Received November 12, 2020, accepted December 10, 2020, date of publication December 17, 2020,
date of current version December 31, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3045529

Link Prediction Based on Orbit Counting and Graph Auto-Encoder

JIAN FENG AND SHAOJIAN CHEN
College of Computer Science and Technology, Xian University of Science and Technology, Xi'an 710054, China

Corresponding author: Jian Feng (fengjian@xust.edu.cn)

This work was supported in part by the Shaanxi Provincial Natural Science Foundation Project under Grant 2020JM533, and in part by the Chinese Postdoctoral Science Foundation under Grant 2020M673446.

ABSTRACT Link prediction aims to predict the missing edge or the edge that may be generated in the future. The key to link prediction is to obtain the characteristic information with strong representation for nodes. In this work, aiming at the problem that the existing link prediction methods fail to take full account of the high-order connection mode of nodes, we present a new representation learning-based approach called OC-GAE (Orbit Counting and Graph Auto-Encoder) that considers rich subgraph structure around nodes. Firstly, the number of orbits on subgraphs is calculated as the high-order structural features of the nodes; then, the number of orbits is used as the input of Graph Auto-Encoder to learn the efficient representation of the nodes; finally, the network adjacency matrix is reconstructed by the learned representation to realize the link prediction. By comparing with five classical link prediction methods and two mainstream network representation learning methods on four real network datasets, the effectiveness of the proposed method and the prediction accuracy are proved to be optimal in general.

INDEX TERMS Graph auto-encoder, link prediction, orbit counting, representation learning, subgraph.

I. INTRODUCTION

Inferring missing links or predicting future ones based on the currently observed network is known as the link prediction problem, which has tremendous real-world applications, such as predicting user relationships in social networks [1], recommending items to users in recommendation systems [2], [3], predicting drug-target interactions [4], etc.

The most intuitive assumption of link prediction is that the more similar the two nodes are, the more likely they are to have edges. Based on this assumption, the basic problem of link prediction is how to describe and calculate the similarity between nodes. At present, based on different methods of node similarity characterization, link prediction models rely on topological features of the graph and domain-specific attributes of the nodes. The link prediction methods based on node attributes mainly use the external attributes and label information of nodes to describe the similarity between nodes, while the link prediction methods based on network structure measure the structural similarity of nodes based on the network topology. In many cases, the attribute information of nodes has the particularity of domain and application, which is difficult to obtain and has more noise. Compared with node attributes, network structure information is easy to obtain and more reliable, so the link prediction methods based on network structure have been paid more and more attention.

In the link prediction method based on network structure, typically, there are three types of methods: ① method based on local information; ② method based on path information; ③ method based on random walk. Among them, the path information-based method considers higher-order structural information with high computational complexity, while the random walk-based method takes computational efficiency into account with information loss. The local information-based method mainly predicts the link probability of two nodes according to their common neighbors and is mostly used for its calculation similarity and high prediction accuracy. Typical methods include Common Neighbors (CN), Adamic/Adar (AA), Jaccard, Resource Allocation (RA), Preferential Attachment (PA), etc. [5]. However, since only the simple local structure information is considered, the deeper structural relationship between nodes cannot be captured. Take Fig. 1 as an example, here we have 2 networks,

The associate editor coordinating the review of this manuscript and approving it for publication was Donghyun Kim.

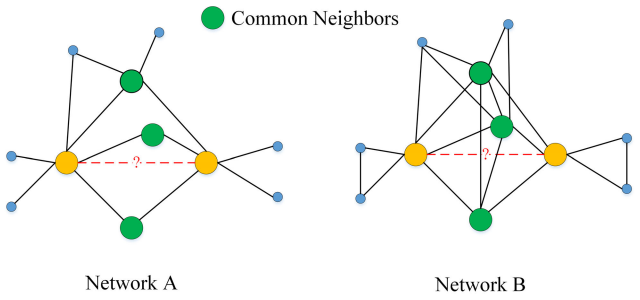


FIGURE 1. Similarity of node pair in two networks.

Network A and Network B. The yellow node pairs in each network have 3 common neighbors, shown in green. If Jaccard algorithm is used to calculate the similarity between yellow nodes, the results will be 3/8, the same in both networks. But obviously, the node pair in Network B is more closely related than the node pair in Network A. It can be seen that the local link prediction method based on common neighbors cannot fully represent the similarity between nodes in some cases, still need more fine-grained topology information.

