W by backpropagation
    end for
end for
return H^s, H^t, C^{s→t}, C^{t→s}, R^s, R^t
Compute alignment matrix Y^{s→t} and Y^{t→s} from embeddings H^s and H^t, respectively, by Eq. (12) or Eq. (13)
```

where $(\boldsymbol{\mu}, v)$ is a fixed point of the iterates if $f(\boldsymbol{\mu}, v) = \mathbf{0}$. Applying the implicit function theorem yields that $(\partial \boldsymbol{\mu}/\partial \mathbf{H}) = -[(\partial f(\boldsymbol{\mu}, \mathbf{H}))/\partial \boldsymbol{\mu}]^{-1}((\partial f(\boldsymbol{\mu}, \mathbf{H}))/\partial \mathbf{H})$, from which $(\partial \mathbf{R}/\partial \mathbf{H})$ can be easily obtained via the chain rule. It could be found that $(\partial f/\partial \boldsymbol{\mu})$ will often be dominated by its diagonal terms (the identity matrix). Therefore, we can approximate $(\partial f/\partial \boldsymbol{\mu})$ by its diagonal, which in turn gives $(\partial \boldsymbol{\mu}/\partial \mathbf{H}) \approx -(\partial f/\partial v)$. From (14), we could find that $\boldsymbol{\mu}^l_i$ is constant with respect to v, since here $\boldsymbol{\mu}$ is a fixed value. Hence, $-(\partial f/\partial v) = (\partial/\partial v)(\sum_j \mathbf{R}_{jk}\mathbf{h}_j)/(\sum_j \mathbf{R}_{jk})$. For a specific node v_i, the updating rule of its embedding $\mathbf{h}_i$ becomes

$$\mathbf{h}_i := \mathbf{h}_i + \eta \frac{\partial f}{\partial v_i} = \mathbf{h}_i - \eta \frac{\partial}{\partial v_i} \frac{\sum_j \mathbf{R}_{jk}\mathbf{h}_j}{\sum_j \mathbf{R}_{jk}}. \tag{15}$$

Note that the update equation is relevant with community $\boldsymbol{\mu}_k$. Consider the case of combining cross-network community information shown in Fig. 1(b), according to (8) and (9), nodes in cross-network communities will gradually get closer as the community representation is close.

As shown in Fig. 1(c), without prior knowledge, we regard the alignment relations between nodes within communities as indicative of relationships between cross-network communities. According to (6), cross-network communities with a greater number of known anchor pairs and similar community embeddings possess higher relevance. Due to the varying number of known anchor nodes in communities, different perspectives from each network yield asymmetric alignment

relations between cross-network communities. Hence, the aforementioned update processes mutually influence one another, facilitating the integration of node and community information. In this way, the model could mitigate the impact of anchor node heterogeneity.

F. Time Complexity

Without loss of generality, we can define the following variables: n_s and n_t as the total number of nodes in G^s and G^t, respectively; n_a as the number of known anchor pairs; d as the dimension of node and community embeddings; e_s and e_t as the total number of edges in G^s and G^t, respectively; K_s and K_t as the number of communities in G^s and G^t, respectively. There are mainly four steps to analyze for time complexity in one network.

1) *GCN-Based Encoder:* Take G^s as an example, the normalized Laplacian matrix $\tilde{\mathbf{A}}^s$ is computed once. Since the adjacency matrix $\mathbf{A}^s$ is sparse and $\mathbf{D}^s$ is a diagonal matrix, the time complexity for calculating $\tilde{\mathbf{A}}^s$ is $O(e_s)$. Therefore, the propagation of l-layers GCN in two networks takes $O(l(e_s d + n_s d^2) + l(e_t d + n_t d^2)) = O((n_s + n_t)d^2)$.
2) *Community Discovery:* As introduce in Section IV-E, forward-pass updates [(2) and (3)] are differentiable functions, and it can automatically compute the approximate backward pass with respect to v (i.e., compute products with approximations to $(\partial \boldsymbol{\mu}/\partial \mathbf{H})$ and $(\partial \mathbf{R}/\partial \mathbf{H})$) by applying standard auto differentiation tools to the final update of the forward pass. Instead of inverting $(\partial f/\partial \boldsymbol{\mu})$, the final iteration requires time $O((n_s + n_t)d(K_s + K_t)) = O((n_s + n_t)d)$.
3) *Cross-Network Community Alignment Relations Learning:* The first term of (6) is a constant, and the main calculation of (6) is the cosine similarity. For a pair of communities, its time complexity is $O(2d)$. Thus, the total time complexity of calculating the similarity of community pairs between two networks is $O(4K_s K_t d^2) = O(K_s K_t d^2)$.
4) *Community-Level and Node-Level Constraint:* The main calculation of communities constraint is the cosine similarity between community pairs, similar to cross-network community alignment relations learning, and the total time complexity is $O(K_s K_t d^2)$. We constrain the known anchor pairs by reducing their embedding distance, and the time complexity is $O(n_a^2 d)$.

Therefore, the total time complexity is $O((n_s + n_t)d^2 + (n_s + n_t)d + 2K_s K_t d^2 + n_a^2 d) = O((n_s + n_t)d^2 + (n_s + n_t)d)$.

V. Experiments

A. Experimental Settings

1) Datasets: We validate the effectiveness of the proposed CHNA using real-world networks, including Aminer (a free online service that offers comprehensive search and mining services for researcher social networks) and LinkedIn (a professional network where users can maintain their profile page and connections). These two networks were collected by [50]. Twitter (a worldwide microblog), Foursquare (a location-based social network), LiveJournal (a free online social network where users can keep a blog, journal, or diary), Flickr (a photo-sharing network), Last.fm (online music service where users can get music recommendations), and MySpace (a social networking website for users to share blogs, photographs, and music). Twitter and Foursquare used in our experiment are obtained from [51], LiveJournal, Flickr, Last.fm, and MySpace are provided by [50]. The specific information of datasets is shown in Table I.

