HUNA: A Method of Hierarchical Unsupervised Network Alignment for IoT

Dongjie Zhu, Yundong Sun, Haiwen Du, Ning Cao, Thar Baker, *Member, IEEE*, and Gautam Srivastava, *Senior Member, IEEE*

***Abstract*—With the advent of the era of the Internet of Things (IoT), a large number of interconnected smart devices form a huge network. The network can be abstracted as a graph, and we propose to identify similar IoT devices in different networks by graph alignment. However, most methods rely on prelabeled cross-network node pairs such as anchor links, which are difficult to obtain due to personal privacy and security restrictions, especially in IoT. In addition, existing network entity alignment methods focus on individual pairs of nodes but ignore the tightly connected group structure in the network, which is a significant feature of IoT devices. In this article, we propose a method of hierarchical unsupervised network alignment (HUNA) to identify similar IoT devices in different networks by a deep learning approach. First, we propose an unsupervised network alignment method based on cycle adversarial networks (UNA), which utilizes the adversarial characteristics of cycle adversarial networks to achieve entity alignment under unsupervised conditions. Second, we further expand the model by carefully designing the group structure aggregation optimization module to aggregate the nodes with closely related attributes and structures into a coarse-grained node and align the coarse-grained nodes. Finally, we evaluate HUNA with real and synthetic data sets. Experimental results show that this method can improve the accuracy of node alignment by 10% and perform well in terms of parameter sensitivity.**

***Index Terms*—AI, deep learning, Internet of Things (IoT), network alignment, network security.**

Manuscript received April 9, 2020; revised June 27, 2020 and August 19, 2020; accepted August 28, 2020. Date of publication September 1, 2020; date of current version February 19, 2021. This work was supported in part by the State Grid Science and Technology Project under Grant 520613180002 and Grant 62061318C002; in part by the Fundamental Research Funds for the Central Universities Grant HIT.NSRIF. 201714; in part by the Weihai Science and Technology Development Program under Grant 2016DXGJMS15; and in part by the Key Research and Development Program in Shandong Provincial under Grant 2017GGX90103. *(Corresponding author: Ning Cao.)*

Dongjie Zhu is with the School of Computer Science and Technology, Harbin Institute of Technology, Weihai 264209, China.

Yundong Sun and Haiwen Du are with the School of Astronautics, Harbin Institute of Technology, Harbin 150001, China.

Ning Cao is with the School of Natural and Computational Sciences, Massey University, Palmerston North 4442, New Zealand (e-mail: ning.cao2008@hotmail.com).

Thar Baker is with the Department of Computer Science, Liverpool John Moores University, Liverpool L3 3AF, U.K.

Gautam Srivastava is with the Department of Mathematics and Computer Science, Brandon University, Brandon, MB R7A 6A9, Canada, and also with the Research Centre for Interneural Computing, China Medical University, Taichung 404, Taiwan.

Digital Object Identifier 10.1109/JIOT.2020.3020951

I. Introduction

GRAPH data are a data structure with a strong expressive ability that can describe the attributes (nodes) of entities and capture the relationships (edges) between entities. In the area of Internet of Things (IoT), the interconnection between devices constitutes a network with different scales and different characteristics, which can be abstracted into graph data to better explore the correlation between devices, where the devices can be abstracted as nodes and the link relationship between the devices can be abstracted as edges in the graph data [1], [2]. The research on graph data has attracted widespread attention in the academic and industrial fields. Although the research on a single graph has achieved many achievements, such as graph representation learning [3]–[5], community discovery [6], and link prediction [7], some problems cannot be solved by a limitation to a single graph, for example, cross-network user relationship prediction [8], cross-network product recommendation [9], and cross-graph financial crime tracking [10]. In cross-graph data research, entity alignment is a fundamental problem, and its purpose is to find the same entities of different graphs. It is common for similar entities to appear in different networks. For example, with the rise and development of various online social networks, one person may register for different social accounts; a scholar may exist in different academic networks such as Google Scholar and Baidu Scholar [11], [12]. Similar devices used by the same person can appear in different networks. Therefore, the research on the network entity alignment method has important academic and application value for identification and tracking of similar devices in IoT scenarios, as shown in Fig. 1.

The most intuitive method is to align the nodes in different networks by comparing artificially defined reference attributes, such as username, age, gender, and hobbies [13], [14]. Since the attributes of matching nodes in different networks may vary greatly, considering only the basic attributes of nodes causes high volatility in the results. For example, the attributes of a device in different networks may be different, along with the settings that may be set according to the functions of different networks. In addition, entity alignment through structural similarity is a common method [15], [16]. This type of method is very sensitive to the network structure, which is not static. In addition, such methods rely on complex operations such as matrix decomposition and are not suitable for decomposing large-scale network data. With the rapid development of deep learning, in recent years, methods based

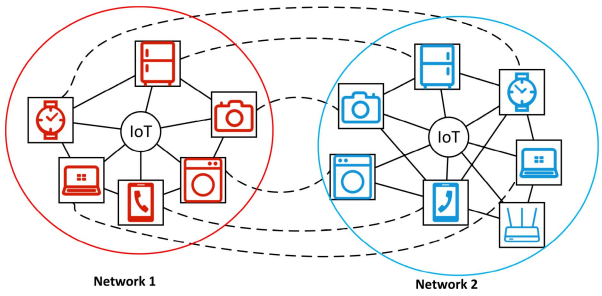

Fig. 1. Diagram of network entity alignment in IoT. Each network can be abstracted into a graph. Nodes represent IoT devices, and edges represent connections between devices. Different colors represent different networks, and devices connected by dotted lines are similar device pairs in different networks identified by network alignment.

on network representation learning [17]–[20] have attracted increasing attention. These methods use network embedding methods to convert node attribute and structural information into a low-dimensional vector space in each graph and then learn mapping functions of vectors between different networks. This kind of method combines the attribute and structural information in the network, which greatly improves the accuracy and stability of entity alignment. However, most such methods rely on manually labeled anchor links. Due to personal privacy and security restrictions, obtaining these anchor links is very difficult, and in large-scale network data scenarios, labeling these data is a very labor-intensive task. In addition, the existing network entity alignment methods simply align the nodes but do not leverage the tightly connected group structure in the network, which has a great effect on improving the performance of entity alignment. Because personal smart home and other IoT devices often present the phenomenon of local network aggregation, the group structure of nodes in graph data is particularly important in the recognition of similar IoT devices across the network.

In this article, we propose a method of hierarchical unsupervised network alignment (HUNA) to identify similar IoT devices in different networks by a deep learning approach. First, we propose an unsupervised network alignment method based on adversarial networks (UNA), which leverages the adversarial properties of adversarial networks to achieve network entity alignment under unsupervised conditions. Second, we design a cycle adversarial structure to solve the problem of abnormal state oscillations in adversarial learning [21]–[23] and achieve bidirectional conversion between networks. Finally, we validate the superiority of our method through experiments on real and synthetic data sets. As far as we know, this is the first AI-based entity alignment method to analyze IoT. We provide a new idea and method for the research of multinetwork data fusion and network security in the 5G era.