With the explosive growth of data, traditional topology-based network representation has become a bottleneck and limit large-scale network processing and analysis. With the help of machine learning, especially deep learning technology, the network representation learning technology that embeds high-dimensional network topological space into low-dimensional vector space has attracted great attention in both academic research and industrial applications in recent years [6]. For the link prediction problem, the method based on representation learning first learns the low-dimensional expression of nodes or edges and then uses this low-dimensional expression to predict the possible links between nodes. Typical methods include DeepWalk [7], node2vec [8], GraRep [9], etc. Recently, Graph Convolutional Networks (GCN) has emerged as a powerful tool for representation learning on graphs [10]. GCN has also been successfully applied for link prediction on normal graphs [3], [11]–[13]. However, these network representation learning methods usually only consider the simple first-order or second-order neighborhood features around the nodes when learning the representation vector of nodes or edges, and fail to fully consider the high-order structural features around the nodes. For link prediction, it is especially important to get more representative structural elements as mentioned.

With the deepening of research, researchers found that the graphlets composed of more than three nodes can express the high-order structural characteristics of the network, and often contain more information that can reflect the deep characteristics of nodes [14]. Therefore, in view of the prediction effect of traditional low-order structural features (such as common neighbors) in link prediction tasks, the assumption of this paper is that graphlets contain complex nonlinear patterns, which actually determine the formation of links. Capturing these high-order features is more conducive to predict link formation than low-order structural features. The problems to be solved for link prediction through the high-level structure of the network are: ① how to mine the graphlet structure of nodes to capture higher-order patterns; ② how to design a model to learn the characteristics of graphlet structure.

To address these problems, this paper proposes a link prediction method named OC-GAE (Orbit Counting and Graph Auto-Encoder) based on orbits and GAE with the help of the latest research on graphlet and graph representation learning. Orbit is the basic unit for counting graphlet, which is used to extract high-order features of nodes [14]. The different orbits of each node indicate the potential structural relationship between the node and its neighbors, and express the connection mode between nodes from different perspectives. GAE uses encoding and decoding to learn the structural characteristics of the network [13]. By using GCN as the encoder, GAE can fuse the high-order structure features of nodes themselves and neighbors in a way of message passing, and generate node representation with high expression ability.

The main contributions of the paper can be summarized as follows:

- We consider the higher-order structure of nodes. Unlike existing works that consider the simple structural features around nodes, we take the number of orbits of nodes as features.
- We propose a GAE-based framework OC-GAE for link prediction, which uses encoder-decoder mode to learn the potential representation of the network. To the best of our knowledge, this is the first work to combine the orca algorithm with a neural network model for learning network representations.
- Considering the limitation that traditional GCN can only fuse 1-2 order neighborhood features around nodes, we use the number of orbits of each node as the input of GAE, so that the GCN encoder in GAE can achieve the goal of fusing the higher-order neighborhood features around the nodes.
- Through extensive experiments on multiple real-world datasets, we show the effectiveness of OC-GAE for link prediction on all evaluated datasets.

The rest of this paper is organized as follows: Section 2 introduces the related works; Section 3 introduces the link prediction method based on the orbit counting and GAE; Section 4 shows the experimental results; Section 5 concludes this article and points out the future works.

II. RELATED WORKS

Link prediction is essential to dig out the reasons and driving forces for the connection of network nodes, so as to explain the reasons for the formation of the network structure from a micro perspective. At present, the methods of link prediction can be roughly divided into two categories: traditional methods and methods based on representation learning.

A. TRADITIONAL METHODS

Traditional approaches for link prediction are heuristic methods. They calculate the similarity of nodes through the

structural characteristics of the network or the attribute information of the nodes and then use this similarity to predict whether there is a link between the nodes or not. This type of method is further divided into the following three subcategories: method based on local information, method based on path information, and method based on random walk.

1) METHOD BASED ON LOCAL INFORMATION

This type of method only considers the local information such as one-hop neighbors of the node or the node itself. CN is the most basic method of this kind. The basic idea is that if there are more co-neighbors between two nodes, the probability of connecting edges between them gets bigger. AA extends CN by using the degree of co-neighbors between nodes to measure similarity and then using the logarithm of the degree to distinguish the contribution of common neighbors to node similarity. Similar to AA, RA defines the similarity between two nodes as the number of resources received by the current node from another node. Jaccard defines the similarity by the ratio of the common neighbor between two nodes to the sum of the neighbors of two nodes. PA considers the similarity between nodes to be related to the degree of nodes, regardless of the common neighbors between nodes.