TABLE I
Statistics of the Datasets

Network Pair	#Nodes	#Edges	#Anchor Links	#Average Community	**H**
Aminer LinkedIn	1,056,941 6,726,011	3,929,876 19,360,689	4,269	29,777 326	39
Twitter Foursquare	5,120 5,313	164,919 76,972	1,609	21 31	35
LiveJournal Flickr	3,017,286 214,626	87,037,566 9,114,557	134	1,682 141	119
LiveJournal Last.fm	3,017,286 136,409	87,037,566 1,685,524	471	1,682 346	125
LiveJournal MySpace	3,017,286 854,498	87,037,566 6,489,736	615	1,682 20	104
Flickr Last.fm	214,626 136,409	9,114,557 1,685,524	510	141 346	35
Flickr MySpace	214,626 854,498	9,114,557 6,489,736	378	141 20	33
Last.fm MySpace	136,409 854,498	1,685,524 6,489,736	1,381	346 20	38

As discussed in Section I, the heterogeneity of anchor nodes primarily manifests in their local structures. To quantify this heterogeneity, we employ the degree difference between pairs of anchor nodes. In other words, if a pair of anchor nodes exhibits a significant difference in the number of neighboring nodes between the two networks, it indicates strong heterogeneity between the anchor nodes. Consequently, we calculate the average degree difference of known anchor node pairs between the two networks, denoted as **H**. This average degree difference serves as a measure of the dataset's heterogeneity. Table I provides evidence of the existence of heterogeneity among anchor nodes in these datasets.

2) Baseline Methods: We apply the following state-of-the-art baseline methods to compare with our model: NAME [24], CAPER [23], MEgo2Vec [52], IONE-Con-Ex [53], CrossMNA [54], CAMU [16], GAlign [17], and DHNA [46].

3) Evaluation Metrics: For each matching pair (v_i^s, v_j^t) in the test set, we rank the target nodes in the result according to $\cos(\mathbf{h}_i^s, \mathbf{h}_j^t)$. To quantitatively evaluate this ranking, we select Precision@α ($P@\alpha$) and mean reciprocal rank (MRR) [20] as metrics. Specifically, $P@\alpha = (|M@\alpha|/|U|)$ indicates whether the true positive match occurs in top-α candidates, where $|M@\alpha|$ is the count of the correct alignments between networks G^s and G^t in top-α choices, and $|U|$ is the

TABLE II

EXPERIMENTAL RESULTS ON DIFFERENT DATASETS. THE BEST AND RUNNER-UP RESULTS ARE HIGHLIGHTED IN BOLDFACE AND UNDERLINED, RESPECTIVELY. ⋆% DENOTES THE IMPROVEMENT OF CHNA COMPARED TO THE BEST BASELINE METHODS RESULTS

Metric	Method	Dataset							
		Aminer LinkedIn	Twitter Foursquare	LiveJournal Flickr	LiveJournal Last.fm	LiveJournal Myspace	Flickr Last.fm	Flickr Myspace	Last.fm Myspace
$P@10$	NAME	0.1809	0.7200	0.2559	0.2582	0.2566	0.2886	0.2632	0.3109
	CAPER	0.1706	0.6793	0.2431	0.2749	0.2529	0.2504	0.2432	0.2805
	MEgo2Vec	0.1083	0.6500	0.2271	0.2279	0.2221	0.2450	0.2400	0.2479
	IONE-Con-Ex	0.1401	0.7320	0.2706	0.2812	0.2706	0.2950	0.2000	0.2217
	CrossMNA	0.1628	0.6541	0.1625	0.1857	0.2502	0.2010	0.2152	0.2470
	CAMU	0.1715	0.7132	**0.2812**	0.2966	0.2788	**0.3122**	0.2600	0.2839
	GAlign	0.1938	0.7200	0.2742	0.2843	0.2851	0.2919	0.2600	0.2947
	DHNA	-	**0.7533**	-	-	-	-	-	-
	CHNA	**0.2000**	0.7401	0.2738	**0.3098**	**0.2978**	0.3000	**0.2715**	**0.3287**
	▲%	3.20%	1.11%	-2.63%	4.45%	4.45%	-3.91%	3.15%	5.73%
MRR	NAME	0.1176	0.5700	0.1382	0.1344	0.1604	0.1901	0.2005	0.1900
	CAPER	0.1208	0.5527	0.1409	0.1396	0.1600	0.1627	0.1700	0.1782
	MEgo2Vec	0.0136	0.3500	0.0222	0.0270	0.0399	0.0297	0.0215	0.0818
	IONE-Con-Ex	0.0231	0.3725	0.0280	0.0280	0.0423	0.0281	0.0326	0.0876
	CrossMNA	0.0210	0.3111	0.0350	0.0372	0.0584	0.0359	0.0379	0.0491
	CAMU	0.1176	0.3509	**0.2044**	0.1846	0.1880	0.1722	**0.2520**	0.2044
	GAlign	0.1443	0.5741	0.1655	0.1723	0.1886	0.1888	0.1855	0.2001
	DHNA	-	0.5620	-	-	-	-	-	-
	CHNA	**0.1679**	**0.5790**	0.1876	**0.1912**	**0.2040**	**0.2000**	0.2366	**0.2192**
	⋆%	16.35%	0.85%	-8.22%	3.58%	8.16%	5.21%	-6.11%	7.24%

Due to the consideration of the asymmetric alignment relations between cross-network communities in our approach, there may be some differences between the alignment results from G^s to G^t and from G^t to G^s. Thus we use the average value of $P^{(s\to t)}@10$ and $P^{(t\to s)}@10$ as the final result of $P@10$, where $P^{(s\to t)}@10$ and $P^{(t\to s)}@10$ represent the ranking result from the view that from G^s to G^t and G^t to G^s, respectively. The same for the MRR evaluation.

