This article has been accepted for publication in IEEE Transactions on Big Data. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TBDATA.2024.3407543

Accurate Network Alignment via Consistency in Node Evolution

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

Abstract—Network alignment, which integrates multiple network resources by identifying anchor nodes that exist in different networks, is beneficial for conducting comprehensive network analysis. Although there have been many studies on network alignment, most of them are limited to static scenarios and only can achieve acceptable top-α ($\alpha > 10$) results. In the absence of considering dynamic changes in networks, accurate network alignment (i.e., top-1 result) faces two problems: 1) Missing information: focusing solely on aligning networks at a specific time leads to low top-1 performance due to the lack of information from other time periods; 2) Confusing information: ignoring temporal information and focusing on aligning networks across the entire time span leads to low top-1 performance due to inability to distinguish the neighborhood nodes of anchor nodes. In this paper, we propose a dynamic network alignment method, which aims to achieve better top-1 alignment results with consider changing network structures over time. Towards this end, we learn the representations of nodes in the changing network structure with time, and preserve the consistency of anchor node pairs during the time-evolution process. Firstly, we employ a Structure-Time-aware module to capture network dynamics while preserving network structure and learning node representations that incorporate temporal information. Secondly, we ensure the global and local consistency of anchor node pairs over time by utilizing linear and similarity functions, respectively. Finally, we determine whether two nodes are anchor node pairs by maintaining consistency between global, local, and node representations. Experimental results obtained from real-world datasets demonstrate that the proposed model achieves performance comparable to several state-of-the-art methods.

Index Terms—Network alignment, network evolution, global consistency, local consistency.

1 Introduction

Network alignment is a process that compares two networks to identify common nodes. It has gained attention for its broad applications, such as comparing gene networks or protein-protein interaction networks in biology [1], [2], collecting the accounts belonging to the same person in different online platforms in sociology [3], [4], inferring the cross-layer alignment of wired and wireless networks in computer science [5], [6], inferring relationships among entities from different sources and to facilitate transfer learning in knowledge graphs [7], [8], [9], [10]. The common nodes found in different networks are referred to as anchor nodes [11]. Anchor nodes represent entities such as the accounts of the same person across different social networks. The correspondence between pairs of anchor nodes, which represent the same entity, is known as anchor links.

Network alignment tasks have been thoroughly investigated and many methods have been proposed [11], [12], [13], [14], [15], [16], [17], [18], [19], [20], [21], [22], [23], [24]. When predicting anchor links, a majority of these methods hold a consistency assumption, i.e., the higher the percentage of shared nodes, the more likely two nodes are corresponding anchor nodes. However, most existing research focuses on the alignment between static networks, ignoring the dynamic evolution of the network structure in real-world scenarios.

The dynamic evolution of network structures refers to the appearance of new nodes and/or edges and the disappearance of existing nodes and/or edges over time. Typically, the dynamic evolution of network structures can be represented by a series of static networks: $G = \{G^1, G^2, \ldots, G^t, \ldots, G^T\}$, G^t can be interpreted as a snapshot captured at a specific moment in time t. While executing static network alignment methods directly on G or G^t can produce respectable top-α ($\alpha > 10$ generally) prediction accuracy, achieving satisfactory top-1 accuracy is often challenging. This is due to the following factors:

(1) Missing information. Static alignment methods usually have poor top-1 accuracy on snapshot network pairs due to snapshots only retaining part of the network information. As shown in Fig. 1(a), the consistency-based method will misalign v_i^1 and v_j^2 at the timestamps $t = t_2$ and t_3, because v_i^1 and v_j^2 share more anchor node pairs as neighbors between snapshot networks of G^1 and G^2 at $t = t_2$ and t_3.

(2) Confusing information. When ignoring the timing information of dynamic networks and merging all snapshot networks as a static network, directly applying static alignment methods may have poor top-1 accuracy due to confusing local structure. As shown in Fig. 1(b), it is confusing for v_i^1 in G^1 to judge whether its corresponding

Y. Wang and W. Wang are with the College of Intelligence and Computing, Tianjin University, Tianjin, 300350, China. Georgia Tech Shenzhen Institute, Tianjin University, Shenzhen 518055, China. Email: {wangyinghui, wjwang}@tju.edu.cn.
Q. Peng is with the School of New Media and Communication, Tianjin University, Tianjin, 300072, China. Email: qypeng@tju.edu.cn.
P. Jiao is with the School of Cyberspace, Hangzhou Dianzi University, Hangzhou, 310018, China. Email: pjiao@hdu.edu.cn.
H. Wu is with the Center for Applied Mathematics, Tianjin University, Tianjin, 300072, China. Email: whming@tju.edu.cn.
*(Corresponding author: Wenjun Wang. * is equal contribution.)*

Authorized licensed use limited to: TIANJIN UNIVERSITY. Downloaded on July 22,2024 at 05:17:01 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Big Data. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TBDATA.2024.3407543

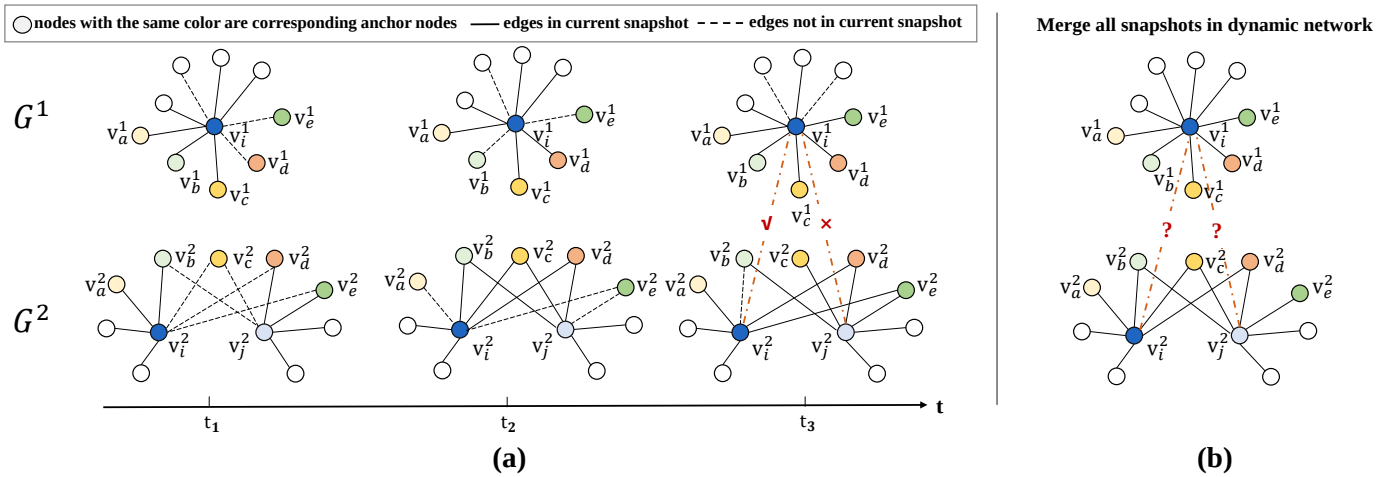

Fig. 1: Illustration of network alignment with/without dynamic evolution of network structures. Nodes with different colors are corresponding anchor nodes that are known in advance. (a) In the dynamic scenario, it can determine the corresponding anchor node of v_i^1 is v_i^2 at t_3 according to their similar neighborhood change pattern. (b) In the static scenario, it is difficult to determine the corresponding anchor node of v_i^1 since v_i^2 and v_j^2 both have a similar neighborhood with v_i^1.

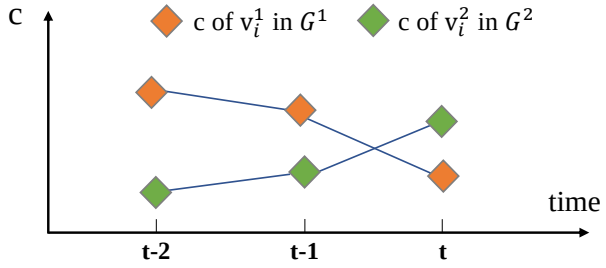

Fig. 2: Illustration of the global consistency of anchor node pairs. c is the number of the first-order neighbors of nodes. With the evolution of G^1 and G^2, the neighbor number of nodes v_i^1 and v_i^2 in G^1 and G^2 changes differently, but the total neighbor number of v_i^1 and v_i^2 in the two networks obeys a certain pattern.

anchor node is v_i^2 or v_j^2 in G^2. Since v_i^1 and v_i^2 have the same number of shared nodes as v_i^1 and v_j^2, and v_i^2 and v_j^2 have similar local structures, it is difficult to make a distinction between v_i^2 and v_j^2 without more auxiliary information.

In order to achieve better top-1 performance, it is crucial to implement a more effective strategy for differentiating neighboring nodes. The dynamic evolution of the network can provide more information. As shown in Fig. 1(a), v_i^1 and v_i^2 share anchor nodes at three timestamps (t_1, t_2, and t_3) in the networks G^1 and G^2. The shared anchor nodes change over time from (v_a^1, v_a^2), (v_b^1, v_b^2), (v_c^1, v_c^2) to (v_d^1, v_d^2), (v_e^1, v_e^2). Therefore, at $t = t_3$, it can be determined that the corresponding anchor node of v_i^1 is v_i^2 instead of v_j^2. Therefore, leveraging the evolution information of networks can effectively distinguish nodes within the network and consequently achieve superior network alignment performance.

To improve the top-1 performance of network alignment, we propose a novel alignment model called GLDyNA. This model takes into account not only the dynamic evolution of the network structure but also the consistent evolution of anchor node pairs in both global and local neighborhoods. For local consistency, its implication is similar to the common assumption in static network alignment methods, i.e., in each pair of snapshot networks, anchor node pairs need to maintain a similar local structure. For global consistency, it reveals the inherent evolution pattern of anchor node pairs over time and imposes constraints at a higher structural level in network alignment, i.e., it determines the number of neighbors an entity should maintain relationships with across different networks. Due to the evolution of network scale over time, there are fluctuations in the number of first-order neighbors, denoted as c, of anchor nodes across different networks, as shown in Fig. 2. Despite the dissimilar variations in the number of first-order neighbors of anchor nodes within networks G^1 and G^2, there exists a latent regularity in the total number of first-order neighbors of anchor nodes in these networks.