2) METHOD BASED ON PATH INFORMATION

This type of method takes the multi-order path from which the node departs into account. For example, from a path perspective, CN considers the number of second-order paths between two nodes. LP (Local Path) method further considers the third-order path based on CN; and if the number of paths is extended to n-order, it is the Katz method [5]. This kind of method is computationally expensive and complex.

3) METHOD BASED ON RANDOM WALK

This kind of method uses a random walk strategy to obtain the structural information of the network. ACT (Average Commute Time) makes two nodes to walk randomly with each other [5]. The smaller the sum of the average steps required is, the more similar the two nodes are. Cos+ needs to calculate ACT and other information in advance and then calculates the cosine similarity. The disadvantage is that it cannot be used for large-scale networks; RWR (Random Walk with Restart) adds a restart mechanism to the random walk strategy, whereas the literature [15] proposes a modified method SRW (Superposed Random Walk). Although this type of method can reduce the computational complexity, it always misses some information.

B. METHODS BASED ON REPRESENTATION LEARNING

While heuristic methods focused on straightforward similarity measures between pairs of nodes, the representation learning-based methods take the link prediction as a binary classification problem by considering the label of a node pair representing the presence or absence of an edge (link) between the pair. Similar to the traditional approaches, the learning-based methods also exploit graph topological features and attribute information. The majority of the existing research works such as DeepWalk, node2vec, GraRep extract feature sets from the network topology, because topological information is generic and domain-independent. In the early work, features are typical neighborhood and path-based. For example, DeepWalk considers the first-order similarity of nodes by random walk, and node2vec improves DeepWalk by considering the second-order similarity between nodes. To keep the high-order information instead of local topology, the methods such as GraRep learn the low-order spatial representation of the original adjacency matrices by factorization. However, due to the high computational cost for large matrices, this type of method has obvious limitations in large networks.

In order to express high-order structure efficiently, small induced subgraphs called graphlets (motifs) are considered in some researches [16]–[20]. For example, [17] considers the high-order motif features formed by 3-5 nodes, and constructs the classification model by combining the traditional machine learning method of logical regression (LR); HONE constructs the matrix based on motifs, and then uses a generalized Bregman Divergence learns the embedding of nodes and makes link prediction [18]; Role2Vec takes motif count as node attribute, maps nodes with attributes into node types through attributed random walk, and determines whether there are connected edges according to whether adjacent nodes belong to the same type [19]. A new similarity measure based on the motif is proposed in the literature [20]. It is considered that the more motifs shared between two nodes, the higher the quality, and the more similar the two nodes are. But these methods are just a simple application of graphlets, so as to enrich the training data, rather than directly contributing to the representation learning.

Recently, the representation learning-based methods extended from network representation learning to Graph Neural Network (GNN). In GNN, GCN has emerged as a powerful tool by naturally integrating the node information as well as the topological structure, and have also been successfully applied for link prediction [3], [11]–[13].

Based on above analysis, we found that higher-order structural information of nodes often contains more features that can describe the local structure of the network more precisely and play a decisive role in the connection between nodes. In particular, counts of graphlet patterns were shown to be useful for a variety of network analysis tasks and can be computed quickly and efficiently with parallel algorithms. At the same time, as an end-to-end network representation learning technology, GNN can effectively improve the performance of applications. Therefore, this paper proposes a link prediction method based on GNN considering graphlets structure.

III. PROPOSED METHOD

In this section, a list of abbreviations of the definitions and notations used in the paper is firstly given in Table 1, and then next an overview is presented on the proposed framework, following by the details of each model component.

TABLE 1. Descriptions of notations IN OC-GAE.