The remainder of this article is arranged as follows. Section II discusses some research work related to this article, including network alignment and deep learning. Section III presents our proposed methodology in detail. Section IV evaluates the method and baselines with real experiments and discusses the experimental results in detail. Section V concludes this article and Section VI provides the discussions of the directions for future research.

II. Related Work

In this section, the work related to this article is discussed, mainly including the research work on network alignment and deep learning.

A. Network Alignment

Network entity alignment technology, as an important research point in the field of network data mining, has attracted many researchers in the past decade. The goal of network alignment is to identify the node pairs in different networks, as shown in Fig. 1. The method of network entity alignment based on attribute matching [13], [14] relies on artificially defined features, such as username, age, gender, and hobbies. Because the attributes of nodes have great uncertainty, considering only the basic attributes of nodes causes high volatility in the results. For example, the names of the same user in different networks may be intentionally set to be different, and different parameters may be set according to the functions of different networks.

The second method is based on network topology matching [15], [16]. For example, the core idea of IsoRank [15] is that if the local structures of the nodes in two networks are similar, the more likely they are corresponding nodes. BigAlign [16] considered both the manually defined attributes and the network topology, which greatly improved performance. These kinds of methods are very sensitive to the network structure, which is not static. In addition, such methods rely on complex operations such as matrix decomposition and are not suitable for decomposing large-scale network data.

In the area of IoT, data are accumulating in a variety of formats, and traditional methods are no longer suitable for large-scale and diverse data scenarios. In recent years, with the development of deep learning and network representation learning [24], network entity alignment based on network embedding [12], [25] has received increasing attention, for example, PALE [12]. First, PALE embeds the network into a low-dimensional feature space in which each node is represented as a feature vector that contains rich structural and semantic features. The network entity alignment task is transformed into a mapping function that learns node vector transformation between different network spaces. PALE requires prematched node pairs (anchor links) for supervised training. However, due to personal privacy, security, and commercial barriers, anchor links are difficult to obtain in reality. Therefore, the alignment of network entities in unsupervised learning is the future development trend, especially in massive data scenarios. The approach of Chen *et al.* [26] belongs to the category of unsupervised learning, which does not need the anchor links. Although Chen *et al.* [26] used a similar adversarial network to achieve network entity alignment under unsupervised conditions, it can achieve only a one-way conversion from one network to another, but not a two-way conversion, and it does not use the group hierarchy to optimize the efficiency of entity alignment.

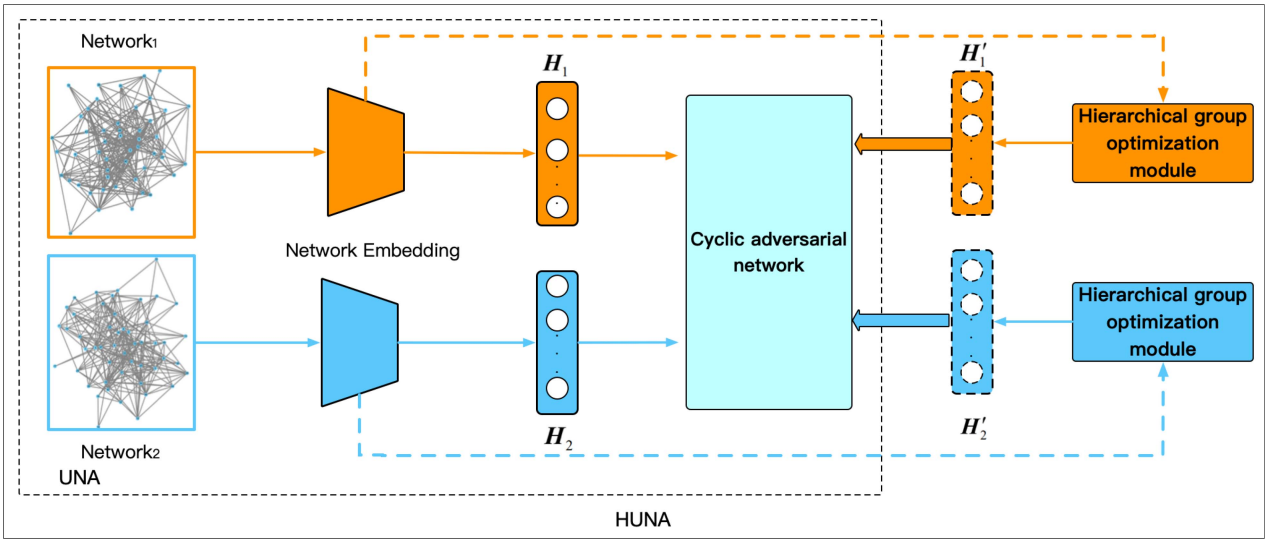

Fig. 2. Diagram of the overall framework. H_1 and H_2 denote low-dimensional vector representations of nodes learned by network embedding methods in Network$_1$ and Network$_2$, respectively. H_1' and H_2' denote the low-dimensional vector representation of the group obtained by the nodes in Network$_1$ and Network$_2$ through our proposed hierarchical group optimization module, respectively.

B. Deep Learning

Deep learning is a technique of nonlinear transformation or representation learning of an input by a neural network with no less than two hidden layers. A deep neural network consists of an input layer, several hidden layers, and an output layer. There are several neurons in each layer, and there are connection weights between the neurons. Each neuron simulates a living organism's neural cells, and connections between nodes simulate connections between neural cells.

Convolutional neural networks (CNNs) are a kind of feed-forward neural network with depth structure and convolution computation. It is one of the representative algorithms of deep learning. CNN [27], [28] has the capability of representation learning and can carry out shift-invariant classification of input information according to its hierarchical structure, so it is also known as "translation-invariant artificial neural networks." Hubel and Wiesel [27] recorded the electrical activity of various neurons in a cat's brain. They used a slide projector to show the cat certain patterns and noted that certain patterns stimulated activity in certain parts of the brain. This single-neuron recording was an innovative work, made possible by special recording electrodes invented by Hubel earlier, which they used to systematically create maps of the visual cortex. In the ImageNet recognition contest of 2012, Krizhevsky *et al.* [28] introduced Alexnet into the new deep structure and dropout method and reduced the error rate from above 25% to 15%, which upended the image recognition field. The former CNN also has many applications in natural language processing, speech signal processing, and other fields.

Recursive neural networks (RNNs), which are used to process sequence data, are another representative model in the field of deep learning. In the traditional neural network model, the layers are fully connected, and the nodes between each layer are disconnected. However, this kind of ordinary neural network cannot process the sequence data, in which there is a dependency relationship between each of the sequence data. For example, predicting the next word in a sentence usually involves using the previous word because the words before and after a given word are not independent. RNN refers to a sequence in which the current output is related to the previous output. The specific manifestation is that the network will remember the previous information, keep it in the internal state of the network, and apply it to the calculation of the current output. That is, the nodes between the hidden layers are no longer connectionless but linked, and the input of the hidden layer contains not only the output of the input layer but also the output of the hidden layer at the previous time. In theory, RNN can process sequence data of any length, but in practice, to reduce complexity, it is often assumed that the current state is associated with only a small number of previous states. LSTM [29] and GRU [30] are common RNN variants and have been applied in machine translation, speech signal processing, and other fields.