Specifically, to model the dynamics and preserve the network structure within a single network, we introduce a Structure-Time-aware module. This module randomly samples sub-networks of a given node v_i, which include the neighbors of v_i from different snapshots. It then learns representations of v_i by maximizing the probability of co-occurrence of two nodes within these sub-networks. In order to capture the global consistency of anchor node pairs, we define a linear function that relates to the total number of neighbors of anchor node pairs across different networks. This function imposes constraints on how the neighborhood scale of the anchor node pair evolves with the network. To capture the local consistency of anchor node pairs, we define a similarity equation based on structural equivalence. Finally, we combine the global and local consistency, along with the similarity of node representations learned from different networks, to predict anchor node pairs.

The main contributions of this paper could be summarized as follows:

Authorized licensed use limited to: TIANJIN UNIVERSITY. Downloaded on July 22,2024 at 05:17:01 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Big Data. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TBDATA.2024.3407543

- We analyze why static network alignment methods cannot be directly applied in dynamic scenarios. Additionally, we explain the factors contributing to the poor performance of existing methods in terms of top-1 precision.
- We propose a novel dynamic network alignment method GLDyNA, which takes into account both the influence of network evolution and the global-local consistency of anchor node pairs.
- We evaluate the proposed GLDyNA on different real-world datasets. Extensive experiments demonstrate the effectiveness of our method against state-of-the-art methods, especially in top-1 precision.

2 Related Work

Many fields can benefit from network alignment, such as user anonymous identification in social networks [11], comparing schemas between databases [25], linking entities among multiple knowledge graphs [26] and aligning proteins between species [27]. Research on static network alignment has seen much development, and numerous approaches have been proposed. In the past two years, researchers have also begun to focus on dynamic network alignment. According to different scenarios, we introduce existing network alignment methods in two parts: static network alignment and dynamic network alignment.

Static network alignment is mostly based on learning network structure and node attributes to judge whether two nodes represent the same entity. Recent network alignment methods mostly use network embedding [28], [29] since that can map the network structure to a low-dimensional space and is beneficial to maintain network structure and learned node representation could be used for comparing the similarity between nodes [13], [14], [15], [16], [17], [18], [19], [24], [30], [31], [32], [33], [34], [35], [36]. According to whether the two networks are merged into one network through known anchor nodes for representation learning, the existing methods can be roughly divided into the following two categories:

(1) The first category of alignment methods learns node embeddings in different networks respectively, then construct constraints [13], [14], [18], [35] or utilize adversarial learning [17], [19], [31], [34], [36] to make the embedding of the anchor nodes or the distributions of the two networks similar. The former is generally supervised utilizing known anchor nodes as constraint information. The latter is generally unsupervised, and most utilize generative adversarial networks to approximate the distribution of the two networks. For example, GINA [14] utilizes two different encoders to learn reliable spatial features of networks firstly, then uses anchor nodes to constrain learned node representations for following anchor link prediction. DANA [31] learns node embeddings via maximizing the posterior probability distribution of anchor nodes which is based on the parameter space of graph convolutional networks.

(2) The second category of alignment methods utilizes anchor nodes forming a unified space and then learns their embeddings for alignment [15], [16], [24], [32], [33]. For example, DHNA [32] learns node embeddings of different networks by a variational autoencoder in the same embedding space and uses a dual constraint mechanism to balance the consistency and heterogeneity in network alignment. BRIGHT [33] uses the one-hot vectors of anchor links to form the bases of common embedding space, and other initial embeddings of non-anchor nodes are obtained by a random walk with the restart. Then it uses a shared linear layer to train the weights of scores from different anchor links by keeping node embeddings of different networks in the same embedding space.

Although these static network alignment methods have demonstrated good performance, ignoring the temporal information of the network makes them unable to accurately model real-world scenarios to achieve more accurate performance, usually having poor top-1 precision.

Dynamic network alignment focuses more on how to use time information to improve the alignment effect in contrast to static network alignment. DNA [37] uses an LSTM encoder to learn evolvement neighborhood of nodes, and puts a consistency regularization onto the heart of the LSTM to keep the representation similarity with the neighbors of the node. DGA [38] expands based on DNA that uses an attentive graph convolution to model the structural information of nodes and the LSTM unit to incorporate the temporal evolvement pattern of nodes in the dynamic network. Unlike DNA and DGA which focus on the evolution of the entire network, HDyNA [39] only focuses on newly emerging nodes in the network. As a new node is added, its weights are learned heuristically, and then second-order proximity is preserved in updating the local network. CTSA [40] aligns the same entity across different snapshots in one dynamic network, which differs from our work that aligns snapshots in two different networks.

Compared with existing methods, our method not only focuses on the evolution of the local structure of nodes in the network, but also pays attention to the global-local evolution patterns of anchor node pairs and keeps its consistency changing over time, which is important for predicting potential anchor links.

3 Formal Definition

Referring to Fig.1(a), in dynamic scenarios, we focus solely on the structure of a time-stamped network. We partition the network into slices, as depicted in Fig.1(a), and construct a series of snapshots in the time domain. Each snapshot represents the network's characteristics in the corresponding time slice. Thus, a network with time-stamped can be defined as $G = \{G^1, G^2, \ldots, G^t, \ldots, G^T\}$. T is the number of snapshots. Each $G^t = (V^t, E^t)$ is an undirected and unweighted network snapshot at time t. V^t is the set of nodes and E^t is the set of edges at t. Considering that we align two time-stamped networks, we use superscripts to distinguish them, i.e., $G^1 = \{G^{1,1}, G^{1,2}, \ldots, G^{1,T}\}$ and $G^2 = \{G^{2,1}, G^{2,2}, \ldots, G^{2,T}\}$. In general, these two networks are partially overlapped by anchor nodes, which appear in both G^1 and G^2. Anchor nodes of network snapshot t are stored in set $\mathcal{A}^t = \left\{(v_i^{1,t}, v_j^{2,t})\right\}$. Each element in $\mathcal{A}^t$ represents an anchor link, i.e., the element $(v_i^{1,t}, v_j^{2,t})$ describes $v_i^{1,t}$ in $G^{1,t}$ corresponds to $v_j^{2,t}$ in $G^{2,t}$.

Authorized licensed use limited to: TIANJIN UNIVERSITY. Downloaded on July 22,2024 at 05:17:01 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Big Data. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TBDATA.2024.3407543

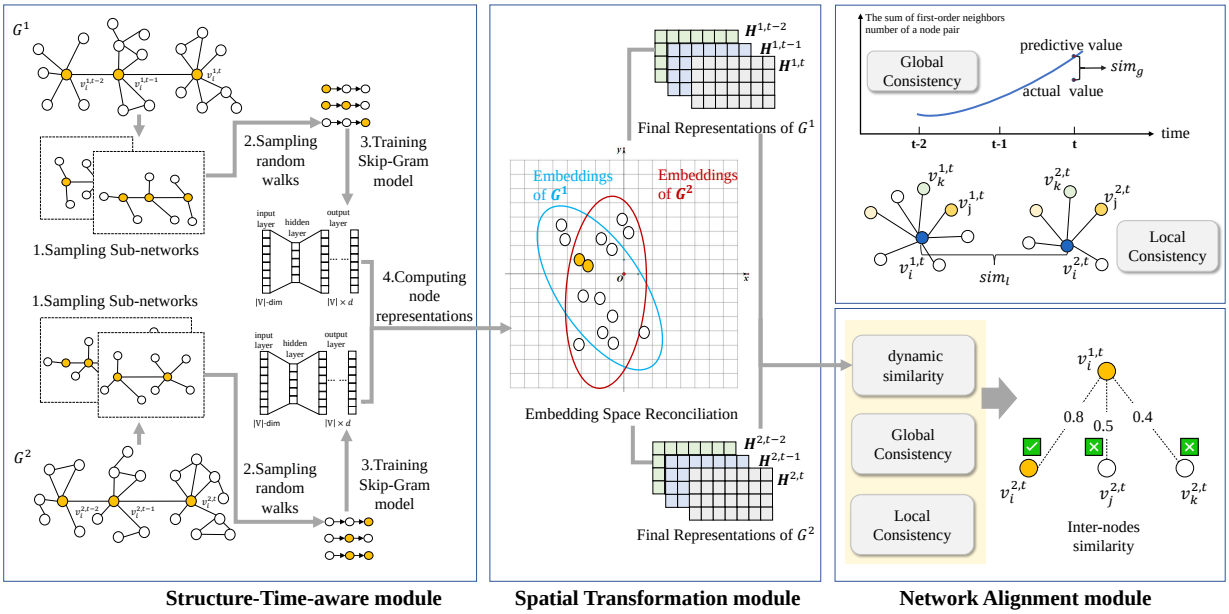

Fig. 3: The overview architecture of GLDyNA: For each dynamic network, the snapshots are sampled and trained to learn node representations. Then, the node representations of G^1 are reconciled with the embedding space of G^2's node representations. Finally, we perform network alignment based on dynamic similarity and Global-Local Consistency.

Dynamic Network Alignment. Given two partially overlapped networks G^1 and G^2 with anchor link set $\mathcal{A} = \{\mathcal{A}^1, \mathcal{A}^2, \ldots, \mathcal{A}^T\}$, the dynamic network alignment task is to find all potential anchor links in each network snapshot, i.e., we aim to learn a predictive function for each network snapshot: $f^t : (G^{1,t}, G^{2,t}, \mathcal{A}^t) \rightarrow Y$, Y_{ij}^t represents the probability that $v_i^{1,t}$ and $v_j^{2,t}$ are predicted to be a pair of potential anchor nodes across networks $G^{1,t}$ and $G^{2,t}$.

4 METHODOLOGY

In this section, we introduce the details of the proposed GLDyNA. As shown in Fig. 3, GLDyNA includes three modules, i.e., **Structure-Time-aware module**, **Spatial Transformation module**, and **Network Alignment module**. Through the Structure-Time-aware module, we first learn the node representations of the two networks separately so that we can capture information about the network structure and the temporal information for the following alignment. Then, we utilize a Spatial Transformation module to map the node representations of G^1 into the node representation space of G^2. The Network Alignment module performs network alignment based on the inter-nodes similarity measures, which include dynamic similarity and global-local consistency of nodes.