Notation	Definition and Description
G=(V, E)	An unweighted and undirected network with nodes V and edges E
V	Node set in G
E	Edge set in G
v_i, v_j	The ith and jth nodes of G
N	Number of nodes in G
M	Number of edges
A	The adjacency matrix of G
I	The identity matrix with size N×N
$\widetilde{A}$	$A+I$
G_i	The ith graphlets
O_i	The ith orbits
Z	Node embedding matrix
F	The dimension of embedding vector
$A^{'}$	The symmetric normalized adjacency matrix of G
W_i	The weight matrix in the encoder
D	The degree matrix of G
$\widehat{A}$	The reconstructed adjacency matrix of G
z_i, z_j	The embedding vectors of nodes v_i and v_j
z_{ij}	The inner product of z_i and z_j
y_{ij}	The real edge between nodes v_i and v_j in $\widetilde{A}$
$\widehat{y}_{ij}$	The ijth element in $\widehat{A}$

A. PROBLEM FORMULATION

Like most works on link prediction denoted, we also considered an unweighted and undirected network $G = (V, E)$. $V = \{v_1, v_2, \ldots, v_N\}$ is the node set and $E \subseteq V \times V$ is the link set. The total number of nodes in the network is $N = |V|$, and the total number of edges is $M = |E|$. A link, denoted as a node pair (v_i, v_j) with $v_i, v_j \in V$, represents certain interaction, association, or physical connection between nodes v_i and v_j. The problem of link prediction in a network is simply to predict the possibility of links between two nodes in a network that have not yet been connected by known or potential information in the network.

B. METHOD OVERVIEW

Formally, the link prediction problem is transformed into a binary classification problem, and the classification problem is solved by constructing a learning model.

The first challenge is to choose suitable features for the classification task. We hypothesize that more complex topological features, which look deeper in the structure of the neighborhood of two nodes, are more powerful predictors of link formation than features that rely on shallow structural information such as common neighbors. Different from the existed researches, this work focuses on using graphlets for learning and extracting more useful and meaningful structural information. This allows us to capture suitable fine-grained network features such as neighborhood information of nodes, while also accounting for the global structure.

The second challenge is how to learn representation from features. Here an encoder-decoder model-GAE is used, which is based on GCN to realize node representation through fusing the high-order features of the node itself and its neighbors. The intuition behind the idea of encoding and decoding is that if we can learn to decode high-dimensional graph information from encoded low-dimensional embedding, such as the structure of global or local graph neighborhood of nodes in the graph, then in principle, these embeddings should contain the information needed by downstream machine learning tasks. Putting these together, a new link prediction method OC-GAE combining orbit counting and GAE is proposed.

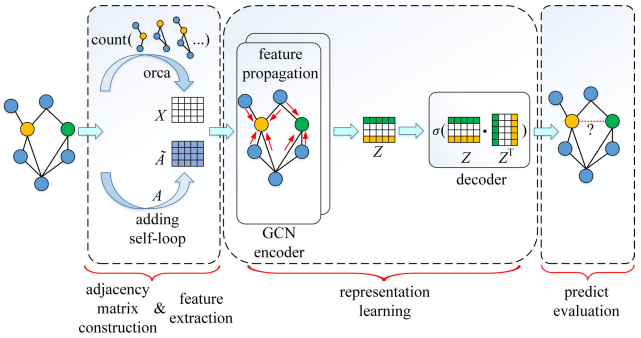


FIGURE 2. Architecture of OC-GAE.

We notice that the literature [21] also considered the importance of subgraphs in network embedding, and it formed an adjacency matrix of subgraphs and then used convex matrix factorization to get subgraphs embeddings directly. But in OC-GAE, we study node embeddings for nodes instead of embeddings for subgraphs.

A simplified illustration of OC-GAE is given in Fig. 2. Firstly, we extract subgraphs for each node of the network in various depths, and then we use the orca algorithm to obtain the different types of orbits on different subgraphs as the characteristics of each node, take the formed characteristic matrix and the adjacency matrix with self-loop of the network as the input, learn the representation of nodes by combining the GAE model realized by GCN, and use the final learned representation to predict the link.

The above scheme highlights the importance of higher-order structures in the network and utilizes the advanced GCN-based GAE model to learn node representations to get better expression ability. The key steps of OC-GAE are adjacency matrix construction, subgraph feature extraction, representation learning, and evaluation, which will be described in detail next.

C. ADJACENCY MATRIX CONSTRUCTION

The adjacency matrix of the network is denoted as A, if nodes $v_i \in V$ and $v_j \in V$ have a link, then $A_{ij} = 1$; otherwise $A_{ij} = 0$. In the subsequent process of representation learning, we need to consider the characteristics of the node itself, so we update the original adjacency matrix A by self-loop.

$$\tilde{A} = A + I \tag{1}$$

where I is the identity matrix with size $N \times N$.