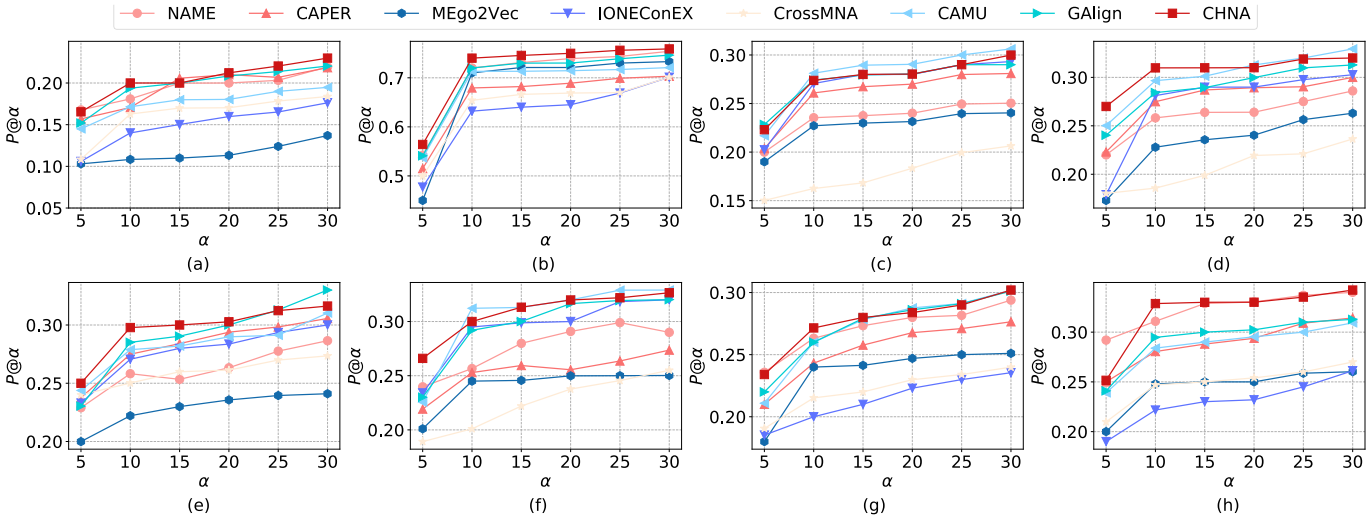

Fig. 3. $P@\alpha$ results over different α values. (a) Aminer-LinkedIn. (b) Twitter-Foursquare. (c) LiveJournal-Flickr. (d) LiveJournal-Last.fm. (e) LiveJournal-Myspace. (f) Flickr-Last.fm. (g) Flickr-Myspace. (h) Last.fm-Myspace.

number of anchor links in the ground truth set. MRR $= (1/|U|)\sum_{(v_i^s,v_j^t)}(1/(\text{rank}(v_j^t)))$, where $\text{rank}(v_j^t)$ is the rank of true anchor target in the sorted list of anchor candidates.

4) Implementation Details: We utilize a two-layer (i.e., $l = 2$) GCN as the encoder, with the ReLU$(\cdot)$ as the activation function. To perform community discovery, we employ the Leuven algorithm [55] iteratively for ten times and calculate the average number of communities in each network as the initial value of K. For training our method, we use the Adam optimizer with the following hyperparameter settings: $\beta = 70$, $\xi = 0.4$, learning rate $= 0.0001$, and dropout rate $= 0.2$ for all experiments. To measure the performance in the network alignment task, we randomly split the known anchor node pairs into a training set and a test set at a ratio of 4:1. We set the same node embedding dimension $d = 200$ for all methods and use d as the community embedding dimension in CHNA. Regarding the baseline methods, we apply their respective best parameter settings for each method.

B. Model Performance Analysis

Following the widely adopted way of validating network alignment, we evaluate the proposed CHNA and baseline methods via predicting anchor links between networks, where $P@\alpha$ and MRR are adopted as the metrics. Table II shows the

experimental results of our method and baselines at the metric of $P@10$ and MRR. Fig. 3 shows the results of all methods on different α varying from 5 to 30.

1) Precision Improvement: Our model outperforms most of the baselines across all datasets in terms of the two metrics. Compared to the best baseline, we achieve at most a 5.73% improvement at the metric of $P@10$, and a significant improvement over the state-of-the-art methods at the metric of MRR. The results demonstrate the effectiveness of our method. Besides, CAMU, GAlign, and NAME also show good performance. Although NAME also considers community information, it primarily focuses on multiorder topological consistency, resulting in slightly worse performance compared to our CHNA on datasets with heterogeneity. CAPER finds the coarse structure (i.e., community structure) in the network, then perform the coarse structure and node-level alignment, satisfying the one-to-one constraint. However, as illustrated in Fig. 1(c), the heterogeneity of anchor nodes can result in an asymmetric cross-network community relationships. This can lead to cross-network community pairs no longer satisfying the one-to-one constraint, thereby affecting the performance of CAPER.