4.1 Structure-Time-aware module

In the Structure-Time-aware module, we design a method based on random walks to encode node representations that consider the changes in node behavior over time. As shown in Fig. 3, the behavior of a node (e.g., $v_i^{1,t}$) can be influenced by its nearby nodes in the current snapshot network as well as the preceding self-nodes (e.g., $v_i^{1,t-1}$ and $v_i^{1,t-2}$) and their first-level neighbors in previous snapshot networks. Therefore, we conduct the following steps to sample Spatial-Temporal sub-networks starting from the given node to learn the node representations:

For a given node $v_i^{1,t}$ in snapshot $G^{1,t}$, we consider the influence of the nodes in the current snapshot and the nodes of the previous τ historical snapshots on it. For each snapshot $G^{1,k}$, $k \in \{0, 1, \cdots, \tau\}$, we sample the first-order neighborhood nodes of $v_i^{1,t-k}$ and denote the sampled sub-network as $G_{sample}^{1,k}$. Then we merge $G^{1,t}$, all $v_i^{1,t-k}$ and all $G_{sample}^{1,k}$ to get the Spatial-Temporal sub-network $G_{ST}^{1,k}$ of $v_i^{1,t}$.

After obtaining the Spatial-Temporal sub-network $G_{ST}^{1,k}$ of $v_i^{1,t}$, the representations of nodes in $G_{ST}^{1,k}$ are learned by performing random walks on $G_{ST}^{1,k}$. Specifically, for each given node $v_i^{1,t}$ we generate a random walk sequence of length l and denote the sequence as $\mathcal{W}_{v_i^{1,t}}$. When two nodes have many edges or neighbors, they will be visited more frequently during random walks, showing that their network structures are similar. As a result, representations of these two nodes in the embedding space should be close to each other. Given a random walk sequence $\mathcal{W}_{v_i^{1,t}}$, it is now possible to formulate learning spatial-temporal-preserving node embeddings as an optimization problem as follows:

$$\max_{\boldsymbol{h}} \sum_{v_j \in \mathcal{W}_{v_i^{1,t}}} \log \Pr(\mathcal{W}_w = \{v_{j-w}, \cdots, v_{j+w}\} \setminus v_j \mid \boldsymbol{h}(v_j)), \tag{1}$$

where $\boldsymbol{h} : V \rightarrow \mathcal{R}^d$ is the embedding function that maps a given node to a d-dimensional representation, $\boldsymbol{h}(v_j)$ is the representation of v_j. w is the context window size for optimization. We assume conditional independence between the nodes of a context window when observed with respect

Authorized licensed use limited to: TIANJIN UNIVERSITY. Downloaded on July 22,2024 at 05:17:01 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Big Data. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TBDATA.2024.3407543

to the source node v_j:

$$\Pr(\mathcal{W}_w = \{v_{j-w}, \cdots, v_{j+w}\} \setminus v_j \mid \boldsymbol{h}(v_j)) = \prod_{v_k \in \mathcal{W}_w} \Pr(v_k \mid \boldsymbol{h}(v_j)). \quad (2)$$

The probability could be calculated as follows:

$$\Pr(v_k \mid \boldsymbol{h}(v_j)) = \frac{\exp(\boldsymbol{h}(v_k)\boldsymbol{h}(v_j))}{\sum_{v_n \in \mathcal{N}(v_j)} \exp(\boldsymbol{h}(v_n)\boldsymbol{h}(v_j))}, \quad (3)$$

where $\mathcal{N}(v_j)$ represents the neighbors set of node v_j. To learn such a representation that captures the relationship of a node with other co-occurring nodes in a window, we use a similar Skip-Gram algorithm as proposed in [41] to learn the node representations from random walks of a network.

Therefore, through Eq(1) which maximizes the co-occurrence of neighborhood nodes at snapshot t and in previous $(t-\tau)$ snapshots together, the Structure-Time-aware module learns node representations in G^1 and G^2 respectively. We denote $\boldsymbol{h}^1$ as the representations of nodes in G^1, and $\boldsymbol{h}^2$ as the representations of nodes in G^2.

4.2 Spatial Transformation module

Across different networks, the same nodes may exhibit varying characteristics and interaction relationships due to semantic distinctions of networks. As a consequence, the embedding spaces of node representations differ, making it infeasible to directly utilize the learned representations for similarity measurement of nodes and network alignment. Therefore, we propose a method to reconcile the embedding spaces of networks G^1 and G^2. To accomplish nonlinear spatial transformations, we employ a feed-forward neural network to map node representations from G^1 to the embedding space of node representations in G^2:

$$\phi(\boldsymbol{h}|\boldsymbol{W}_1, \boldsymbol{W}_2, \boldsymbol{b}) = \sigma(\boldsymbol{h}\boldsymbol{W}_1 + \boldsymbol{b})\boldsymbol{W}_2, \quad (4)$$

where $\sigma(\cdot)$ is an activation function and we use the Sigmoid function in this paper. $\boldsymbol{W}_1$, $\boldsymbol{W}_2$, and $\boldsymbol{b}$ are trainable parameters.

In the reconciled embedding space, the representations of the same entity (i.e., anchor nodes) should be as similar as possible, and even be the same ideally. Therefore, we use labeled anchor links to constrain the representation of anchor nodes and conduct the training of the above-mentioned feed-forward neural network to obtain the ideal spatial transformation function $\phi(\cdot)$:

$$\mathcal{O}_\phi = \sum o_i, \quad (5)$$

$$o_i = \begin{cases} 1 - \theta\left(\phi\left(\boldsymbol{h}^1(v_i^1)\right), \boldsymbol{h}^2(v_j^2)\right) & (v_i^1, v_J^2) \in \mathcal{A} \\ max\left(0, \theta\left(\phi\left(\boldsymbol{h}^1(v_i^1)\right), \boldsymbol{h}^2(v_j^2)\right) - \varepsilon\right) & (v_i^1, v_J^2) \notin \mathcal{A}, \end{cases} \quad (6)$$

where $\theta(\cdot,\cdot)$ is the cosine value of two node representations, ε is a hyperparameter, and we adopt SGD [42] to minimize Eq(5). In this way, the representations of the corresponding anchor nodes are almost identical. Meanwhile, the representations of other non-anchor nodes in the two networks can maintain their own structural features.

4.3 Network Alignment module

The Network Alignment module includes three parts:

Dynamic similarity. With the node representation $\phi\left(\boldsymbol{h}^1\right)$ and $\boldsymbol{h}^2$ that in the same embedding space, a direct way of determining the alignments for a node is to calculate pairings of similarity between the representations that contain dynamics, i.e., nodes dynamic similarity:

$$sim_\theta(v_i^{1,t}, v_j^{2,t}) = \theta(\phi\left(\boldsymbol{h}^1(v_i^1)\right), \boldsymbol{h}^2(v_j^2)). \quad (7)$$

As we introduced in Section 4.1, the Structure-Time-aware module focuses on the dynamic evolution of nodes in one network over time, sim_θ reflects the evolution similarity of two nodes in their respective networks. The larger the sim_θ value, the higher the probability that v_i^1 and v_j^2 are corresponding anchor nodes.

When aligning nodes between different networks, in addition to considering the evolution process of nodes in different networks, we also consider patterns (global and local) between node pairs over time.

Global consistency. At the global level, for the dynamic network, its scale usually evolves with obvious distributions over time at the global level, such as a sigmoid curve [43] or a power-law distribution [44]. For each anchor node pair, there also exists a similar pattern. As shown in Fig. 2, when the network evolves, the first-order neighbors number c_i^1 and c_i^2 of an anchor node pair (v_i^1, v_i^2) vary over time in different networks respectively, and their sum (i.e., c_i^1+c_i^2) obeys a certain underlying pattern, such as linear increasing. Such a global evolutionary pattern is common in real network alignment scenarios. For example, a scholar who collaborates with other scholars to publish papers in journals and conferences, respectively, will have different propensities in the journal and conference collaboration over time, but the total number of people he keeps collaborating with generally remains the same or even increases. Hence, we use a linear neural network to learn the sum collaboration (i.e., first-order neighbors number) of anchor node pairs at snapshot t as shown in the top rightmost side of Fig. 3:

$$\psi(c_p^t|\boldsymbol{W}_g, \boldsymbol{b}) = \boldsymbol{c}\boldsymbol{W}_g + \boldsymbol{b}, \quad (8)$$

where $\boldsymbol{c} = \left\{c^0, c^1, \cdots, c^{t-1}\right\}$ is a vector composed of the sum of first-order neighbors number of an anchor node pair before snapshot t. $\boldsymbol{W}_g$ and $\boldsymbol{b}$ are trainable parameters. c_p^t is the predicted sum first-order neighbors number of an anchor node pair at snapshot t.

Thus, for an arbitrary node pair (v_i^1, v_j^2), we use the difference between the predicted sum first-order neighbors number $c_p^t(v_i^1, v_j^2)$ and the true sum first-order neighbors number $c^t(v_i^1, v_j^2)$ at snapshot t as the global consistency measure to determine the probability that two nodes are the corresponding anchor nodes at snapshot t:

$$sim_g(v_i^{1,t}, v_j^{2,t}) = \exp^{-\left|c_p^t(v_i^1, v_j^2) - c^t(v_i^1, v_j^2)\right|}. \quad (9)$$

The larger the sim_g value, the higher the probability that v_i^1 and v_j^2 are corresponding anchor nodes.

Local consistency. At the local level, nodes in networks usually exist homophily equivalence [45]. Homophily equivalence refers to the phenomenon in which adjacent nodes in a network display similar characteristics or attributes. This concept bears resemblance to the main idea

Authorized licensed use limited to: TIANJIN UNIVERSITY. Downloaded on July 22,2024 at 05:17:01 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Big Data. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TBDATA.2024.3407543

TABLE 1: Statistics of the datasets.