D. FEATURE EXTRACTION

The feature matrix is constructed by taking the number of orbits of a node as its features. Next, we will see what is orbit and how to construct a feature matrix by orbits.

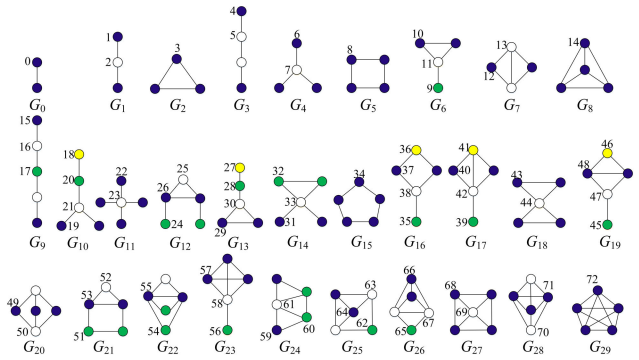


FIGURE 3. Graphlets with 2–5 nodes and automorphism orbits [14].

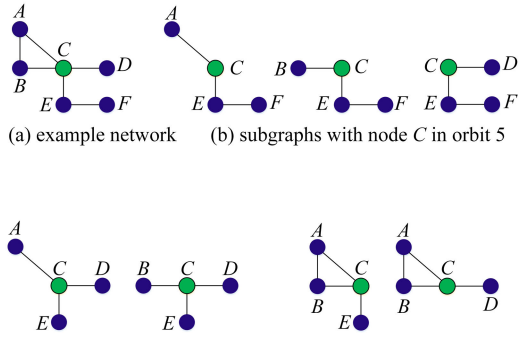


FIGURE 4. An illustration of a partial orbit counts for node C in an example network [14].

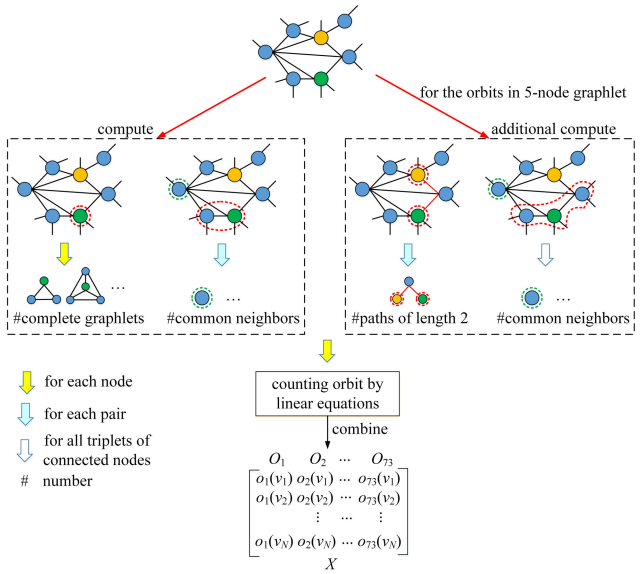


FIGURE 5. Construction of feature matrix.

1) GRAPHLETS AND ORBITS

Graphlets are small, connected, non-isomorphic induced subgraphs [22]. Each graphlet represents a topological pattern of interconnection between k nodes in a graph. There are 9 different graphlets with 2 to 4 nodes and 30 graphlets (G_0-G_{29}) with up to 5 nodes. The number of appearances of graphlets in the network describes the network's structural properties. For a finer description, the nodes of every graphlet are partitioned into a set of automorphism groups called orbits [23]. Two nodes belong to the same orbit if they map to each other in some isomorphic projection of the graphlet onto itself. Nodes of graphlets on 2-5 nodes are grouped into 73 orbits shown by numbers and node colors in Fig. 3.

Orbits define the "roles" of the nodes within a graphlet, express different connection modes between nodes, and contain abundant high-order structural information. For example, in a star on five nodes (graphlet G_{11}), one node represents the center and the remaining four nodes are the leaves; the nodes of the star thus form two different orbits (numbered 23 and 22, respectively). In addition, it plays the role of "central node", "cross node" and "internal node" in a multi-node path. Then we can count how many times a node playing the roles, as shown in Fig. 4.