It is worth noting that we only compared with DHNA on the Twitter-Foursquare dataset. This is because the time complexity of the DHNA method limits its performance on large-scale datasets. The time complexity of DHNA is $O((n_s + n_t)d^2 + (e_s + e_t + e_a + n_s n_t)d)$, where n_s and n_t are the node numbers of G^s and G^t, respectively, while e_s and e_t represent the edge numbers of G^s and G^t, respectively. Additionally, e_a denotes the number of anchor pairs between G^s and G^t. It can be seen that the time complexity of DHNA is positively correlated with the product of the total number of nodes in the two networks. As the number of nodes in the networks to be aligned increases, the time required by DHNA increases significantly. Furthermore, due to the large memory consumption of the DHNA method when dealing with large-scale networks, we only compared with it on the Twitter-Foursquare dataset. With a known anchor node pair ratio of 4:1 for training and testing sets, DHNA achieved a $P@10$ result of 0.7533, slightly higher than the proposed CHNA in this article. DHNA yields superior results by directly searching candidate nodes across the entire target network, as opposed to limiting the search within aligned communities. Because the correctness of cross-network community alignment relations learning and the reduction of candidate node matching scope through community alignment relations both have an impact on node alignment. However, with the increasing scale of networks, the incorporation of cross-network community alignment enhances the scalability of our CHNA model, making it more applicable to practical network alignment tasks.

2) Impact of Different α Choices: Fig. 3 shows the results of $P@\alpha$ over different α values (except for DHNA), ranging from 5 to 30. All methods show a precision ascent with a larger value of α. When α is small, the limited performance of all methods can be attributed to the fact that only network structure is used for alignment. Only using network structure makes it unable to distinguish between nodes with similar local structures hampers the alignment effectiveness.

TABLE III

EXPERIMENTAL RESULTS UNDER THE DIFFERENT RATIO OF TRAINING ANCHOR PAIRS ON TWITTER-FOURSQUARE

		Ratio of training anchor links		
Metric	Method	20%	50%	80%
$P@10$	NAME	0.5003	0.6057	0.7200
	CAPER	0.4802	0.5772	0.6793
	MEgo2Vec	0.3461	0.5597	0.6500
	IONE-Con-Ex	0.3926	0.6382	0.7320
	CrossMNA	0.3829	0.5558	0.6541
	CAMU	0.4590	0.6447	0.7132
	GAlign	0.4821	0.6462	0.7200
	DHNA	0.4402	0.6305	**0.7533**
	CHNA	**0.5115**	**0.6517**	0.7401
MRR	NAME	0.3300	**0.4551**	0.5700
	CAPER	0.3255	0.4191	0.5527
	MEgo2Vec	0.2011	0.2292	0.3500
	IONE-Con-Ex	0.2331	0.3402	0.3725
	CrossMNA	0.2163	0.2977	0.3111
	CAMU	0.3755	0.2672	0.3509
	GAlign	**0.3793**	0.4374	0.5741
	DHNA	0.3175	0.4022	0.5620
	CHNA	0.3657	0.4300	**0.5790**

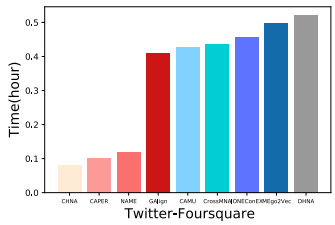

Fig. 4. Time for finding anchor pairs in test set on Twitter-Foursquare dataset.

3) Effect of Anchor Pairs Percentage: The unsupervised embedding-based methods CAMU and GAlign generally perform better than the supervised embedding-based methods IONE-Con-Ex and CrossMNA. Although our method needs known anchor pairs to guide community alignment, their importance is primarily observed during the initialization stage of the cross-network community alignment matrix. Subsequently, the cross-network community alignment relationships are primarily updated and computed based on community representations. Thus, CHNA can guarantee good performance even when the known anchor node pairs account for a small proportion of the entire network. MEgo2Vec has poor performance compared to other baselines because it requires known anchor nodes to construct the ego network and learn the node embedding, which depends on the number of known anchor nodes. Furthermore, we verify the dependence of each method on the known anchor nodes on the Twitter-Foursquare dataset by changing the proportion of anchor nodes used for training and testing. The results are listed in Table III. Each method is influenced to some extent by the ratio of training anchor nodes. Supervised methods, such as IONE-Con-Ex, DHNA, and CrossMNA, are particularly sensitive to the ratio of training anchor nodes and experience a significant impact from it. Hence, when the training anchor pairs are sparse, our method CHNA can obtain superior performance compared with most baselines.

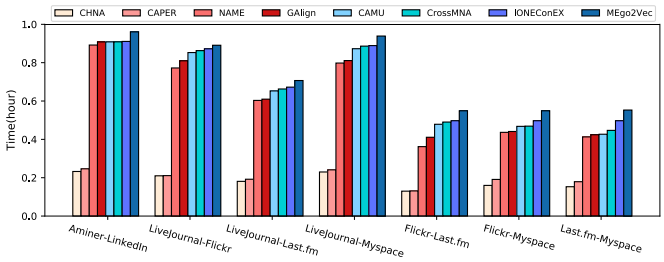

Fig. 5. Time for finding anchor pairs in test set on each dataset (except for Twitter-Foursquare dataset).

4) Time for Searching Anchor Pairs: To investigate the effectiveness of reducing the search range through cross-network community alignment relations, we compare the time required for aligning nodes in the test set across different datasets. The results are shown in Figs. 4 and 5. Due to the high time complexity of DHNA, aligning two larger networks takes nearly a day, which is significantly longer than other baselines. Therefore, we only compare its runtime on the Twitter-Foursquare dataset as shown in Fig. 4. It can be seen that even for relatively small network alignment tasks, DHNA still requires the most alignment time. Our method demonstrate the fastest node matching in the test sets of all datasets. This is because our method find the corresponding node in G^t for a given node in G^s based on calculating similarity and sorting the nodes within the aligned community, which could typically contains fewer nodes compared to G^t. On the other hand, NAME, IONE-Con-Ex, CrossMNA, CAMU, and GAlign require more time as they need to search the entire network. MEgo2Vec exhibits the longest runtime as it necessitates the construction of an additional "ego" network.