Dataset		Networks						
		Twitter		Foursquare		Anchor Links		
TF	Snapshots	# nodes	# edges	# nodes	# edges	# total	# added	# disappeared
	t_0	4,709	107,528	4,836	54,586	596	-	-
	t_1	4,809	124,585	4,936	60,330	854	440	182
	t_2	4,909	140,338	5,036	67,044	987	312	179
	t_3	5,009	152,434	5,136	72,732	1,078	207	116
	t_4	5,109	164,936	5,236	76,874	1,282	204	0
		journal-paper cooperation		conference-paper cooperation		Anchor Links		
JC	Snapshots	# nodes	# edges	# nodes	# edges	# total	# added	# disappeared
	t_0	2,832	11,164	4,343	15,354	1,013	-	-
	t_1	2,997	11,900	4,755	24,055	1,036	570	540
	t_2	3,259	11,911	5,203	26,216	1,141	684	579
	t_3	3,708	12,658	6,333	44,099	1,352	827	616
	t_4	4,153	16,467	6,343	48,965	1,359	741	734
	t_5	4,687	20,851	6,573	41,612	1,401	773	731
	t_6	5,885	27,200	7,134	41,580	1,799	1,099	701

behind the Word2Vec method in natural language processing, where words that frequently co-occur are likely to possess similar meanings or representations. As shown in the top rightmost side of Fig. 3, according to the homophily equivalence hypothesis, nodes $v_i^{1,t}$, $v_j^{1,t}$, and $v_k^{1,t}$ exhibit greater similarity in the embedding space of $G^{1,t}$, and the same for $v_i^{2,t}$, $v_j^{2,t}$, and $v_k^{2,t}$ in the embedding space of $G^{2,t}$. The existence of significant similarities among nodes $v_i^{1,t}$, $v_j^{1,t}$, and $v_k^{1,t}$ in a network can result in confusion when comparing node $v_i^{2,t}$ with node $v_i^{1,t}$ by calculating nodes dynamic similarity sim_θ. Furthermore, this can lead to incorrect alignment of node $v_i^{2,t}$ with nodes $v_j^{1,t}$ or $v_k^{1,t}$. Inspired by the principle of structural equivalence [46], i.e., if two nodes share many common neighbors in the network, then they are structural equivalence, we define the following measurement to calculate the local similarity between two nodes across networks to alleviate the impact of node homophily:

$$sim_l(v_i^{1,t}, v_j^{2,t}) = \sum_{\varsigma}^{t} \frac{\log\left(\left|\mathcal{N}(v_i^{1,t}) \cap \mathcal{N}(v_j^{2,t})\right| + 1\right)}{(t - \varsigma + 1)}, \quad (10)$$

where $\mathcal{N}(v_i^{1,t})$ and $\mathcal{N}(v_j^{2,t})$ denote the set of all the first-order neighbour of node $v_i^{1,t}$ and $v_j^{2,t}$, respectively. $\left|\mathcal{N}(v_i^{1,t}) \cap \mathcal{N}(v_j^{2,t})\right|$ is the number of known pairs of anchor nodes existing in their neighborhood. ς is the hyperparameter that determines the number of previous snapshots, that GLDyNA considers for up to $(t - \varsigma + 1)$ local neighborhood information when enforcing local consistency constraints. The larger the sim_l value, the higher the probability that v_i^1 and v_j^2 are corresponding anchor nodes.

Finally, we compare the similarity of cross-network nodes based on the aforementioned introduced dynamic similarity and global-local consistency of nodes:

$$sim(v_i^{1,t}, v_j^{2,t}) = sim_\theta(v_i^{1,t}, v_j^{2,t}) + \lambda * sim_g(v_i^{1,t}, v_j^{2,t}) + \gamma * sim_l(v_i^{1,t}, v_j^{2,t}), \quad (11)$$

where λ and γ are the weight of global consistency and local consistency respectively. Based on $sim(\cdot, \cdot)$, we obtain pairwise similarities Y_{ij}^t between nodes to be aligned in two networks and sort them according to their similarity scores. Since not all nodes have corresponding nodes in the other network, we set a threshold ε. When $sim(\cdot, \cdot) \geq \varepsilon$, we consider the node pair with the highest $sim(\cdot, \cdot)$ value as a potential anchor node pair.

4.4 Time Complexity

The time complexity of GLDyNA primarily lies in the node representation learning process. GLDyNA involves sampling and generating Spatial-Temporal sub-networks for each node, which has a time complexity of $O(NM)$. Here, N represents the total number of nodes across all snapshots in the network, and M denotes the average number of first-order neighbors for these nodes. Additionally, conducting random walks on the generated Spatial-Temporal sub-networks incurs a time complexity of $O(nl)$, where l is the length of the random walk, and n is the number of walk iterations. Moreover, the time complexity of model training based on the obtained sequences is $O(m)$, where m represents the number of training iterations. Therefore, the overall time complexity of GLDyNA can be expressed as $O(NM + nl + m) = O(NM)$.

5 Experiments

This section introduces the datasets and describes the set of experiments conducted to validate the proposed GLDyNA. Additionally, we analyze the validity of each component of the model and the influence of model parameters.

5.1 Experimental Settings

5.1.1 Datasets

The following two datasets are used to verify the effectiveness of GLDyNA.

- Social Networks: This data set includes users and their followers from Twitter and Foursquare (TF), respectively. The snapshots are at equal intervals

Authorized licensed use limited to: TIANJIN UNIVERSITY. Downloaded on July 22,2024 at 05:17:01 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Big Data. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TBDATA.2024.3407543

of the network, and there exist new nodes in each snapshot [39].
- Academic Networks: This data set includes researchers and their collaborators from DBLP. Depending on the publication channels of researchers' papers, academic networks are divided into journal-paper cooperation networks and conference-paper cooperation networks (JC). And each snapshot represents a year.

Details are illustrated in Table 1. We have expanded based on the above two datasets, referred to as TF+ and JC+. The extension rule is that except for the snapshot network at t_0, all other snapshot networks are merged by themselves, along with all snapshots at all previous times.

5.1.2 Baseline Methods

The proposed method is compared with the seven state-of-the-art methods listed below.

- BRIGHT [33]: a static network alignment method that creates a space by RWR whose bases are anchor node encoding vectors, followed by a shared linear layer to learn node representations.
- NetTrans [30]: a static network alignment method that uses graph convolutional network to learn node representations at different resolutions for alignment from the network transformation view.
- DANA [31]: a static network alignment method that uses GCN to learn node embeddings and train an adversarial domain classifier supervised by the anchor nodes to obtain domain-invariant features for alignment.
- NeXtAlign [24]: a static network alignment method that uses a special graph convolutional network to balance the consistency and disparity in alignment through the learning process.
- DHNA [32]: a static network alignment method that uses a variational autoencoder to learn node embeddings, and considers the different anchor nodes' degrees across networks.
- DGA [38]: a dynamic network alignment method that uses a dynamic graph autoencoder to learn user embeddings in each network, and constructs a common subspace for user alignment across different networks.
- HDyNA [39]: a dynamic network alignment method that learns the local influence weight of new nodes in a single network environment using an attention mechanism and anchor nodes are used as supervised information.

5.1.3 Evaluation Metrics

For each matching pair $(v_i^{1,t}, v_j^{2,t})$ in the test set, we rank the target nodes in the result according to Y_{ij}^t. To quantify the ranking at snapshot t, we use the two evaluation metrics which are commonly used in network alignment tasks.

- $Precision = \frac{|M_t@1|}{|U_t|}$ indicates whether the true positive match occurs in top-1 candidates, where $|M_t@1|$ is the count of the correct alignments between networks $G^{1,t}$ and $G^{2,t}$ in top-1 choices, and $|U_t|$ is the number of anchor links in the train set.
- $MRR = \frac{1}{|U_t|}\sum_{(v_i^{1,t}, v_j^{2,t})\in\mathcal{T}} \frac{1}{rank(v_j^{2,t})}$, where $rank(v_j^{2,t})$ is the rank of true anchor target in the sorted list of anchor candidates. $\mathcal{T}$ is the test set that includes correct alignments between $G^{1,t}$ and $G^{2,t}$.

5.1.4 Implementation Details

To create our training and testing datasets, we randomly partitioned the anchor nodes into two sets. The ratio of the number of anchor nodes in the training set to that in the testing set was 4:1, with the specific numbers randomly sampled. For a fair comparison, hyper-parameters except for node embedding dimension are set to default for all baselines. We set the hyper-parameters of GLDyNA as follows unless otherwise specified:

- For the Structure-Time-aware module, We set the number of historical snapshots considered during sampling $\tau = 1$, and the random walk length $l = 15$, the node representation dimension $d = 64$.
- For the Network Alignment module, we set the $\varsigma = 0$ in local consistency measurement, i.e., we consider the local consistency of all previous snapshots. We set the weight of global consistency $\lambda = 0.15$ and are the weight of local consistency $\gamma = 0.1$. The threshold ε for aligning potential anchor node pairs is set as the average similarity of anchor node pairs in the training set.

The experimental environment uses Python 3.7 languages as the basic development language, and GLDyNA is implemented based on the open-source Pytorch framework. Experiments are performed on a workstation equipped with NVIDIA RTX 1080Ti 20GB video memory. In each experiment, we repeated it 10 times and reported the mean with a 95% confidence interval.

5.2 Model Performance Analysis

Precision Improvement. We first compare GLDyNA with all baselines in four datasets, and results are reported in Table 2 and Table 3. Results show that the proposed GLDyNA mostly outperforms the baselines on both $Precision$ and MRR. On $Precision$, GLDyNA achieves the best performance on all snapshot networks of four datasets, improving by at least $9.97\%, 4.17\%$ compared with the best competitors on dataset TF (TF+), and JC (JC+) respectively. On MRR, GLDyNA achieves the best performance on all snapshot networks except for t_2 snapshot of JC and JC+, improving by an average 17.63%, 5.65% compared with the best competitors on dataset TF (TF+), and JC (JC+) respectively.