In the example network (a), for node C, a total of three types of subgraph structures (numbered (b), (c), (d)) are formed. In the structure of (b) type subgraph, node C is in orbit 5, i.e., in this subgraph, node C plays the role of "internal node" in a path composed of 4 nodes; in the structure of (c) type subgraph, node C is in orbit 7, i.e., in this subgraph, node C plays the role of "central node" in a star composed of 4 nodes and 3 sides; in the structure of (d) type subgraph, node C is in orbit 11, i.e., node C plays the role of "central node" in the tail triangle composed of four nodes. Therefore, in this example network, the number of nodes C in orbit 5, 7, and 11 are respectively 3, 2, and 2.

Counting orbits is an expensive operation, as their number grows exponentially in the size of the original graph. However, thanks to recent advances in algorithms, referring to [24] and [25], we are now able to count orbits on graphs with millions of edges by orca. The former constructs a system of equations to count the subgraphs formed by 4 nodes, whereas the latter extends it to the count of larger subgraphs. We leverage this capability to capture complex topological features for the link prediction task in this paper.

2) FEATURE EXTRACTION FROM SUBGRAPHS

The feature matrix is constructed by taking the number of orbits of the node as its features, this gives a finer description of the node's vicinity. Specifically, the number of different orbits on each node is first calculated by the orca and treated as features, then the features of all nodes are combined to form a feature matrix X. The process of the feature matrix construction is shown in Fig. 5, where O_i ($i = 1, 2, 3 \ldots, 73$) represent types of the orbit, and o_i represents the number of nodes on the types of orbit.

The computational process is as follows. Firstly, precalculation is performed. For each node, calculating the number of complete graphlets (for example, G_8 and G_{29} in Fig. 3) and the number of common neighbors of each pair and each connected triplet of nodes. Then, obtaining the number of

73-dimensional orbits of each node by solving the linear equations given by the orca.

E. REPRESENTATION LEARNING

After obtaining the adjacency matrix and the feature matrix of the network, the representation learning is carried out by combining the GAE model. As shown in Fig. 2, the model consists of two parts: Encoder and Decoder of GCN. What Encoder does is to code according to the adjacency matrix and the feature matrix to get the representation matrix Z. Its main function includes extract effective features and fusing the connection features of the nodes and the global features of the network together.

The representation matrix Z obtained by Encoder is:

$$\boldsymbol{Z} = \mathbf{GCN}(\boldsymbol{X}, \tilde{\boldsymbol{A}}) \tag{2}$$

where the size of Z is $N \times F$, and F represents the dimension of embedding vector. GCN Encoder is composed of two layers of convolutions:

$$\mathbf{GCN}\left(\boldsymbol{X}, \tilde{\boldsymbol{A}}\right) = \boldsymbol{A}'\mathbf{ReLU}(\boldsymbol{A}'\boldsymbol{X}\boldsymbol{W_0})\boldsymbol{W_1} \tag{3}$$

where $\boldsymbol{A}' = \boldsymbol{D}^{-\frac{1}{2}}\tilde{\boldsymbol{A}}\boldsymbol{D}^{-\frac{1}{2}}$ is the symmetric normalized adjacency matrix, D is the degree matrix, W_i is the weight matrix of layer i, $\mathrm{ReLU}(\bullet) = \max(0, \bullet)$.

The Decoder is to realize matrix reconstruction by calculating the inner product of representation matrix Z. The reconstructed adjacency matrix is as follows:

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

where $\boldsymbol{\sigma}$ is the sigmoid function. In the reconstruction of the adjacency matrix, each element in $\hat{\boldsymbol{A}}$ is obtained by performing an inner product operation on the representation vector of each node and then using sigmoid operation on the operation result. The values of each element in $\hat{\boldsymbol{A}}$ can be expressed as:

$$\boldsymbol{P}\left(\tilde{\boldsymbol{A}} \mid \boldsymbol{Z}\right) = \prod_{i=1}^{|V|}\prod_{j=1}^{|V|}\boldsymbol{P}(\tilde{\boldsymbol{A}}_{\boldsymbol{ij}}|\boldsymbol{z_i}, \boldsymbol{z_j}) \tag{5}$$

where $\boldsymbol{z_i}, \boldsymbol{z_j}$ represents the representation vectors of nodes v_i and v_j. If v_i and v_j correspond to $\tilde{\boldsymbol{A}}_{\boldsymbol{ij}} = \mathbf{1}$, then the value of the corresponding position in $\hat{\boldsymbol{A}}$ is:

$$\boldsymbol{P}\left(\tilde{\boldsymbol{A}}_{\boldsymbol{ij}} = \mathbf{1} \mid \boldsymbol{z_i}, \boldsymbol{z_j}\right) = \boldsymbol{\sigma}(\boldsymbol{z_i^{\mathrm{T}}}\boldsymbol{z_j}) \tag{6}$$