C. Ablation Study

In this section, we conduct ablation studies to validate the effectiveness of using community information and modeling the asymmetric cross-network community alignment relationships. We design three model variants as following: CHNA-0 that without both community-level and node-level constraints [(8)–(10)]; CHNA-1 that without community-level constraint [(8) and (9)]; CHNA-2 that regards the cross-network community alignment relationships as symmetric. That is, CHNA-2 calculates the cross-network community correlation between network G^s and G^t as follows:

$$\mathbf{P}_{ij} = \frac{n_{\left(C_i^s, C_j^t\right)}}{n_{C_i^s} + n_{C_j^t}} \cdot e^{\theta\left(\boldsymbol{\mu}_i^s, \boldsymbol{\mu}_j^t\right)} \cdot e^{\mathbf{C}_{ij}} / \xi \tag{16}$$

$$\mathbf{C}_{ij} = \left\lceil \mathbf{P}_{ij} + \varepsilon \right\rceil - 1. \tag{17}$$

Fig. 6 shows the results. From the result of CHNA-0, we can find that the node-level constraint plays a crucial role in network alignment tasks, indicating that the majority of anchor node pairs between networks exhibit consistency. Moreover, it can be seen that the community information we used and the way it is used is helpful to alleviate the impact of heterogeneity compared to CHNA-1 and CHNA-2. Furthermore, we can find that assuming the relationships of cross-network communities as symmetric even leads to worse

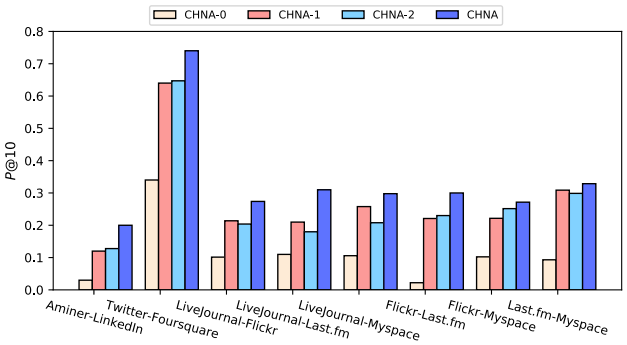

Fig. 6. Results of ablation study.

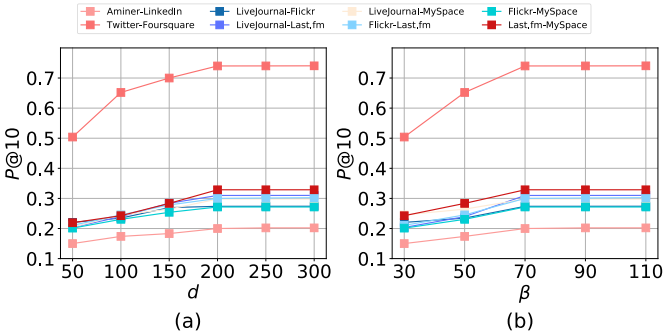

Fig. 7. Parameter study on embedding dimension and community discovery effect control parameter β. (a) $P@10$ versus dimension. (b) $P@10$ versus β.

performance than not considering community information. It is because the wrong correspondence between cross-network communities will cause all nodes in the two communities to miss the correct matching nodes. In all, the experimental results of ablation studies verify our motivation in Section I.

D. Hyperparameter Sensitivity

To assess the impact of CHNA's hyperparameters on performance, we conducted several experiments to analyze accuracy under varying hyperparameter settings.

1) Impact of Embedding Dimensionality d: Fig. 7(a) shows the sensitivity of the embedding dimension of the encoder. Generally, selecting a high number of dimensions does not lead to a significant improvement in performance ($P@10$), while it increases both time and space complexity. Therefore, it is advisable to avoid excessively high dimensions to maintain a favorable trade-off between performance and resource requirements.

2) Impact of the Hyperparameter β: Fig. 7(b) presents the impact of the hyperparameter β, which controls the hardness of community allocation and influences the effectiveness of community discovery. In general, a larger value of β yields better results. Through experimentation and analysis, we determined the optimal value of β to be $\beta = 70$.

3) Impact of the Number of Communities: In the CHNA, we need to set the number of communities in the network before training. We choose the Last.fm-MySpace dataset to verify the influence of the number of communities because there is some difference between the average community number of them and the number of known anchor node pairs is

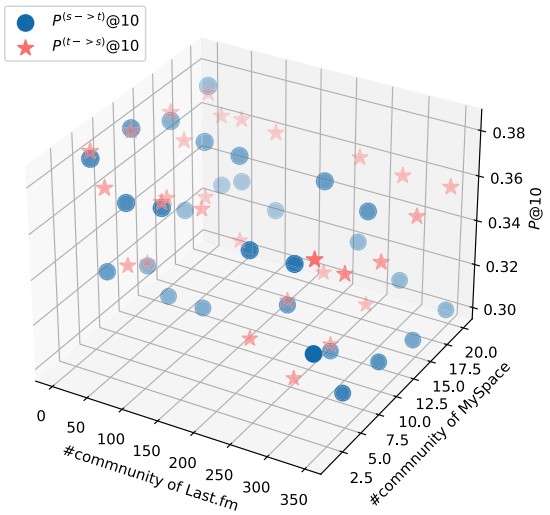

Fig. 8. Parameter study on the number of communities and the heterogeneity at the community level. $P^{(s\to t)}@10$ represents the result ranking from Last.fm to MySpace, and $P^{(t\to s)}@10$ represents the result ranking from MySpace to Last.fm.