Compared with the static network alignment method, GLDyNA has achieved a very significant improvement in alignment accuracy, which demonstrates the effectiveness of considering the evolution characteristics of nodes over time. The suboptimal performance of these static methods on JC and TF datasets suggests that relying solely on one snapshot for alignment may overlook valuable information. When temporal information is disregarded and multiple snapshots are merged, as in the JC+ and TF+ datasets, the performance of static methods does not improve and some

Authorized licensed use limited to: TIANJIN UNIVERSITY. Downloaded on July 22,2024 at 05:17:01 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Big Data. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TBDATA.2024.3407543

TABLE 2: Experimental results on TF and TF+ datasets at different snapshot t. The best and second-best results are highlighted in boldface and underlined, respectively. ▲% denotes the improvement of GLDyNA compared to the best baseline methods results.

		t_1		t_2		t_3		t_4	
		$Precision$	MRR	$Precision$	MRR	$Precision$	MRR	$Precision$	MRR
TF	BRIGHT	0.1111	0.1779	0.1515	0.2036	0.1481	0.2127	0.1167	0.1819
	NetTrans	0.1462	0.1935	0.1263	0.1764	0.1203	0.1710	0.1323	0.1926
	DANA	0.1521	0.2187	0.1313	0.1897	0.1412	0.2188	0.1498	0.2373
	NeXtAlign	0.0702	0.1272	0.0808	0.1321	0.0694	0.1256	0.0739	0.1344
	DHNA	0.1022	0.1100	0.1154	0.1327	0.0912	0.1035	0.1159	0.1677
	DGA	0.6233	0.6419	0.6800	0.6921	0.7301	0.7622	0.8117	0.8337
	HDyNA	0.5246	0.5721	0.6311	0.6871	0.7200	0.7596	0.7657	0.8003
	GLDyNA	**0.7045**	**0.7059**	**0.8871**	**0.8872**	**0.9024**	**0.9025**	**0.9500**	**0.9500**
	▲%	13.03	9.97	30.46	28.19	23.60	18.41	17.04	13.95
TF+	BRIGHT	0.0936	0.1543	0.1060	0.1527	0.1574	0.2263	0.1712	0.2393
	NetTrans	0.1403	0.1871	0.1212	0.1733	0.1157	0.1684	0.1361	0.1947
	DANA	0.1462	0.2173	0.1288	0.1932	0.1412	0.2221	0.1537	0.2386
	NeXtAlign	0.0819	0.1297	0.0657	0.1285	0.0648	0.1265	0.0895	0.1253
	DHNA	0.1092	0.1156	0.1054	0.1335	0.1029	0.1147	0.1377	0.1691
	DGA	0.6233	0.6419	0.6800	0.6921	0.7301	0.7622	0.8117	0.8337
	HDyNA	0.5246	0.5721	0.6311	0.6871	0.7200	0.7596	0.7657	0.8003
	GLDyNA	**0.7045**	**0.7059**	**0.8871**	**0.8872**	**0.9024**	**0.9025**	**0.9500**	**0.9500**
	▲%	13.03	9.97	30.46	28.19	23.60	18.41	17.04	13.95

TABLE 3: Experimental results on JC and JC+ datasets at different snapshot t. The best and second-best results are highlighted in boldface and underlined, respectively. ▲% denotes denotes the improvement of GLDyNA compared to the best baseline methods results.

		t_1		t_2		t_3		t_4		t_5		t_6	
		$Precision$	MRR	$Precision$	MRR	$Precision$	MRR	$Precision$	MRR	$Precision$	MRR	$Precision$	MRR
JC	BRIGHT	0.2836	0.4115	0.2794	0.4247	0.2693	0.4018	0.3014	0.4152	0.2419	0.3898	0.2611	0.3921
	NetTrans	0.1352	0.1818	0.1271	0.1575	0.1037	0.1404	0.0774	0.1185	0.1142	0.1697	0.0612	0.1025
	DANA	0.4589	0.5497	0.4276	0.4691	0.3963	0.4566	0.3856	0.4568	0.3732	0.4468	0.3481	0.4177
	NeXtAlign	0.3188	0.4310	0.2456	0.3514	0.2815	0.3795	0.2583	0.3487	0.2500	0.3833	0.2256	0.2882
	DHNA	0.2907	0.3011	0.3270	0.3609	0.2571	0.2834	0.2279	0.2630	0.2800	0.3107	0.2112	0.2971
	DGA	0.5022	0.5500	0.4729	**0.5273**	0.5000	0.5388	0.4992	0.5236	0.5235	0.5700	0.6122	0.6503
	HDyNA	0.4304	0.4972	0.4641	0.5210	0.4090	0.5319	0.4175	0.5100	0.5010	0.5541	0.5399	0.6110
	GLDyNA	**0.5702**	**0.5743**	**0.4926**	0.4969	**0.6126**	**0.6070**	**0.5608**	**0.5605**	**0.5974**	**0.6100**	**0.7078**	**0.7057**
	▲%	13.54	4.42	4.17	-5.77	22.52	12.66	12.34	7.05	14.12	7.02	15.62	8.52
JC+	BRIGHT	0.2711	0.3900	0.2807	0.4166	0.2702	0.4110	0.3158	0.4252	0.2700	0.3961	0.2600	0.3851
	NetTrans	0.1357	0.1899	0.1193	0.1496	0.1201	0.1370	0.1022	0.1257	0.1100	0.1636	0.0895	0.1103
	DANA	0.4402	0.5152	0.4270	0.4672	0.4117	0.4983	0.4000	0.4794	0.4457	0.5029	0.3665	0.4570
	NeXtAlign	0.2901	0.3510	0.2500	0.3766	0.3011	0.4067	0.2594	0.3499	0.3143	0.4402	0.3147	0.3800
	DHNA	0.2801	0.2900	0.3206	0.3551	0.2764	0.3004	0.2300	0.2719	0.2807	0.3233	0.2410	0.3306
	DGA	0.5022	0.5500	0.4729	**0.5273**	0.5000	0.5388	0.4992	0.5236	0.5235	0.5700	0.6122	0.6503
	HDyNA	0.4304	0.4972	0.4641	0.5210	0.4090	0.5319	0.4175	0.5100	0.5010	0.5541	0.5399	0.6110
	GLDyNA	**0.5702**	**0.5743**	**0.4926**	0.4969	**0.6126**	**0.6070**	**0.5608**	**0.5605**	**0.5974**	**0.6100**	**0.7078**	**0.7057**
	▲%	13.54	4.42	4.17	-5.77	22.52	12.66	12.34	7.05	14.12	7.02	15.62	8.52

even degrade. This highlights the detrimental impact of disregarding the evolution of the nodes on alignment, as confusing information may introduce additional confusion into the alignment process, as discussed in Section 1.

Compared to the dynamic network alignment method HDyNA and DGA, GLDyNA still demonstrates better alignment performance. HDyNA solely considers the scenario where new nodes are added to the evolving network over time while disregarding the situation where certain nodes may also vanish over time. As a result, it fails to roundly capture the temporal dynamics of node features and consequently impairs the accuracy of network alignment. DGA utilizes graph attention convolutional units and an LSTM-based encoder to learn representations that capture the dynamic information within nodes from two networks individually. It then aligns the embedded spaces of the two networks by mapping them to a shared subspace. The mechanism employed for learning node representations requires aligned nodes to exhibit similar neighbor evolution characteristics. However, the effectiveness of DGA diminishes when nodes demonstrate divergent evolutionary behaviors. In such scenarios, our method alleviates the challenges posed by limited local consistency in alignment by incorporating global consistency, thereby maintaining a favorable alignment performance. While GLDyNA has achieved promising $Precision$ and MRR alignment results overall, it falls slightly behind DGA in certain snapshot networks. This could be attributed to an imbalanced distribution of anchor nodes in those specific snapshots, where the global consistency negatively affects the alignment performance in those cases.

In addition to evaluating the top-1 accuracy of alignment results, we also compared the alignment accuracy of each method at top-α ($\alpha > 1$) levels. The results, as shown in Fig. 4, indicate that as α increases, the accuracy of all comparative methods improves. However, our method

Authorized licensed use limited to: TIANJIN UNIVERSITY. Downloaded on July 22,2024 at 05:17:01 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Big Data. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TBDATA.2024.3407543

TABLE 4: Experimental results on TF and JC datasets with different anchor node percentage at last snapshot of each dataset (i.e., snapshot 4 of TF and snapshot 6 of JC).

		0.5		0.6		0.7		0.8	
		Precision	*MRR*	*Precision*	*MRR*	*Precision*	*MRR*	*Precision*	*MRR*
TF	BRIGHT	0.0874	0.1498	0.1150	0.1816	0.1351	0.1966	0.1167	0.1819
	NetTrans	0.1310	0.1853	0.1306	0.1931	0.1377	0.1926	0.1323	0.1926
	DANA	0.0926	0.1504	0.1131	0.1799	0.1286	0.1980	0.1498	0.2373
	NeXtAlign	0.0562	0.1180	0.0897	0.1282	0.0857	0.1196	0.0739	0.1344
	DHNA	0.0925	0.1137	0.1009	0.1286	0.1143	0.1602	0.1159	0.1677
	DGA	0.6800	0.6904	0.7101	0.7400	0.7581	0.7720	0.8117	0.8337
	HDyNA	0.6533	0.6894	0.7003	0.7220	0.7129	0.7466	0.7257	0.7503
	GLDyNA	**0.8431**	**0.8431**	**0.7901**	**0.7901**	**0.8525**	**0.8525**	**0.9500**	**0.9500**
JC	BRIGHT	0.1722	0.2805	0.1875	0.3021	0.2241	0.3364	0.2611	0.3921
	NetTrans	0.0690	0.1049	0.0737	0.1062	0.0761	0.1131	0.0612	0.1025
	DANA	0.3054	0.3296	0.3255	0.3693	0.3516	0.4092	0.3481	0.4177
	NeXtAlign	0.1580	0.2577	0.1892	0.2894	0.2171	0.3036	0.2256	0.2882
	DHNA	0.1997	0.2234	0.2217	0.2534	0.2550	0.2796	0.2112	0.2971
	DGA	0.5733	0.5891	0.5900	0.6092	0.6205	0.6331	0.6122	0.6503
	HDyNA	0.5022	0.5571	0.5430	0.5756	0.5402	0.6018	0.5399	0.6110
	GLDyNA	**0.6266**	**0.6259**	**0.6583**	**0.6783**	**0.6778**	**0.6783**	**0.7078**	**0.7057**

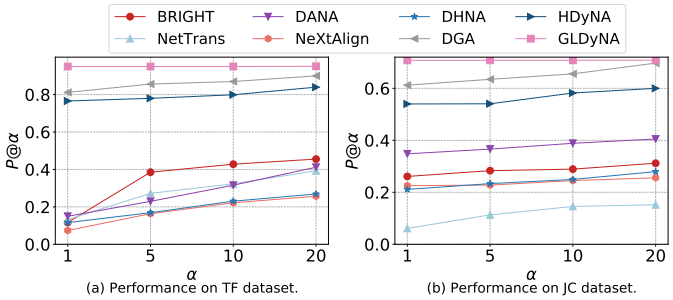

Fig. 4: Experimental results on TF and JC datasets with different top-α metrics at last snapshot of each dataset (i.e., snapshot 4 of TF and snapshot 6 of JC).