CNN is used to process spatial information, and RNN is used to process sequence data. With the rapid development of the Internet and the popularity of mobile devices, we can carry out online chatting, online sharing, and online transactions anytime and anywhere. A large amount of data are generated on the network. There are all kinds of complex and diverse relationships between different individuals, which can be abstracted as graph data. Prior to the advent of graph neural networks (GNNs), the existing models in deep learning could not directly process such graph data. The concept of GNN was first proposed by Scarselli *et al.* [31] in 2009. It extends existing neural networks to process data represented in graphs. In the graph, each node is defined by its characteristics and associated nodes. Later, with the further analysis of graph data, some new GNN variants, such as G2S [32] and R-GCN [33], were proposed. A class of variants is the information processing of edges, such as weight or edge type. With the

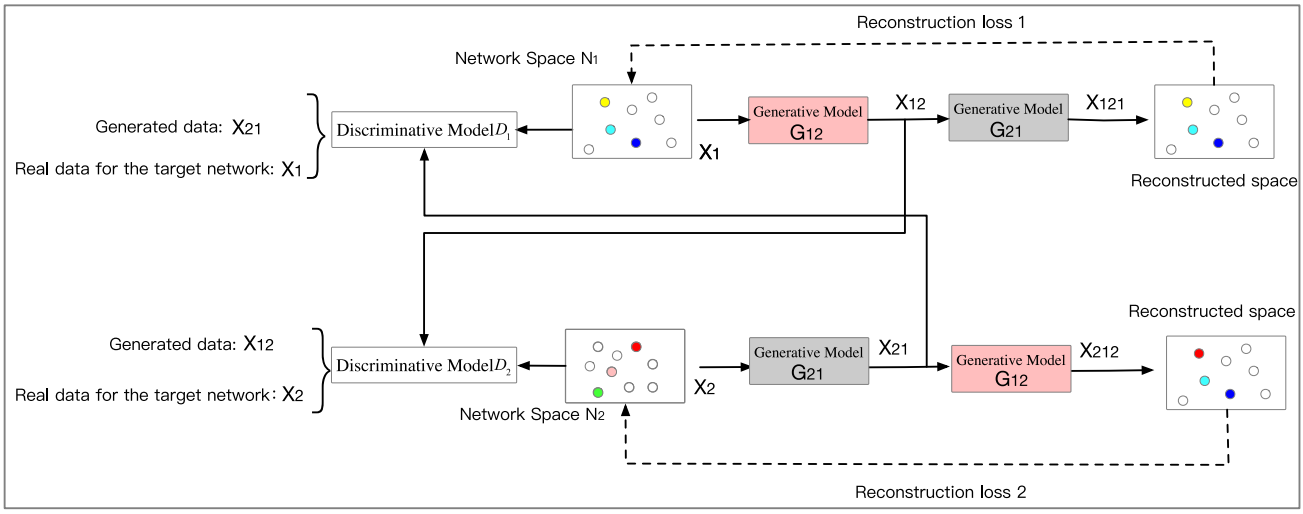

Fig. 3. Diagram of the UNA structure.

continuous development of online networks, a dynamic graph processing algorithm [34] was proposed by Shi *et al.* This model regarded the graph evolution as a time sequence process and continuously predicted the graph information of the next time point by using the information in current and historical graphs. With the emergence and application of various complex systems, networks have evolved from homogeneous networks with the same node type and the same relationship type into heterogeneous networks with multiple node types and multiple relationship types. The latest GNN development direction has also been implemented in heterogeneous network research, such as metapath-based methods [35], [36] and multiple information (text and image) fusion methods [37].

With the continuous development and improvement of deep learning, some methods use it to solve problems in IoT scenarios. Hu *et al.* [38] used network representation learning technology to solve the representation problem of interaction and communication data in IoT and proved that the proposed method can more effectively capture the semantic relationship of IoT data through experiments. Mohammadi *et al.* [39] adopted the method of reinforcement learning to improve intelligent service capability in IoT and smart city scenarios. Its core is the use of reinforcement learning to make proper use of unmarked data to solve the problem of difficult acquisition of training data in IoT scenarios. Abbasi *et al.* [40] proposed a method to optimize the partial distribution of workload and created a balance between latency and edge power consumption. Liao *et al.* [41] proposed a channel selection method based on learning, which combines service reliability awareness, energy awareness, backlog awareness, and improved throughput by 30% and 36%. In addition, there are some studies using related methods to solve data storage [42], [43] and security [44] issues in IoT scenarios. Huang *et al.* [42] proposed ThinO-RAM to remove complicated computations in the client side and requires only O(1) communication cost with the reasonable response time. Liu *et al.* [43] proposed a novel multicloud ORAM scheme, called NewMCOS to achieve better performance, much smaller client cache size, lower read and write overheads. Li *et al.* [44] proposed the concept of "forward search privacy," which ensured that the search operation on the newly added documents would not reveal the past query information, and developed new forward privacy technology to achieve greater security goal.

III. Hierarchical Unsupervised Network Alignment

In this section, we first present the overall idea and framework of the proposed model and then detail the technical implementation and discuss key technologies in each part.

A. Overview

The overall framework of our proposed model is shown in Fig. 2. First, we obtain low-dimensional vector representations of nodes through unsupervised network embedding methods (such as DeepWalk [45]). Second, the low-dimensional vector representation of the nodes is input to the proposed cycle adversarial network for the learning of entity alignment mapping. Finally, with the help of the proposed hierarchical group optimization module, we coarse-grain nodes with similar attributes and a tight structure into a virtual node (representing a group) and perform entity alignment optimization on the virtual node. We detail each core module in the next section.

B. Unsupervised Network Alignment

First, an unsupervised entity alignment method based on adversarial learning is proposed. The overall structure of the model is shown in Fig. 3. The network consists of two parts. One part is the generator model, which maps the node vectors in the source network to the target network space. The other part is the discriminative model, which determines whether the input node is a real node or a node mapped by another network. For example, when we transform network N_1 to network N_2, the generator G_{12} is responsible for mapping the node vector of space N_1 to space N_2, and its goal is to

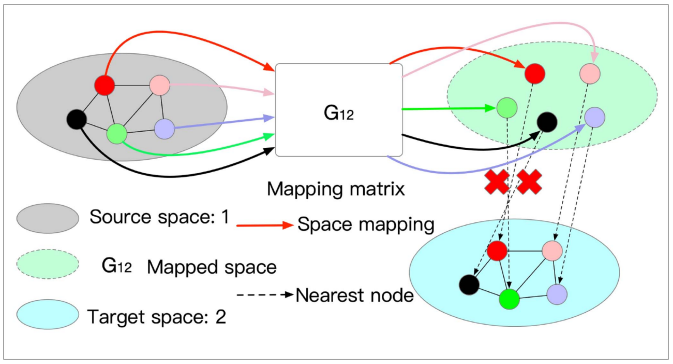

Fig. 4. Diagram of the overall distribution fitting correctly but partial nodes fitting incorrectly. The positions of nodes represent the distribution of nodes in the network space, and the same color represents the node corresponding to the same entity in different spaces. The black and green nodes in Network$_1$ have mapping errors in the spatial distribution after being mapped by G_{12}, and their positions are staggered, which leads to errors in the search for matching the nearest nodes in Network$_2$.