relatively sufficient. Fig. 8 depicts the variation of the number of communities in Last.fm, ranging from 1 to 346, and in MySpace, ranging from 1 to 20. It can be observed that the change in the number of communities does indeed have a specific impact on the matching accuracy. However, overall, it has a negligible effect (with $P@10$ results fluctuating between 0.3 and 0.38). In general, a smaller number of communities corresponds to higher matching accuracy. This is because in such scenarios, a larger pool of nodes can be matched and selected, increasing the likelihood of finding corresponding nodes for those in the test set. This also reflects that asymmetric community alignment relations are essentially caused by the distribution of anchor nodes or the heterogeneity of anchor nodes.

4) Impact of the Heterogeneity: In the proposed CHNA approach, we consider the asymmetry of community alignment relations between different networks as a manifestation of heterogeneity at the community level. Consequently, to align a pair of nodes, we take into account the rankings of results from G^s to G^t and from G^t to G^s, respectively. As shown in Fig. 8, we observe variations in the node alignment results when viewed from different network perspectives, with only a slight overlap in the ranking of results from G^s to G^t and G^t to G^s. Similar to the impact of the number of communities, when the network has a smaller number of communities, each community tends to contain a relatively larger number of nodes. This not only increases the potential for matching more nodes but also raises the likelihood of correctly identifying communities that can be aligned.

E. Case Study

Fig. 9 visually represents the asymmetric community alignment relations within the Twitter-Foursquare social networks, which were employed in our experiments. The intensity of the color corresponds to the likelihood of

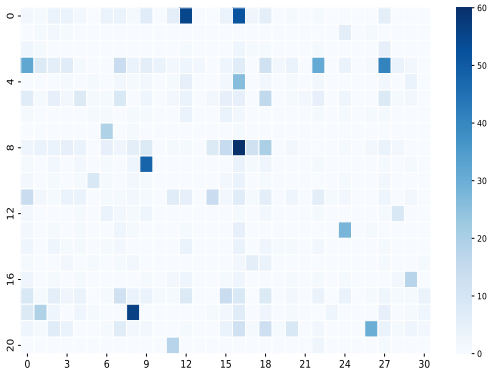

Fig. 9. Asymmetric alignment relations of cross-network communities that reflects the impact of anchor nodes' heterogeneity at the community level.

alignment between community pairs. In an ideal scenario with completely symmetrical community pair alignment, the diagonal color should be the darkest, gradually becoming lighter for the remaining regions. However, we observe an uneven distribution of darker color patches, indicating the presence of asymmetry in community pair alignment. This asymmetry is a consequence of the anchor node heterogeneity described in this article at the community level.

VI. Conclusion

This article focuses on network alignment by designing an alignment model that utilizes cross-community relations information for guidance. The model takes into account both consistency and heterogeneity. To ensure consistency, known anchor nodes are used to bring potential anchor nodes closer together. Regarding heterogeneity, we consider the community as a fundamental network structure that reflects common features of nodes, thereby alleviating the impact of heterogeneity in network alignment from a community perspective. We introduce a framework that simultaneously learns node representations and discovers communities for each network. Then, the alignment of cross-network communities is guided by known anchor nodes, and a pairwise constraint on communities between different networks is applied to establish alignment between communities. Subsequently, node correspondence is established based on the similarities of their representations in the asymmetrically aligned communities, which improves alignment accuracy while reducing time complexity. Experimental results demonstrate that CHNA learns embeddings that preserve the node structure while considering heterogeneity, resulting in improved network alignment performance. Specifically, it achieves an improvement of up to 5.73% in $P@10$ compared to state-of-the-art methods. Additionally, it effectively reduces the time complexity of node matches between different networks.

In future work, we will delve deeper into exploring the heterogeneity caused by anchor nodes. Furthermore, we will also consider filtering anchor nodes, that is, selecting "appropriate" anchor nodes to guide the matching of communities and networks.

REFERENCES

[1] Y. Zhou et al., "Robust network alignment via attack signal scaling and adversarial perturbation elimination," in *Proc. Web Conf.*, 2021, pp. 3884–3895.

[2] A. Almulhim, V. S. Dave, and M. Al Hasan, "Network alignment using graphlet signature and high order proximity," in *Proc. Int. Conf. Mach. Learn., Optim., Data Sci.* Cham, Switzerland: Springer, 2019, pp. 130–142.

[3] W. Zhao et al., "Learning to map social network users by unified manifold alignment on hypergraph," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 12, pp. 5834–5846, Dec. 2018.

[4] 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.* New York, NY, USA: Association for Computing Machinery, Oct. 2013, pp. 179–188.

[5] W. Zhang, X. Lai, and J. Wang, "Social link inference via multiview matching network from spatiotemporal trajectories," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 34, no. 4, pp. 1720–1731, Apr. 2023.

[6] W. Liang and W. Zhang, "Learning social relations and spatiotemporal trajectories for next check-in inference," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 34, no. 4, pp. 1789–1799, Apr. 2023.

[7] E. Epaillard and N. Bouguila, "Variational Bayesian learning of generalized Dirichlet-based hidden Markov models applied to unusual events detection," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 4, pp. 1034–1047, Apr. 2019.

[8] Y. Liu, Z. Li, S. Pan, C. Gong, C. Zhou, and G. Karypis, "Anomaly detection on attributed networks via contrastive self-supervised learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 6, pp. 2378–2392, Jun. 2022.