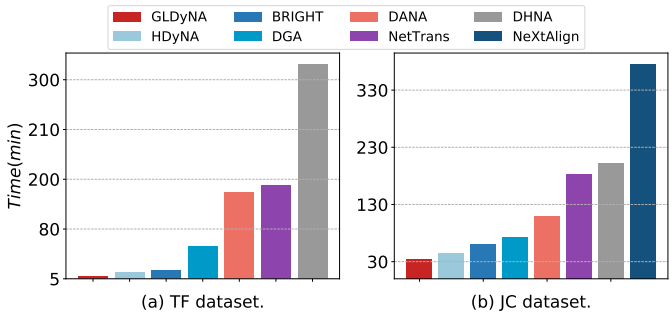

Fig. 5: Model running times on TF and JC datasets.

consistently maintains a stable performance. This observation suggests that our approach effectively distinguishes correctly aligned nodes from others, whereas other methods struggle to make clear differentiations, resulting in lower top-1 accuracy but relatively better top-α ($\alpha > 1$) accuracy.

Effect of Anchor Node Percentage. Based on previous methods, the more anchor nodes in the training set, the better the network alignment performance. We analyze the impact of anchor node percentage in the training set from 0.5 to 0.8. As shown in Table 4, the effectiveness of all methods increases with an increasing proportion of anchor nodes in the training set. We observe that GLDyNA outperforms other methods even when the proportion of anchor nodes in the training set is low. This result is due to its consideration of dynamic changes in node behaviors, which increases the separability of candidate node pairs, and its incorporation of global consistency, which excludes candidate node pairs that do not conform to the overall evolutionary pattern.

Time for Searching Anchor Node Pairs. In Fig. 5, we compare the computational efficiency of each method. The running time of most methods is comparable, except for NeXtAlign. Despite NeXtAlign achieving satisfactory alignment results, it utilizes a complex negative sampling method to calculate node attention, resulting in a longer running time. Compared to static methods, although GLDyNA considers information from different snapshots, it reduces its running time through a sampling approach.

5.3 Ablation Study

In this subsection, we conduct ablation studies to validate the effectiveness of global-level and local-level consistency. Six variants are designed:

- GLDyNA-WG does not consider the global consistency of node pairs, i.e., performs alignment without $sim_g(\cdot,\cdot)$.
- GLDyNA-WG(L-) does not consider global consistency of node pairs and only considers local consistency between the current snapshot and the previous snapshot, i.e., performs alignment without $sim_g(\cdot,\cdot)$ and set $\varsigma = t-1$ in $sim_l(\cdot,\cdot)$.
- GLDyNA-WL does not consider local consistency of node pairs, i.e., performs alignment without $sim_l(\cdot,\cdot)$.
- GLDyNA-WGL does not consider global and local consistency of node pairs, i.e., performs network alignment without $sim_g(\cdot,\cdot)$ and $sim_l(\cdot,\cdot)$.
- GLDyNA-WS does not perform spatial transformation, i.e., uses the node representations learned by Structure-Time-aware module directly.
- GLDyNA-G uses a nonlinear neural network to learn the sum collaboration of anchor node pairs as a replacement for Eq(8).

Authorized licensed use limited to: TIANJIN UNIVERSITY. Downloaded on July 22,2024 at 05:17:01 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Big Data. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TBDATA.2024.3407543

TABLE 5: Results of ablation study on TF dataset at different snapshot t.

	t_1		t_2		t_3		t_4	
	$Precision$	MRR	$Precision$	MRR	$Precision$	MRR	$Precision$	MRR
GLDyNA-WG	0.3750	0.3628	0.3549	0.3855	0.3283	0.3244	0.3000	0.2910
GLDyNA-WG(L-)	0.3295	0.3045	0.3871	0.3845	0.2683	0.3193	0.2750	0.2546
GLDyNA-WL	0.6959	0.6906	0.8387	0.8395	0.8926	0.8926	0.9250	0.9250
GLDyNA-WGL	0.2455	0.2658	0.2806	0.2837	0.2732	0.2915	0.3250	0.3303
GLDyNA-WS	0.2219	0.2370	0.2511	0.2466	0.2501	0.2422	0.2991	0.3009
GLDyNA-G	0.6992	0.7059	0.8822	**0.8875**	0.8906	0.9000	0.9436	0.9461
GLDyNA	**0.7045**	**0.7059**	**0.8871**	0.8872	**0.9024**	**0.9025**	**0.9500**	**0.9500**

TABLE 6: Results of ablation study on JC dataset at different snapshot t.

	t_1		t_2		t_3		t_4		t_5		t_6	
	$Precision$	MRR	$Precision$	MRR	$Precision$	MRR	$Precision$	MRR	$Precision$	MRR	$Precision$	MRR
GLDyNA-WG	0.5263	0.4827	0.4820	0.4704	0.4364	0.4530	0.3649	0.3825	0.4091	0.3986	0.4373	0.4293
GLDyNA-WG(L-)	0.5251	0.5206	0.4706	0.4470	0.4606	0.4483	0.3649	0.4034	0.4156	0.4351	0.3470	0.3460
GLDyNA-WL	0.5263	0.5377	0.4368	0.4398	0.5333	0.5382	0.5149	0.5180	0.5404	0.5379	0.6758	0.6709
GLDyNA-WGL	0.4439	0.4670	0.4515	0.4703	0.4364	0.4562	0.4338	0.4446	0.4519	0.4626	0.4046	0.4144
GLDyNA-WS	0.4011	0.4318	0.4366	0.4419	0.4052	0.4338	0.4216	0.4288	0.4361	0.4423	0.4000	0.3903
GLDyNA-G	0.5701	0.5699	0.4871	0.4799	0.6112	0.5973	0.5603	0.5500	0.5900	0.5927	0.7000	0.6977
GLDyNA	**0.5702**	**0.5743**	**0.4926**	**0.4969**	**0.6126**	**0.6070**	**0.5608**	**0.5605**	**0.5974**	**0.6100**	**0.7078**	**0.7057**

Table 5 and Table 6 compare the different variants of GLDyNA on TF and JC datasets, respectively. Global and local consistency plays a crucial role in network alignment, which is demonstrated by the significant drop in the performance of GLDyNA-WGL. The superior performance of GLDyNA-WGL compared to most static network alignment baseline methods indicates that the consideration of dynamic node behaviors is beneficial for network alignment.

The advantage of our global consistency can be quantified by the reduced performance of GLDyNA-WG. The advantage of our local consistency can be quantified by the reduced performance of GLDyNA-WL. Comparing the results of GLDyNA-WG on two datasets reveals that the impact of global consistency is more significant on the TF dataset compared to the JC dataset. This discrepancy arises due to the reliance of global consistency on changes in the total number of neighbors for nodes in both networks. In the TF dataset, there is a notable increase in the number of edges between different snapshots, resulting in an overall trend of increasing neighbor count for nodes. This trend facilitates the differentiation of nodes using global consistency. Conversely, the JC dataset demonstrates unstable relationships in the changes of edge count across different snapshots, indicating indistinct variations in neighbor count for nodes. This makes it challenging to differentiate nodes using global consistency, resulting in a relatively limited impact of global consistency in this dataset.

Compared to GLDyNA-WG, GLDyNA-WG(L-)'s performance exhibits a slight decline in both $Precision$ and MRR, indicating that when disregarding global consistency, the consideration of local consistency with a limited number of snapshots cannot effectively constrain the node pairs. The significant reduction in the performance of GLDyNA-WS indicates that, in the absence of spatial transformation, the learned node representations of the two networks exhibit certain differences due to their semantic disparities, rendering them unsuitable for direct alignment. The performance of GLDyNA-G is comparable to that of GLDyNA, indicating that for the dataset used in the experiments, the majority of changes in the number of neighbors for anchor node pairs still adhere to linear patterns. This finding aligns with reality, where both in social networks and academic collaboration networks, most individuals experience gradual and non-disruptive changes in the number of their connections or friends under normal circumstances.

Furthermore, to investigate whether the sum of first-order neighbors is the optimal feature for computing global consistency, we conducted experiments to validate the use of different order neighbor counts as features for global consistency. The experimental results are shown in Fig. 6 and Fig. 7, which indicate that considering the sum of higher-order neighbors does not improve the effectiveness of alignment. The sum of higher-order neighbors of a node no longer solely represents its intrinsic characteristics but rather reflects the characteristics of its neighbors. As a result, they provide limited useful information for alignment and may even introduce interference. This observation further supports the rationale behind our approach of utilizing only the sum of first-order neighbors to compute global consistency.

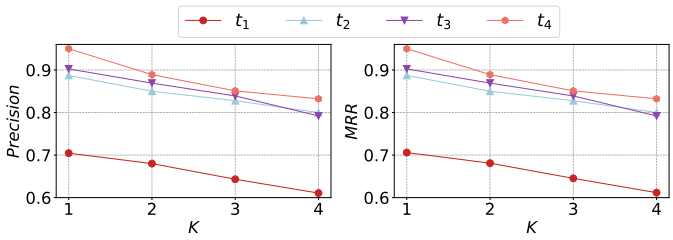

Fig. 6: Experimental results of considering different K-order neighbors in the global consistency on the TF dataset.