"cheat" the discriminative model as much as possible, so that it is impossible to distinguish whether the current node is G_{12} mapped to space N_2 or the real node of space N_2. The purpose of the discriminative model is to distinguish the mapped node and the real node as much as possible. When training, some of the vector of space N_2 is used as the positive sample, and the generated node of G_{12} is used as the negative sample to train the discriminative model while maintaining G_{12}. With the same D_1, the training of G_{12} is then optimized by using the decision results of D_1. Repeatedly, the generator model and the discriminative model interact until the generator reaches a certain level of precision.

Therefore, the loss function of the adversarial network is calculated as

$$\min_{G_{12}} \max L(G_{12}, D_1, N_1, N_2) = \log D_1(N_2) + \log(1 - D_1(G_{12}(N_1))). \tag{1}$$

The objective function of network N_2 to network N_1 conversion is calculated as

$$\min_{G_{21}} \max L(G_{21}, D_2, N_1, N_2) = \log D_2(N_1) + \log(1 - D_2(G_{21}(N_2))). \tag{2}$$

Through the above objective function, the mutual mapping of the two network vector spaces is realized under the condition of no labeled training data. However, the problem shown in Fig. 4 will occur; i.e., the current antagonistic network can achieve the correct mapping of the overall distribution of the two network nodes, but corresponding errors of specific nodes will occur. The details are shown in Fig. 5(a) and (b).

To solve the above problems, the reconstruction loss module is added based on the original adversary network to ensure the alignment of nodes as shown by the dotted line in Fig. 3. After G_{12} mapping the nodes in the source network space N_1 to the destination network space N_2, we expect G_{21} to have the ability to restore the mapped nodes to the source network

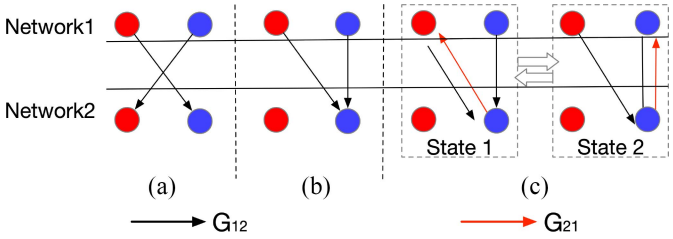

Fig. 5. Detail diagram of partial nodes fitting incorrectly in (a) and (b) and the state oscillation problem in (c). In (a), there is a cross-mapping error between the node pairs in Network$_1$ and Network$_2$. In (b), two nodes in Network$_1$ map to the same node in Network$_2$, while other nodes in Network$_2$ have no corresponding nodes in Network$_1$. In (c), after adding the reconstruction loss, the phenomenon of oscillation between the two states is prone to occur during the training.

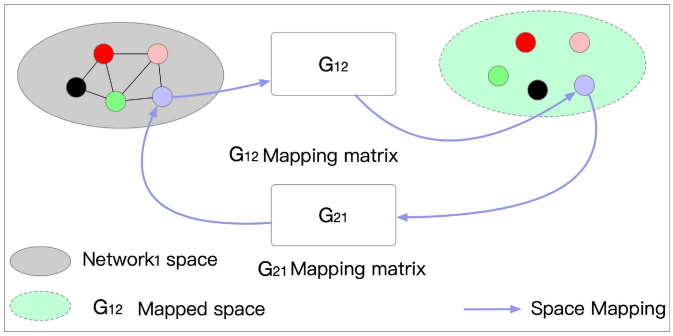

Fig. 6. Diagram of node reconstruction. After nodes in network 1 are mapped by G_{12}, they can be reverted to network 1 by G_{21} ideally. After continuous training to reduce the reconstruction loss.

space N_1, as shown in (3). The diagram of node reconstruction is shown in Fig. 6

$$G_{21}(G_{12}(N_1)) \approx N_1. \tag{3}$$

Therefore, the loss function of the Cycle-GAN is calculated as

$$\min_{G_{12},G_{21}} L_{cyc}(G_{12}, G_{21}, N_1, N_2) = \|G_{21}(G_{12}(N_1)) - N_1\|_1 + \|G_{12}(G_{21}(N_2)) - N_2\|_1. \tag{4}$$

However, the instability problem of state oscillation will still occur in the model after the reconstruction loss is added, as shown in Fig. 5(c). To solve this problem and further optimize the network structure, we utilize a cycle adversarial network with two pairs of generative and discriminative models, which contains two reconstruction loss parts. Generators of the same color represent parameter sharing. Through two-way training of the two networks, mutual conversion between the two network nodes under unsupervised conditions is achieved.

C. Hierarchical Group Optimization Module

The module of the overall process is diagrammed in Fig. 7. By the network embedding method, we can obtain the vector representation of the node that contains the semantic relationship between nodes, and then similar nodes can be gathered by a clustering algorithm (such as k-means). To provide effective input in the next group alignment, the information aggregation