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

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

[11] M. Heimann, X. Chen, F. Vahedian, and D. Koutra, "Refining network alignment to improve matched neighborhood consistency," in *Proc. SIAM Int. Conf. Data Mining (SDM)*, 2021, pp. 172–180.

[12] 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*, Feb. 2013, pp. 495–504.

[13] R. Zafarani and H. Liu, "Connecting users across social media sites: A behavioral-modeling approach," in *Proc. 19th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*. New York, NY, USA: Association for Computing Machinery, Aug. 2013, pp. 41–49.

[14] J. Feng et al., "DPLink: User identity linkage via deep neural network from heterogeneous mobility data," in *Proc. World Wide Web Conf.* New York, NY, USA: Association for Computing Machinery, May 2019, pp. 459–469.

[15] W. Chen, H. Yin, W. Wang, L. Zhao, and X. Zhou, "Effective and efficient user account linkage across location based social networks," in *Proc. IEEE 34th Int. Conf. Data Eng. (ICDE)*, Apr. 2018, pp. 1085–1096.

[16] 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.

[17] H. T. Trung, T. Van Vinh, N. T. Tam, H. Yin, M. Weidlich, and N. Q. V. Hung, "Adaptive network alignment with unsupervised and multi-order convolutional networks," in *Proc. IEEE 36th Int. Conf. Data Eng. (ICDE)*, Apr. 2020, pp. 85–96.

[18] Y. Yan, S. Zhang, and H. Tong, *BRIGHT: A Bridging Algorithm for Network Alignment*. New York, NY, USA: Association for Computing Machinery, 2021, pp. 3907–3917.

[19] Q. Peng, Y. Wang, P. Jiao, H. Wu, and W. Wang, "Accurate network alignment via consistency in node evolution," *IEEE Trans. Big Data*, pp. 1–14, 2024.

[20] 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. (IJCAI)*, New York, NY, USA, 2016, pp. 1823–1829.

[21] Z. Chen, X. Yu, B. Song, J. Gao, X. Hu, and W.-S. Yang, "Community-based network alignment for large attributed network," in *Proc. ACM Conf. Inf. Knowl. Manage.* New York, NY, USA: Association for Computing Machinery, Nov. 2017, pp. 587–596.

[22] S. Zhang, H. Tong, R. Maciejewski, and T. Eliassi-Rad, "Multilevel network alignment," in *Proc. World Wide Web Conf.* New York, NY, USA: Association for Computing Machinery, May 2019, pp. 2344–2354.

[23] J. Zhu, D. Koutra, and M. Heimann, "CAPER: Coarsen, align, project, refine—A general multilevel framework for network alignment," in *Proc. 31st ACM Int. Conf. Inf. Knowl. Manage.* New York, NY, USA: Association for Computing Machinery, Oct. 2022, pp. 4747–4751, doi: 10.1145/3511808.3557563.

[24] T. T. Huynh et al., "Network alignment with holistic embeddings," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 2, pp. 1881–1894, Feb. 2023.

[25] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. 5th Int. Conf. Learn. Represent. (ICLR)*, Toulon, France, Apr. 2017, pp. 1–14.

[26] J. Zhang and P. S. Yu, "Multiple anonymized social networks alignment," in *Proc. IEEE Int. Conf. Data Mining*, Nov. 2015, pp. 599–608.

[27] V. Saraph and T. Milenković, "MAGNA: Maximizing accuracy in global network alignment," *Bioinformatics*, vol. 30, no. 20, pp. 2931–2940, Oct. 2014.

[28] S. Zhang and H. Tong, "FINAL: Fast attributed network alignment," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*. New York, NY, USA: Association for Computing Machinery, Aug. 2016, pp. 1345–1354.

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

[30] Y. Zhou, J. Ren, R. Jin, Z. Zhang, D. Dou, and D. Yan, "Unsupervised multiple network alignment with multinominal GAN and variational inference," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Dec. 2020, pp. 868–877.

[31] 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.

[32] J. Gao, X. Huang, and J. Li, "Unsupervised graph alignment with Wasserstein distance discriminator," in *Proc. 27th ACM SIGKDD Conf. Knowl. Discovery Data Mining*, Aug. 2021, pp. 426–435.

[33] Z. Liang et al., *Unsupervised Large-Scale Social Network Alignment Via Cross Network Embedding*. New York, NY, USA: Association for Computing Machinery, 2021, pp. 1008–1017.

[34] Y. Wang et al., "Geometry interaction network alignment," *Neurocomputing*, vol. 501, pp. 618–628, Aug. 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0925231222008086

[35] D. Zhang, J. Yin, X. Zhu, and C. Zhang, "Network representation learning: A survey," *IEEE Trans. Big Data*, vol. 6, no. 1, pp. 3–28, Mar. 2020.

[36] M. Gao, P. Jiao, R. Lu, H. Wu, Y. Wang, and Z. Zhao, "Inductive link prediction via interactive learning across relations in multiplex networks," *IEEE Trans. Computat. Social Syst.*, vol. 11, no. 3, pp. 3118–3130, 2024.

[37] W. Zhang, Y. Yu, T. Pan, L. Pan, P. Jiao, and W. Wang, "Generating structural node representations via higher-order features and adversarial learning," in *Proc. IEEE Int. Conf. Data Mining (ICDM)*, Dec. 2021, pp. 1487–1492.

[38] Q. Peng, W. Wang, H. Liu, C. Huo, and M. Shao, "Graph collaborative expert finding with contrastive learning," in *Proc. 33rd Int. Joint Conf. Artif. Intell.*, Aug. 2024, pp. 2288–2296.