5.4 Hyperparameter Sensitivity

To understand the effect of hyperparameters, we analyze accuracy by varying hyperparameters in several experiments. When analyzing each hyperparameter, all other parameters

Authorized licensed use limited to: TIANJIN UNIVERSITY. Downloaded on July 22,2024 at 05:17:01 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Big Data. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TBDATA.2024.3407543

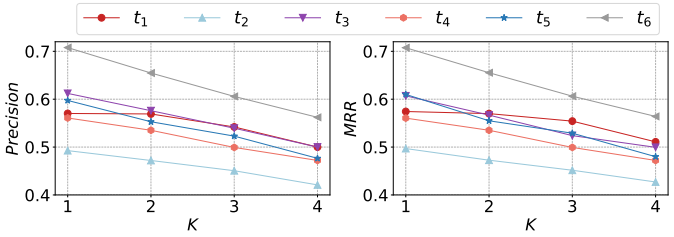

Fig. 7: Experimental results of considering different K-order neighbors in the global consistency on the JC dataset.

are held constant at their default values. The results on TF and JC datasets are shown in Fig. 8 and Fig. 9 respectively.

- Impact of the weight of global consistency λ. We examine the impact of varying the global consistency weight across the range of $[0.05, 0.1, 0.15, 0.2]$, and our results reveal that GLDyNA achieves superior performance with $\lambda = 0.15$ in most snapshots. Although anchor node pairs maintain global consistency between them over time, there may be some deviations from global consistency during the evolution process. Therefore, there are some snapshots where $\lambda = 0.15$ does not achieve the best performance.
- Impact of the weight of local consistency γ. We examine the impact of varying the local consistency weight across the range of $[0.05, 0.1, 0.15, 0.2]$, and our results reveal that GLDyNA achieves superior performance with $\gamma = 0.1$. In comparison to global consistency, the performance of $\gamma = 0.1$ across different snapshots is consistently stable, exhibiting negligible occurrences of anchor nodes deviating from local consistency within any given snapshot.
- Impact of the random walk length l. We examine the impact of varying the random walk length across the range of $[5, 10, 15, 20]$, and our results reveal that GLDyNA achieves superior performance with $l = 15$. The walk length l affects the length of the sampled paths and the coverage of the network, thus influencing the learned node representations. A smaller l leads to denser path sampling, capturing local structure better, but may ignore global structure. A larger l can traverse the network more comprehensively but may overlook local dependencies between nodes. Therefore, we choose $l = 15$ considering a balance between the desired representation accuracy and computational efficiency.
- Impact of the dimension of node representations d. We examine the impact of varying the dimension of node representations across the range of $[32, 64, 128, 256]$, and our results reveal that GLDyNA achieves superior performance with 64-dimension node representations. In general, higher node representation dimensions can better preserve the features of nodes in a network. However, in dynamic network alignment tasks, increasing node dimensions may introduce irrelevant information from the evolutionary process, leading to decreased performance.

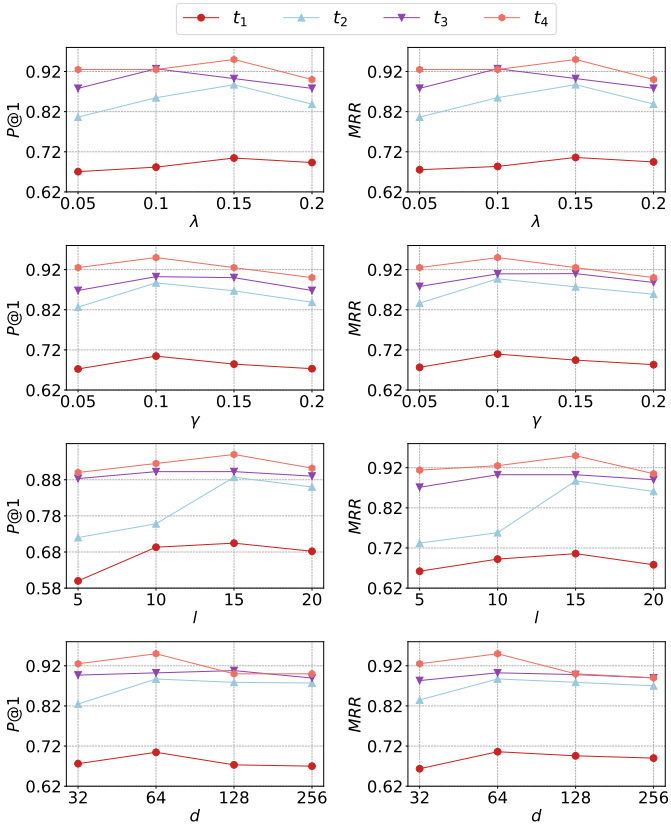

Fig. 8: Experimental results of different model parameters (the weight of global consistency λ, local consistency γ, random walk length l, and node representation dimension d) on the TF dataset.

6 Conclusion

This paper mainly investigates the problem of network alignment in dynamic scenarios. The dynamic nature of networks harbors distinctive patterns that can aid in network alignment. To efficiently utilize the dynamics of networks, we propose a method called GLDyNA to improve the accuracy of network alignment. In the proposed GLDyNA, to capture the intra-network dynamics, we design a Structure-Time-aware module to learn the node representations with network dynamics. To address the inter-network alignment, we ensure the consistency of anchor node pairs from global and local views, respectively. Compared to the STOA alignment methods on real-world datasets, GLDyNA can achieve comparable accuracy performance in dynamic scenarios.

In further research, we endeavor to investigate the intrinsic mechanism of the neighborhood structures of a pair of anchor nodes across disparate networks. Specifically, we aim to generate the neighborhood structure of an anchor node in one network based on its neighborhood structure and historical evolution in another network. By examining the intrinsic mechanism, we can gain a deeper understanding of how these structures are formed and how they evolve. It can not only improve the network alignment but also enhance the interpretability of the results.

Authorized licensed use limited to: TIANJIN UNIVERSITY. Downloaded on July 22,2024 at 05:17:01 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Big Data. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TBDATA.2024.3407543

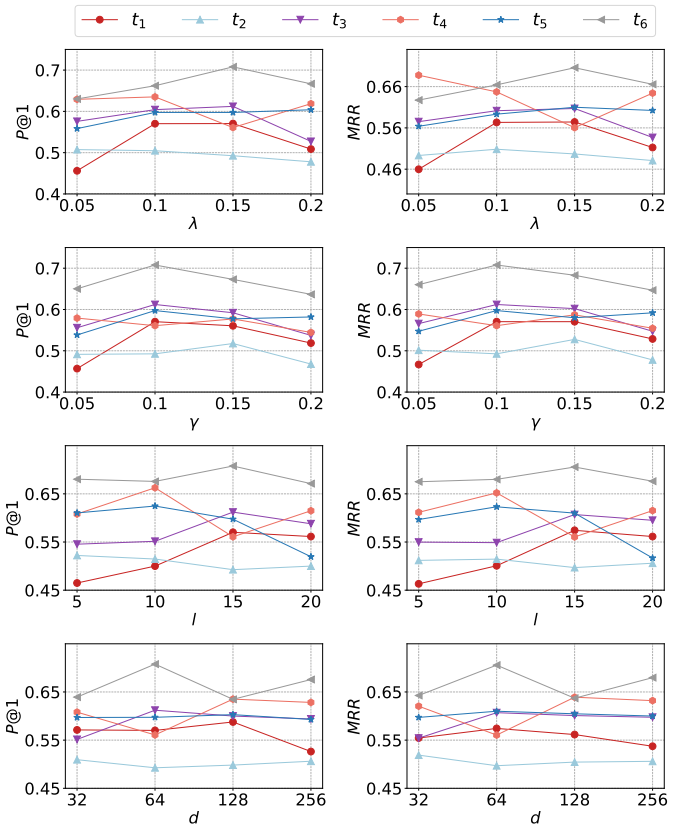

Fig. 9: Experimental results of different model parameters (the weight of global consistency λ, local consistency γ, random walk length l, and node representation dimension d) on the JC dataset.

ACKNOWLEDGMENTS

This work was supported in part by the Sustainable Development Project of Shenzhen (KCXFZ20201221173013036), the Zhejiang Provincial Natural Science Foundation of China under Grant LDT23F01015F01, in part by the Fundamental Research Funds for the Provincial Universities of Zhejiang Grant GK229909299001-008, in part by the Zhejiang Laboratory Open Research Project under Grant K2022QA0AB01.

REFERENCES

[1] C.-Y. Ma and C.-S. Liao, "A review of protein–protein interaction network alignment: From pathway comparison to global alignment," *Computational and Structural Biotechnology Journal*, vol. 18, pp. 2647–2656, 2020.

[2] M. Girisha, V. P. Badiger, and S. Pattar, "A comprehensive review of global alignment of multiple biological networks: background, applications and open issues," *Network Modeling Analysis in Health Informatics and Bioinformatics*, vol. 11, no. 1, p. 9, 2022.

[3] 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, p. 1503–1511.

[4] B. Chen and X. Chen, "Mauil: Multilevel attribute embedding for semisupervised user identity linkage," *Information Sciences*, vol. 593, pp. 527–545, 2022.

[5] S. R. Krishnamurthy and S. A. Jafar, "Precoding based network alignment and the capacity of a finite field x channel," in *2013 IEEE International Symposium on Information Theory*. IEEE, 2013, pp. 2701–2705.

[6] A. Ramakrishnan, "Precoding-based techniques for multiple unicasts in wired and wireless networks," Ph.D. dissertation, UC Irvine, 2015.

[7] Y. Zheng, "Methodologies for cross-domain data fusion: An overview," *IEEE Transactions on Big Data*, vol. 1, no. 1, pp. 16–34, 2015.

[8] Z. Wang, J. Yang, and X. Ye, "Knowledge graph alignment with entity-pair embedding," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020, pp. 1672–1680.

[9] W. Cai, Y. Wang, S. Mao, J. Zhan, and Y. Jiang, "Multi-heterogeneous neighborhood-aware for knowledge graphs alignment," *Information Processing and Management*, vol. 59, no. 1, p. 102790, 2022.

[10] R. Zhang, B. D. Trisedya, M. Li, Y. Jiang, and J. Qi, "A benchmark and comprehensive survey on knowledge graph entity alignment via representation learning," *The VLDB Journal*, vol. 31, no. 5, pp. 1143–1168, 2022.

[11] L. Liu, W. K. Cheung, X. Li, and L. Liao, "Aligning users across social networks using network embedding," in *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, ser. IJCAI'16. AAAI Press, 2016, p. 1774–1780.