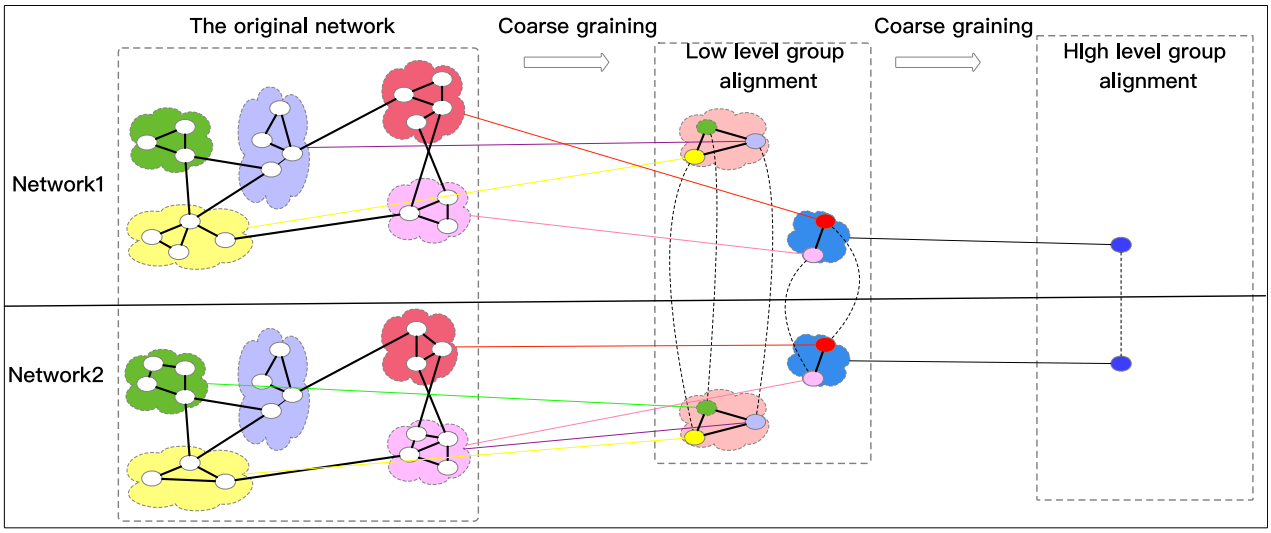

Fig. 7. Flow diagram of the hierarchical group optimization module. The black bold line represents the link edge between nodes or groups, the colored thin lines represent the coarse-grained process or group gathering process, the different node colors represent the different groups divided, and the black dotted line represents the entity alignment process.

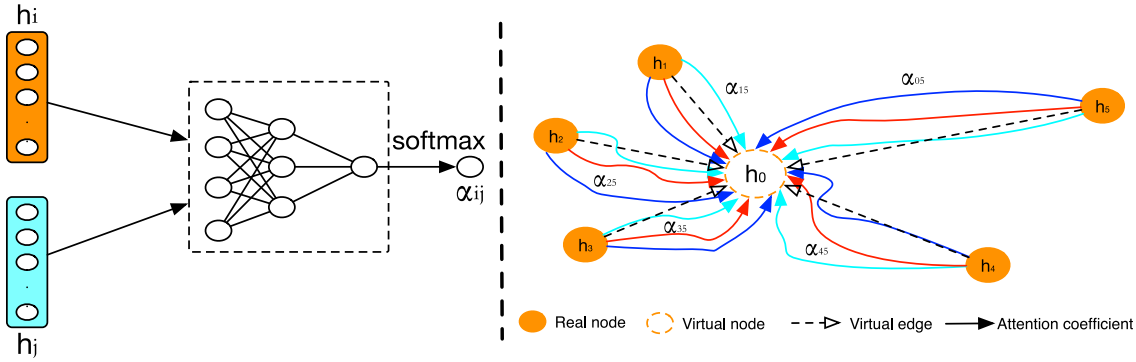

Fig. 8. Left image is the schematic of the graph attention network structure. The right image is a diagram of information aggregation. The entity node is the real node divided into the unified group, the virtual node represents the group after the aggregation, and the dotted line represents the virtual connection edge from the real node to the virtual node.

process is also needed to represent all nodes in a group as a virtual node vector.

One simple method is to use the sum of all node vectors or an averaging operation. However, the aggregation results of these methods can be easily affected by special nodes, and changes to a few attributes have a greater influence on the final result, which is not conducive to the stability of the model training. Therefore, we need a smooth mechanism within the group on the properties of the nodes for smooth operation.

A hierarchical group optimization method based on graph attention is proposed in this article, which uses the graph attention network to perform information aggregation operations on nodes in the group.

First, a virtual node is introduced into each group, and a virtual edge with a real node pointing to the virtual node is added between each real node and the virtual node. Second, the real node h_i and the virtual node vector h_j are input to the graph attention network to calculate the attention weight, which is calculated as

$$e_{ij} = \text{LeakyRelu}\left(W\left[h_i \| h_j\right]\right) \tag{5}$$

where W denotes the trainable parameters. The normalized weight is obtained through the SoftMax function, which is calculated as

$$\alpha_{ij} = \frac{\exp\left(e_{ij}\right)}{\sum_{s \in N_i} \exp\left(e_{is}\right)} \tag{6}$$

where N_i represents the neighbors of node i. After the attention weight of each node to the virtual node is obtained, the final vector representation of the virtual node is further calculated, it will also be the final coarse-grained representation of the group, which is calculated as

$$h_i = \sigma\left(\sum_{k \in N_i} \alpha_{ik} \cdot h_k\right). \tag{7}$$

To make the training more stable and further reduce the impact brought by the variability of the network structure, we added multiple attention, which is calculated as

$$h_i = \Big\|_{k=1}^{K} \sigma\left(\sum_{j \in N_i} \left[\alpha_{ij}\right]_k \cdot h_k\right) \tag{8}$$

where $[\alpha_{ij}]_k$ represents the weight calculated by the kth attention network, and K denotes the header numbers. The schematic is shown in Fig. 8.

As for the link relationship between the two groups after coarse graining, we stipulate here that if there are real nodes connected between the two groups, an edge is added between the virtual nodes of the two groups.

IV. Experiment

In this part, we comprehensively measured our method based on the data set generated from the real data set and compared and analyzed it with other supervised learning methods and unsupervised learning methods. We also verified the stability of our model through case study experiments

A. Data Set

In the experiment, we selected three real data sets and one generated data set from [3] to simulate IoT data.

Douban Online Versus Douban Offline Real Data Sets: These data were taken from the Douban network. Network 1 is an online interactive relationship network; network 2 is a Douban offline network. The edges between nodes and points indicate that two users have met in the same social event offline. Offline network 2 contains 1118 users, and online network 1 contains 3906 users, including the 1118 offline network users. The user's geographical location is used to build the node properties.

Flickr Versus Lastfm Real Data Sets: The Flickr network has 12 974 nodes, and the Lastfm network has 15 436 nodes. The gender of the user is used to construct the node properties, and the alignments between some nodes can be obtained.

Twitter Versus Facebook (FB-TW) Real Data Sets: These data sets were gathered from Singapore-based Facebook and Twitter accounts. The Facebook subnetwork has 17 359 nodes, and the Twitter subnetwork has 20 024 nodes. The partial ground truth of 1998 user identity pairs is determined by users' short biographical descriptions on their Twitter accounts in which they state their Facebook accounts. Node features are not provided for this data set.

Protein–Protein Synthetic Data Sets: In order to better measure the performance impact of network characteristics on the model, we adopted a method like [3] to generate data sets. Synthetic data sets from the protein–protein data set. To learn the effect of structural noise, we remove edges with different probability without leaving any nodes isolated.

The statistics for all data sets are shown in Tables I and II, where RN denotes the ratio of two nodes from ground truth, in which neighbors in the source network also have a connection in the target network. RSFG is the ratio to which nodes in ground truth share similar attributes in the source and target networks.