[39] W. Wang, M. Luo, C. Yan, M. Wang, X. Zhao, and Q. Zheng, "Cross-graph representation learning for unsupervised graph alignment," in *Proc. Int. Conf. Database Syst. Adv. Appl.* Cham, Switzerland: Springer, 2020, pp. 368–384.

[40] C. Li et al., "Adversarial learning for weakly-supervised social network alignment," in *Proc. AAAI Conf. Artif. Intell.*, 2019, vol. 33, no. 1, pp. 996–1003.

[41] H. Chen, H. YIN, X. Sun, T. Chen, B. Gabrys, and K. Musial, *Multi-Level Graph Convolutional Networks for Cross-Platform Anchor Link Prediction*. New York, NY, USA: Association for Computing Machinery, 2020, pp. 1503–1511.

[42] Z. Zeng, S. Zhang, Y. Xia, and H. Tong, "PARROT: Position-aware regularized optimal transport for network alignment," in *Proc. ACM Web Conf.*, Austin, TX, USA, Apr./May 2023, pp. 372–382.

[43] C. Li et al., "Semi-supervised variational user identity linkage via noise-aware self-learning," *IEEE Trans. Knowl. Data Eng.*, pp. 1–14, 2023.

[44] M. Long, S. Chen, X. Du, and J. Wang, "DegUIL: Degree-aware graph neural networks for long-tailed user identity linkage," in *Proc. ECML PKDD* (Lecture Notes in Computer Science), vol. 14174. Turin, Italy: Springer, Sep. 2023, pp. 122–138.
[45] X. Guo, Y. Liu, D. Gong, and F. Liu, "Dual graph convolutional networks for social network alignment," *IEEE Trans. Big Data*, pp. 1–12, 2024.
[46] Y. Wang et al., "Network alignment enhanced via modeling heterogeneity of anchor nodes," *Knowl.-Based Syst.*, vol. 250, Aug. 2022, Art. no. 109116.
[47] G. Ciriello, M. Mina, P. H. Guzzi, M. Cannataro, and C. Guerra, "AlignNemo: A local network alignment method to integrate homology and topology," *PLoS ONE*, vol. 7, no. 6, Jun. 2012, Art. no. e38107.
[48] B. Wilder, E. Ewing, B. Dilkina, and M. Tambe, "End to end learning and optimization on graphs," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, 2019, pp. 4672–4683.
[49] M. E. J. Newman, "Modularity and community structure in networks," *Proc. Nat. Acad. Sci. USA*, vol. 103, no. 23, pp. 8577–8582, 2006.
[50] Y. Zhang, J. Tang, Z. Yang, J. Pei, and P. S. Yu, *COSNET: Connecting Heterogeneous Social Networks with Local and Global Consistency*. New York, NY, USA: Association for Computing Machinery, 2015, pp. 1485–1494.
[51] J. Zhang and P. S. Yu, "Integrated anchor and social link predictions across social networks," in *Proc. 24th Int. Conf. Artif. Intell.* Buenos Aires, Argentina: AAAI Press, 2015, pp. 2125–2131.
[52] J. Zhang et al., "MEgo2Vec: Embedding matched ego networks for user alignment across social networks," in *Proc. 27th ACM Int. Conf. Inf. Knowl. Manage.* New York, NY, USA: Association for Computing Machinery, 2018, pp. 327–336.
[53] L. Liu, X. Li, W. K. Cheung, and L. Liao, "Structural representation learning for user alignment across social networks," *IEEE Trans. Knowl. Data Eng.*, vol. 32, no. 9, pp. 1824–1837, Sep. 2020.
[54] 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.*, May 2019, pp. 273–284.
[55] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding of communities in large networks," *J. Stat. Mech., Theory Exp.*, vol. 2008, no. 10, Oct. 2008, Art. no. P10008.

Qiyao Peng received the bachelor's degree in electrical and information engineering from Shandong University, Shandong, China, in 2018, and the master's degree from Tianjin University, Tianjin, China, in 2021, where he is currently pursuing the Doctor degree.

His research interests include graph neural networks, expert finding, and large-scale data mining.

Yinghui Wang received the B.E., M.S., and Eng.D. degrees from Tianjin University, Tianjin, China, in 2015, 2018, and 2024, respectively.

She is currently an Engineer at the Key Laboratory of Information System and Technology, Beijing Institute of Control and Electronic Technology, Beijing, China. Her research interests include complex network analysis and its applications, graph neural networks, and network embedding.

Pengfei Jiao (Member, IEEE) received the Ph.D. degree in computer science from Tianjin University, Tianjin, China, in 2018.

From 2018 to 2021, he was a Lecturer with the Center of Biosafety Research and Strategy, Tianjin University. He is currently a Professor with the School of Cyberspace, Hangzhou Dianzi University, Hangzhou, China. His current research interests include complex network analysis and its applications.

Huaming Wu (Senior Member, IEEE) received the B.E. and M.S. degrees in electrical engineering from Harbin Institute of Technology, Harbin, China, in 2009 and 2011, respectively, and the Ph.D. degree (Hons.) in computer science from Freie Universität Berlin, Berlin, Germany, in 2015.

He is currently an Associate Professor at the Center for Applied Mathematics, Tianjin University. His research interests include wireless networks, mobile edge computing, internet of things, and complex networks.

Lin Pan received the Ph.D. degree in computer science from Tianjin University, Tianjin, China, in 2016.

He is a Lecturer at the School of Marine Science and Technology, Tianjin University. His current research interests include complex network analysis, social computing, and marine data mining.