[12] X. Chu, X. Fan, D. Yao, Z. Zhu, J. Huang, and J. Bi, "Cross-network embedding for multi-network alignment," in *The World Wide Web Conference*, ser. WWW '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 273–284.

[13] T. T. Huynh, T. C. Duong, T. T. Nguyen, V. V. Tong, A. Sattar, H. Yin, and Q. V. H. Nguyen, "Network alignment with holistic embeddings," in *2022 IEEE 38th International Conference on Data Engineering (ICDE)*, 2022, pp. 1509–1510.

[14] Y. Wang, W. Wang, Z. Zhen, Q. Peng, P. Jiao, W. Liang, M. Shao, and Y. Sun, "Geometry interaction network alignment," *Neurocomputing*, vol. 501, pp. 618–628, 2022.

[15] H. Gao, Y. Wang, S. Lyu, H. Shen, and X. Cheng, "Gcn-alp: addressing matching collisions in anchor link prediction," in *2020 IEEE International Conference on Knowledge Graph (ICKG)*. IEEE, 2020, pp. 412–419.

[16] M. Yang, B. Chen, and X. Chen, "Jarua: Joint embedding of attributes and relations for user alignment across social networks," *Applied Sciences*, vol. 12, no. 24, 2022. [Online]. Available: https://www.mdpi.com/2076-3417/12/24/12709

[17] L. Yang, X. Wang, J. Zhang, J. Yang, Y. Xu, J. Hou, K. Xin, and F.-Y. Wang, "Hackgan: Harmonious cross-network mapping using cyclegan with wasserstein-procrustes learning for unsupervised network alignment," *IEEE Transactions on Computational Social Systems*, pp. 1–14, 2022.

[18] X. Chu, X. Fan, Z. Zhu, and J. Bi, "Variational cross-network embedding for anonymized user identity linkage," in *Proceedings of the 30th ACM International Conference on Information and Knowledge Management*, ser. CIKM '21. New York, NY, USA: Association for Computing Machinery, 2021, p. 2955–2959. [Online]. Available: https://doi.org/10.1145/3459637.3482214

[19] F. Zhou, C. Li, Z. Wen, T. Zhong, G. Trajcevski, and A. Khokhar, "Uncertainty-aware network alignment," *International Journal of Intelligent Systems*, vol. 36, no. 12, pp. 7895–7924, 2021.

[20] M. Heimann, H. Shen, T. Safavi, and D. Koutra, "REGAL: representation learning-based graph alignment," in *Proceedings of the 27th ACM International Conference on Information and Knowledge Management, CIKM 2018, Torino, Italy, October 22-26, 2018*. ACM, 2018, pp. 117–126.

[21] H. Wang, W. Yang, W. Wang, D. Man, and J. Lv, "A novel cross-network embedding for anchor link prediction with social adversarial attacks," *ACM Trans. Priv. Secur.*, vol. 26, no. 1, nov 2022. [Online]. Available: https://doi.org/10.1145/3548685

[22] 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 *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 01, 2020, pp. 147–155.

[23] S. Zhang and H. Tong, "FINAL: Fast attributed network alignment," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '16. New York, NY, USA: Association for Computing Machinery, 2016, p. 1345–1354.

[24] S. Zhang, H. Tong, L. Jin, Y. Xia, and Y. Guo, "Balancing consistency and disparity in network alignment," in *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data*

Authorized licensed use limited to: TIANJIN UNIVERSITY. Downloaded on July 22,2024 at 05:17:01 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Big Data. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TBDATA.2024.3407543

Mining, ser. KDD '21. New York, NY, USA: Association for Computing Machinery, 2021, p. 2212–2222.

[25] C. J. Zhang, L. Chen, H. V. Jagadish, M. Zhang, and Y. Tong, "Reducing uncertainty of schema matching via crowdsourcing with accuracy rates," *IEEE Transactions on Knowledge and Data Engineering*, vol. 32, no. 1, pp. 135–151, 2020. [Online]. Available: https://doi.org/10.1109/TKDE.2018.2881185

[26] K. Xu, L. Wang, M. Yu, Y. Feng, Y. Song, Z. Wang, and D. Yu, "Cross-lingual knowledge graph alignment via graph matching neural network," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, Jul. 2019, pp. 3156–3161. [Online]. Available: https://aclanthology.org/P19-1304

[27] L. Ma, Z. Shao, L. Li, J. Huang, S. Wang, Q. Lin, J. Li, M. Gong, and A. K. Nandi, "Heuristics and metaheuristics for biological network alignment: A review," *Neurocomputing*, 2022.

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

[29] P. Jiao, X. Guo, T. Pan, W. Zhang, Y. Pei, and L. Pan, "A survey on role-oriented network embedding," *IEEE Transactions on Big Data*, vol. 8, no. 4, pp. 933–952, 2022.

[30] S. Zhang, H. Tong, Y. Xia, L. Xiong, and J. Xu, "Nettrans: Neural cross-network transformation," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 986–996.

[31] H. Hong, X. Li, Y. Pan, and I. W. Tsang, "Domain-adversarial network alignment," *IEEE Transactions on Knowledge and Data Engineering*, vol. 34, no. 07, pp. 3211–3224, 2022.

[32] Y. Wang, Q. Peng, W. Wang, X. Guo, M. Shao, H. Liu, W. Liang, and L. Pan, "Network alignment enhanced via modeling heterogeneity of anchor nodes," *Knowledge-Based Systems*, vol. 250, p. 109116, 2022.

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

[34] W. Qiang, J. Li, C. Zheng, B. Su, and H. Xiong, "Robust local preserving and global aligning network for adversarial domain adaptation," *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, no. 3, pp. 3014–3029, 2023.

[35] T. Man, H. Shen, S. Liu, X. Jin, and X. Cheng, "Predict anchor links across social networks via an embedding approach," in *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016*, S. Kambhampati, Ed. IJCAI/AAAI Press, 2016, pp. 1823–1829.

[36] Y. Zhou, J. Ren, R. Jin, Z. Zhang, D. Dou, and D. Yan, "Unsupervised multiple network alignment with multinominal gan and variational inference," in *2020 IEEE International Conference on Big Data (Big Data)*. IEEE, 2020, pp. 868–877.

[37] L. Sun, Z. Zhang, P. Ji, J. Wen, S. Su, and P. S. Yu, "DNA: dynamic social network alignment," *CoRR*, vol. abs/1911.00067, 2019.

[38] L. Sun, Z. Zhang, J. Wen, F. Wang, P. Ji, S. Su, and P. Yu, "Aligning Dynamic Social Networks: An Optimization over Dynamic Graph Autoencoder," *IEEE Transactions on Knowledge and Data Engineering*, vol. 14, no. 8, pp. 1–1, 2022.

[39] J. He, L. Liu, Z. Yan, Z. Wang, M. Xiao, and Y. Zhang, "User alignment across dynamic social networks based on heuristic algorithm," in *2021 7th International Conference on Systems and Informatics (ICSAI)*, 2021, pp. 1–7.

[40] L. Chen and S. Liang, "Cross-temporal snapshot alignment for dynamic multi-relational networks," *Journal of Physics: Conference Series*, vol. 2253, no. 1, p. 012038, apr 2022. [Online]. Available: https://dx.doi.org/10.1088/1742-6596/2253/1/012038

[41] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[42] S. Sra, S. Nowozin, and S. J. Wright, *Optimization for machine learning*. Mit Press, 2012.

[43] J. Leskovec, J. Kleinberg, and C. Faloutsos, "Graphs over time: Densification laws, shrinking diameters and possible explanations," in *Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining*, ser. KDD '05. New York, NY, USA: Association for Computing Machinery, 2005, p. 177–187. [Online]. Available: https://doi.org/10.1145/1081870.1081893

[44] C. Zang, P. Cui, and C. Faloutsos, "Beyond sigmoids: The nettide model for social network growth, and its applications," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '16. New York, NY, USA: Association for Computing Machinery, 2016, p. 2015–2024. [Online]. Available: https://doi.org/10.1145/2939672.2939825

[45] P. D. Hoff, A. E. Raftery, and M. S. Handcock, "Latent space approaches to social network analysis," *Journal of the American Statistical Association*, vol. 97, no. 460, pp. 1090–1098, 2002.

[46] Z. Maoz, R. D. Kuperman, L. Terris, and I. Talmud, "Structural equivalence and international conflict: A social networks analysis," *Journal of Conflict Resolution*, vol. 50, no. 5, pp. 664–689, 2006.

Yinghui Wang is a Eng.D. student in School of College of Intelligence and Computing, Tianjin University, China. Before that, she received her Master degree from School of Computer Science and Technology at Tianjin University in 2018. Her current research interests include complex network analysis and computational social science.

Qiyao Peng received the Bachelor degree in electrical and information engineering from Shandong University in 2018 (Shandong, China). And he received Master's degree from Tianjin University. He is currently pursuing the Doctor degree from Tianjin University, Tianjin, China. His research interests include graph neural networks, expert finding, and large-scale data mining.

Pengfei Jiao 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 of 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 received the B.E. and M.S. degrees from Harbin Institute of Technology, China in 2009 and 2011, respectively, both in electrical engineering. He received the Ph.D. degree with the highest honor in computer science at Freie Universität Berlin, Germany in 2015. He is currently an Associate Professor at the Center for Applied Mathematics, Tianjin University, China. His research interests include wireless networks, mobile edge computing, internet of things and complex networks.

Authorized licensed use limited to: TIANJIN UNIVERSITY. Downloaded on July 22,2024 at 05:17:01 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Big Data. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TBDATA.2024.3407543

Wenjun Wang is currently a Professor at the School of College of Intelligence and Computing, Tianjin University, Chief expert of major projects of the National Social Science Foundation, the big data specially-invited expert of Tianjin Public Security Bureau and the director of the Tianjin Engineering Research Center of Big Data on Public Security. His research interests include computational social science, large-scale data mining, intelligence analysis and multi-layer complex network modeling. He was the principal investigator or was responsible for more than 50 research projects, including the Major Project of National Social Science Fund, the Major Research Plan of the National Natural Science Foundation, the National Science–technology Support Plan Project of China, etc. He has published more than 50 papers in main international journals and conferences.

Authorized licensed use limited to: TIANJIN UNIVERSITY. Downloaded on July 22,2024 at 05:17:01 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.