TABLE I
Data Sets Information

	Douban	Flickr-Lastfm	FB-TW
Number of nodes	3906-1118	12974-15436	6714-10733
Number of edges	8164-1511	16149-16319	7333-11081
Average degree	4.18-2.70	2.49-2.11	2.18-2.06

TABLE II
Data Sets Insight

	Douban	Flickr-Lastfm	FB-TW
RN	0.3851	0.0017	0.0149
RSFG	1.000	0.8248	No features

TABLE III
Real Data Accuracy

Methods	Douban	Flickr-Lastfm	FB-TW
IsoRank	0.0418	0.3967	0.0017
BigAlign	0.0123	0.0051	0.0009
FINAL	0.4134	0.6642	0.0006
UNA	0.4317	0.6297	0.0123
HUNA	**0.5287**	**0.6834**	**0.02361**

B. Comparison Methods

In this article, we compare the following methods.

ISORANK [15]: IsoRank is a popular and typical technique for this category. The general idea of IsoRank is that two nodes from the two networks are similar if their neighborhoods are similar.

BIGALIGN [16]: BigAlign aims to solve the network alignment problem by converting to bipartite graphs using the original feature and handcrafted features of network nodes, such as node degree, weight, and cluster coefficient. Given the specific structure of bipartite graphs, BigAlign hopes to achieve better alignment quality.

FINAL [23]: FINAL is a typical example that defines three criteria to align two networks: 1) structural similarity; 2) node feature similarity; and 3) edge feature similarity.

DEEPLINK [25]: Deep Link involves a preprocessing step in which prior mappings between two graphs are used to fill in missing edges that are available in one graph but not the other. Deep Link achieves the approach by constructing the graph embedding, and its mapping function differs as it considers the mapping direction as well.

UNA (Our): Our model without hierarchical model optimization.

HUNA (Our-Full): Our model with hierarchical model optimization.

C. Experiment Settings

For the method we proposed, first, the network representation learning process was required. In the experiment, Deepwalk was used for unsupervised network embedding to obtain the low-dimensional vector representations of nodes. The window size was set to 5, and the embedded dimension was set to 32. Therefore, the generator of the anti-neural network was 32×32, and the judges for the anti-neural network were set as two-layer neural networks with a size of 2048. Leaky-Relu was used as the activation function. For our hierarchical structure optimization module, we adopted two-layer iterative optimization. The number of the coarse k-means group in the first layer was set to $= 200$, the number of the coarse k-means group in the second layer was set to $= 50$,

TABLE IV
RESULT COMPARISON ON PROTEIN–PROTEIN SYNTHETIC DATA SETS

Metrics	Edge removal ratio	IsoRank	BigAlign	FINAL	Our(full)
Accuracy	0	0.990	0.607	0.393	**0.992**
	0.01	0.262	0.582	0.368	**0.974**
	0.05	0.233	0.514	0.342	**0.932**
	0.1	0.151	0.467	0.324	**0.894**
	0.2	0.122	0.321	0.315	**0.833**
MAP	0	0.160	0.211	0.603	**0.928**
	0.01	0.140	0.183	0.360	**0.892**
	0.05	0.122	0.154	0.320	**0.881**
	0.1	0.103	0.130	0.293	**0.872**
	0.2	0.081	0.112	0.214	**0.845**
Precision@10	0	0.113	0.221	0.623	**0.998**
	0.01	0.091	0.212	0.490	**0.962**
	0.05	0.082	0.198	0.381	**0.951**
	0.1	0.071	0.161	0.342	**0.922**
	0.2	0.060	0.144	0.315	**0.898**

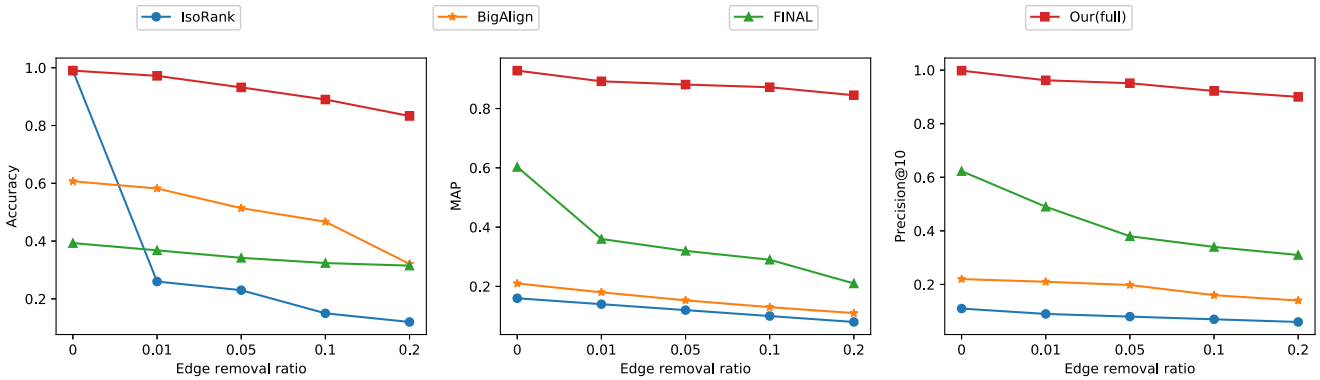

Fig. 9. Result of the experiment to explore the effects of the edge removal ratio. The horizontal axis is the edge removal ratio and represents the difference between the two networks. The vertical axis shows the performance of various models in terms of three indicators. Different colors represent different models.

and the number of multiple attention was set to $k = 3$. For the benchmark method, we adopted the same optimal parameters as in [1] by default, and each result was averaged after ten runs. The experimental environment was a Pytorch with an Intel Core i7-9700k CPU, 32 GB of RAM, and an NVIDIA GeForce RTX 2080 TI 11G GPU.

D. Evaluation

In this article, we use the following indicators to evaluate the model.

1) *Accuracy:* Accuracy is a critical evaluation index of network alignment, which is calculated as

$$\text{Accuracy} = \frac{n_1}{n} \tag{9}$$

where n_1 is the number of correctly identified node pairs, and n is the number of all ground-truth node pairs.

2) *PRECISION@K:* In addition to accuracy, we also use Precision@k evaluation, which is calculated as

$$\text{Precision@k} = \frac{n_{rk}}{n} \tag{10}$$

where n_{rk} is the number of times that the target node appears in the top n similarity candidates, and is the number of all ground-truth node pairs.

3) *Mean Average Precision (MAP):* In practical applications, we want the correct results to be ranked as high as possible among all potential results each time. Therefore, to measure the performance of the model more comprehensively, we also employ MAP as a metric to measure its performance in sorting multiple potential results. MAP can be calculated as

$$\text{MAP} = \frac{1}{ra} \tag{11}$$

where ra is the rank of a positive matching node in the sequence of sorted candidates.

E. Experimental Results and Analysis

The experimental results of each method on real data are shown in Table III. It can be seen in the table that BigAlign had the worst performance, it ignores the rich attribute information on nodes and edges in many real graphs and is prone to lead to suboptimal results. IsoRank and FINAL achieved better results, mainly due to the use of username similarity. The result of the UNA without the hierarchical optimization method is worse than FINAL on the Flickr-Lastfm data set. This may be caused by the instability of the training of GAN, which is also the pain point in the field of deep learning. The HUNA achieved the best results. This shows that our hierarchical optimization module has a very good optimization effect. It also eliminates the performance problems caused by the training instability of GAN to some extent.

F. Case Study

We use the graph size imbalance to measure the sensitivity of the model. Graph size imbalance is another possible factor influencing the alignment method. In this experiment, we use source graphs with 1000 nodes and 5000 nodes and then randomly remove nodes from the source graph to generate the target graph. The ratio is from 0 to 0.5, so the imbalance between the source and target networks is 1:1 to 2:1. It can be seen in Fig. 9 that compared with the contrasting method, the stability of the proposed method is superior to the other methods. In terms of precision, the performance of IsoRank and our method is best without removing edges, but the performance drops sharply after removing a few edges. The other methods are relatively stable, but the accuracy is always less than 0.6. The results of MAP and Precision@10 are similar. The FINAL performance is good at the beginning but drops with an increasing edge removal ratio. Our proposed model is consistently the best performing and most stable and has significant advantages over the other approaches. The detailed result is shown in Table IV.

V. Conclusion

In this article, we propose a group structure-enhanced unsupervised cross-network entity alignment method to solve the problem of identifying similar devices under different networks. First, we design a node alignment model based on the cycle adversarial network, which makes full use of the adversarial properties of the adversarial network to achieve unsupervised network entity alignment and adds a bidirectional cycle structure to solve the problems of state oscillation and instability in the adversarial learning. Second, we proposed for the first time adopting a group structure to optimize the alignment of network entities. Through a well-designed group structure aggregation optimization module, the nodes closely related to attributes and structures were aggregated into a coarse-grained node, and the coarse-grained nodes were aligned at the group level. Experiments on several real data sets and generated data sets verify that our method improves the accuracy by approximately 10%–20% and the stability by a large amount. As far as we know, this is the first AI-based entity alignment to enable the analysis and identification of similar IoT devices across different networks. This will provide a new idea and method for the research of multinetwork data fusion and network security in the 5G era.

VI. Discussion

However, it is important to note that in a real data set, all methods failed to achieve a result greater than 0.7, so the result was close to zero, suggesting that the present method applied to real data sets achieved a significant difference. We suppose that the main reason is that the difference between network structures is more uncertain (e.g., node degree) than the generated data. This is the direction that network entity alignment researchers should explore. Second, the group information was regarded as the optimization of entity alignment for the first time. How to integrate group discovery with entity alignment and achieve end-to-end hierarchical entity alignment is our future research direction. Finally, in the 5G era, IoT devices are frequently added and updated. Therefore, how to realize data fusion and analysis of dynamic network is a research hotspot in the future.

References

[1] O. Novo, "Scalable access management in IoT using blockchain: A performance evaluation," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4694–4701, Jun. 2018.

[2] H. Alasmary *et al.*, "Analyzing and detecting emerging Internet of Things malware: A graph-based approach," *IEEE Internet Things J.*, vol. 6, no. 5, pp. 8977–8988, Oct. 2019.

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

[4] B. D. Trisedya, J. Qi, and R. Zhang, "Entity alignment between knowledge graphs using attribute embeddings," in *Proc. AAAI Conf. Artif. Intell.*, vol. 33, 2019, pp. 297–304.

[5] L. F. R. Ribeiro, P. H. P. Saverese, and D. R. Figueiredo, "Struc2Vec: Learning node representations from structural identity," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Disc. Data Min.*, 2017, pp. 385–394.

[6] C. Tu *et al.*, "A unified framework for community detection and network representation learning," *IEEE Trans. Knowl. Data Eng.*, vol. 31, no. 6, pp. 1051–1065, Jun. 2019.

[7] T. Li, J. Zhang, P. S. Yu, Y. Zhang, and Y. Yan, "Deep dynamic network embedding for link prediction," *IEEE Access*, vol. 6, pp. 29219–29230, 2018.

[8] Y. Shang *et al.*, "PAAE: A unified framework for predicting anchor links with adversarial embedding," in *Proc. IEEE Int. Conf. Multimedia Expo (ICME)*, 2019, pp. 682–687.

[9] J. Zhang *et al.*, "MEgo2Vec: Embedding matched ego networks for user alignment across social networks," in *Proc. 27th ACM Int. Conf. Inf. Knowl. Manag.*, 2018, pp. 327–336.

[10] Y. Fan *et al.*, "iDev: Enhancing social coding security by cross-platform user identification between github and stack overflow," in *Proc. 28th Int. Joint Conf. Artif. Intell. (IJCAI)*, 2019, pp. 2272–2278.

[11] Y. Li, Y. Peng, Z. Zhang, H. Yin, and Q. Xu, "Matching user accounts across social networks based on username and display name," *World Wide Web*, vol. 22, no. 3, pp. 1075–1097, 2019.

[12] T. Man, H. Shen, S. Liu, X. Jin, and X. Cheng, "Predict anchor links across social networks via an embedding approach," in *Proc. IJCAI*, vol. 16, 2016, pp. 1823–1829.

[13] A.-C. N. Ngomo and S. Auer, "LIMES—A time-efficient approach for large-scale link discovery on the Web of data," in *Proc. 22nd Int. Joint Conf. Artif. Intell.*, 2011, pp. 2312–2317.

[14] M. Pershina, M. Yakout, and K. Chakrabarti, "Holistic entity matching across knowledge graphs," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, 2015, pp. 1585–1590.

[15] 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, 2008.

[16] D. Koutra, H. Tong, and D. Lubensky, "Big-align: Fast bipartite graph alignment," in *Proc. IEEE 13th Int. Conf. Data Min.*, 2013, pp. 389–398.

[17] Y. Hao, Y. Zhang, S. He, K. Liu, and J. Zhao, "A joint embedding method for entity alignment of knowledge bases," in *Proc. China Conf. Knowl. Graph Semantic Comput.*, 2016, pp. 3–14.

[18] L. Liu, W. K. Cheung, X. Li, and L. Liao, "Aligning users across social networks using network embedding," in *Proc. IJCAI*, 2016, pp. 1774–1780.

[19] M. Chen, Y. Tian, M. Yang, and C. Zaniolo, "Multilingual knowledge graph embeddings for cross-lingual knowledge alignment," 2016. [Online]. Available: arXiv:1611.03954.

[20] Z. Yi, H. R. Zhang, P. Tan, and M. Gong, "DualGAN: Unsupervised dual learning for image-to-image translation," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 2868–2876.

[21] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 2242–2251.

[22] T. Kim, M. Cha, H. Kim, J. K. Lee, and J. Kim, "Learning to discover cross-domain relations with generative adversarial networks," in *Proc. 34th Int. Conf. Mach. Learn.*, vol. 70, 2017, pp. 1857–1865.

[23] S. Zhang and H. Tong, "FINAL: Fast attributed network alignment," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Disc. Data Min.*, 2016, pp. 1345–1354.

[24] D. Zhu *et al.*, "BDNE: A method of bi-directional distance network embedding," in *Proc. IEEE Int. Conf. Cyber Enabled Distrib. Comput. Knowl. Disc. (CyberC)*, 2019, pp. 158–161.
[25] F. Zhou, L. Liu, K. Zhang, G. Trajcevski, J. Wu, and T. Zhong, "DeepLink: A deep learning approach for user identity linkage," in *Proc. IEEE INFOCOM IEEE Conf. Comput. Commun.*, 2018, pp. 1313–1321.
[26] C. Chen *et al.*, "Unsupervised adversarial graph alignment with graph embedding," 2019. [Online]. Available: arXiv:1907.00544.
[27] D. H. Hubel and T. N. Wiesel, "Receptive fields, binocular interaction and functional architecture in the cat's visual cortex," *J. Physiol.*, vol. 160, no. 1, pp. 106–154, 1962.
[28] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1106–1114.
[29] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.
[30] K. Cho *et al.* "Learning phrase representations using RNN encoder–decoder for statistical machine translation," 2014. [Online]. Available: arXiv:1406.1078.
[31] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE Trans. Neural Netw.*, vol. 20, no. 1, pp. 61–80, Jan. 2009.
[32] D. Beck, G. Haffari, and T. Cohn, "Graph-to-sequence learning using gated graph neural networks," 2018. [Online]. Available: arXiv:1806.09835.
[33] M. S. Schlichtkrull, T. N. Kipf, P. Bloem, R. van den Berg, I. Titov, and M. Welling, "Modeling relational data with graph convolutional networks," in *Proc. Eur. Semantic Web Conf.*, 2018, pp. 593–607.
[34] Y. Shi, M. Lei, H. Yang, and L. Niu, "Diffusion network embedding," *Pattern Recognit.*, vol. 88, pp. 518–531, Apr. 2019.
[35] Y. Dong, N. V. Chawla, and A. Swami, "Metapath2Vec: Scalable representation learning for heterogeneous networks," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Disc. Data Min.*, 2017, pp. 135–144.
[36] C. Shi, B. Hu, W. X. Zhao, and P. S. Yu, "Heterogeneous information network embedding for recommendation," *IEEE Trans. Knowl. Data Eng.*, vol. 31, no. 2, pp. 357–370, Feb. 2018.
[37] C. Zhang, D. Song, C. Huang, A. Swami, and N. V. Chawla, "Heterogeneous graph neural network," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Disc. Data Min.*, 2019, pp. 793–803.
[38] L. Hu, G. Wu, Y. Xing, and F. Wang, "Things2Vec: Semantic modeling in the Internet of Things with graph representation learning," *IEEE Internet Things J.*, vol. 7, no. 3, pp. 1939–1948, Mar. 2020.
[39] M. Mohammadi *et al.*, "Semisupervised deep reinforcement learning in support of IoT and smart city services," *IEEE Internet Things J.*, vol. 5, no. 2, pp. 624–635, Jun. 2017.
[40] M. Abbasi, M. Yaghoobikia, M. Rafiee, A. Jolfaei, and M. R. Khosravi, "Efficient resource management and workload allocation in fog-cloud computing paradigm in IoT using learning classifier systems," *Comput. Commun.*, vol. 153, pp. 217–228, Mar. 2020.
[41] H. Liao *et al.*, "Learning-based context-aware resource allocation for edge computing-empowered industrial IoT," *IEEE Internet Things J.*, vol. 7, no. 5, pp. 4260–4277, May 2020.
[42] Y. Huang *et al.*, "ThinORAM: Towards practical oblivious data access in fog computing environment," *IEEE Trans. Services Comput.*, vol. 13, no. 4, pp. 602–612, Jul./Aug. 2020.
[43] Z. Liu, B. Li, Y. Huang, J. Li, Y. Xiang, and W. Pedrycz, "NewMCOS: Towards a practical multi-cloud oblivious storage scheme," *IEEE Trans. Knowl. Data Eng.*, vol. 32, no. 4, pp. 714–727, Apr. 2020.
[44] J. Li *et al.*, "Searchable symmetric encryption with forward search privacy," *IEEE Trans. Depend. Secure Comput.*, early access, Jan. 22, 2019, doi: 10.1109/TDSC.2019.2894411.
[45] B. Perozzi, R. Al-Rfou, and S. Skiena, "DeepWalk: Online learning of social representations," in *Proc. 20th ACM SIGKDD Int. Conf. Knowl. Disc. Data Min.*, 2014, pp. 701–710.

Dongjie Zhu received the Ph.D. degree in computer architecture from Harbin Institute of Technology, Harbin, China, in 2015.

He is an Assistant Professor with the School of Computer Science and Technology, Harbin Institute of Technology, Wehai, China. His research interests include parallel storage systems and social computing.

Yundong Sun is currently pursuing the Ph.D. degree with the School of Astronautics, Harbin Institute of Technology, Harbin, China.

His research interests include social computing and network embedding.

Haiwen Du is currently pursuing the Ph.D. degree with the School of Astronautics, Harbin Institute of Technology, Harbin, China.

His research interests include storage system architecture and massive data management.

Ning Cao received the Ph.D. degree in computer science from University College Dublin, Dublin, Ireland, in 2015.

He has published over 100 papers in the area of IoT, security, and AI.

Thar Baker (Member, IEEE) received the Ph.D. degree in autonomic cloud applications from Liverpool John Moores University (LJMU), Liverpool, U.K., in 2010.

He is a Reader of cloud engineering with the Department of Computer Science, Faculty of Engineering and Technology, LJMU. He has published numerous refereed research papers in multidisciplinary research areas, including cloud computing, distributed software systems, big data, algorithm design, green and sustainable computing, and autonomic Web science. He has been actively involved as a member of editorial board and review committee for a number of peer-reviewed international journals, and is on programme committee for a number of international conferences.

Dr. Baker was appointed as the Expert Evaluator in the European FP7 Connected Communities CONFINE Project from 2012 to 2015.

Gautam Srivastava (Senior Member, IEEE) received the B.Sc. degree from Briar Cliff University, Sioux City, IA, USA, in 2004, and the M.Sc. and Ph.D. degrees from the University of Victoria, Victoria, BC, Canada, in 2006 and 2012, respectively.

He then taught for three years with the Department of Computer Science, University of Victoria, where he was regarded as one of the top undergraduate professors in the Computer Science Course Instruction. In 2014, he joined a tenure-track position with Brandon University, Brandon, MB, Canada, where he currently is active in various professional and scholarly activities. He was promoted to the rank of Associate Professor in January 2018. He, as he is popularly known, is active in research in the field of cryptography, data mining, security and privacy, and blockchain technology. In his five years as a research academic, he has published a total of 90 papers in high-impact conferences in many countries and in high-status journals (SCI and SCIE) and has also delivered invited guest lectures on big data, cloud computing, Internet of Things, and cryptography at many universities worldwide.

Dr. Srivastava is an Editor of several SCI/SCIE journals. He currently has active research projects with other academics in Taiwan, Singapore, Canada, Czech Republic, Poland, and the USA.