F. PREDICT EVALUATION

The goal of OC-GAE is to minimize the gap between the reconstructed adjacency matrix $\hat{\boldsymbol{A}}$ and the original matrix A. Therefore, the cross-entropy is used as the loss function in model training, and the parameters are updated by minimizing the error between the predicted value and the real value.

This paper extends the traditional cross-entropy loss function to the weighted cross-entropy loss function by introducing a weighting factor α as the multiplication coefficient for the positive items. The recall rate and the accuracy rate can be balanced by adjusting the value of α. When $\alpha > 1$, the false negative count will be reduced and the recall rate will be increased; when $\alpha < 1$ the false positive count will be reduced and the accuracy will be increased. The loss function is defined as follows:

$$\boldsymbol{L}\left(\boldsymbol{y_{ij}}, \hat{\boldsymbol{y}}_{\boldsymbol{ij}}\right) = \frac{\mathbf{1}}{\boldsymbol{2N}}\sum_{\boldsymbol{i=1}}^{\boldsymbol{N}}\sum_{\boldsymbol{j=1}}^{\boldsymbol{N}}(-\boldsymbol{y_{ij}}\mathbf{log}\left(\boldsymbol{\sigma}\left(\hat{\boldsymbol{y}}_{\boldsymbol{ij}}\right)\right)\boldsymbol{\alpha} - \left(\mathbf{1} - \boldsymbol{y_{ij}}\right)\mathbf{log}(1 - \boldsymbol{\sigma}\left(\hat{\boldsymbol{y}}_{\boldsymbol{ij}}\right))) \tag{7}$$

where $\hat{\boldsymbol{y}}_{\boldsymbol{ij}} = \boldsymbol{z_i^{\mathrm{T}}}\boldsymbol{z_j}$, represents the predicted probability of the connected edges between nodes v_i and v_j, and $\boldsymbol{y_{ij}} \in \{\mathbf{0},\mathbf{1}\}$ represents the real connection values in adjacency matrix $\tilde{\boldsymbol{A}}$. Brings $\boldsymbol{\sigma}\left(\hat{\boldsymbol{y}}_{\boldsymbol{ij}}\right) = \frac{\mathbf{1}}{\mathbf{1}+\boldsymbol{e}^{-\hat{\boldsymbol{y}}_{\boldsymbol{ij}}}}$ to (7) and derives the final loss function as follows:

$$\boldsymbol{L}\left(\boldsymbol{y_{ij}}, \hat{\boldsymbol{y}}_{\boldsymbol{ij}}\right) = \frac{\mathbf{1}}{\boldsymbol{2N}}\sum_{\boldsymbol{i=1}}^{\boldsymbol{N}}\sum_{\boldsymbol{j=1}}^{\boldsymbol{N}}((\mathbf{1} - \boldsymbol{y_{ij}})\hat{\boldsymbol{y}}_{\boldsymbol{ij}} + (\mathbf{1} + (\boldsymbol{\alpha} - \mathbf{1}) \times \boldsymbol{y_{ij}}\mathbf{log}\left(\mathbf{1} + \boldsymbol{e}^{-\hat{\boldsymbol{y}}_{\boldsymbol{ij}}}\right)) \tag{8}$$

The setting of α value follows the literature [13], where $\boldsymbol{\alpha} = \frac{(\boldsymbol{N}\cdot\boldsymbol{N} - \sum_{\boldsymbol{i=1}}^{\boldsymbol{N}}\sum_{\boldsymbol{j=1}}^{\boldsymbol{N}}\boldsymbol{A_{ij}})}{\sum_{\boldsymbol{i=1}}^{\boldsymbol{N}}\sum_{\boldsymbol{j=1}}^{\boldsymbol{N}}\boldsymbol{A_{ij}}}$.

IV. EXPERIMENTATION AND ANALYSIS

A. RESEARCH QUESTIONS

In the experimental design stage, we hope to evaluate the proposed method from the following three aspects and design corresponding experiments.

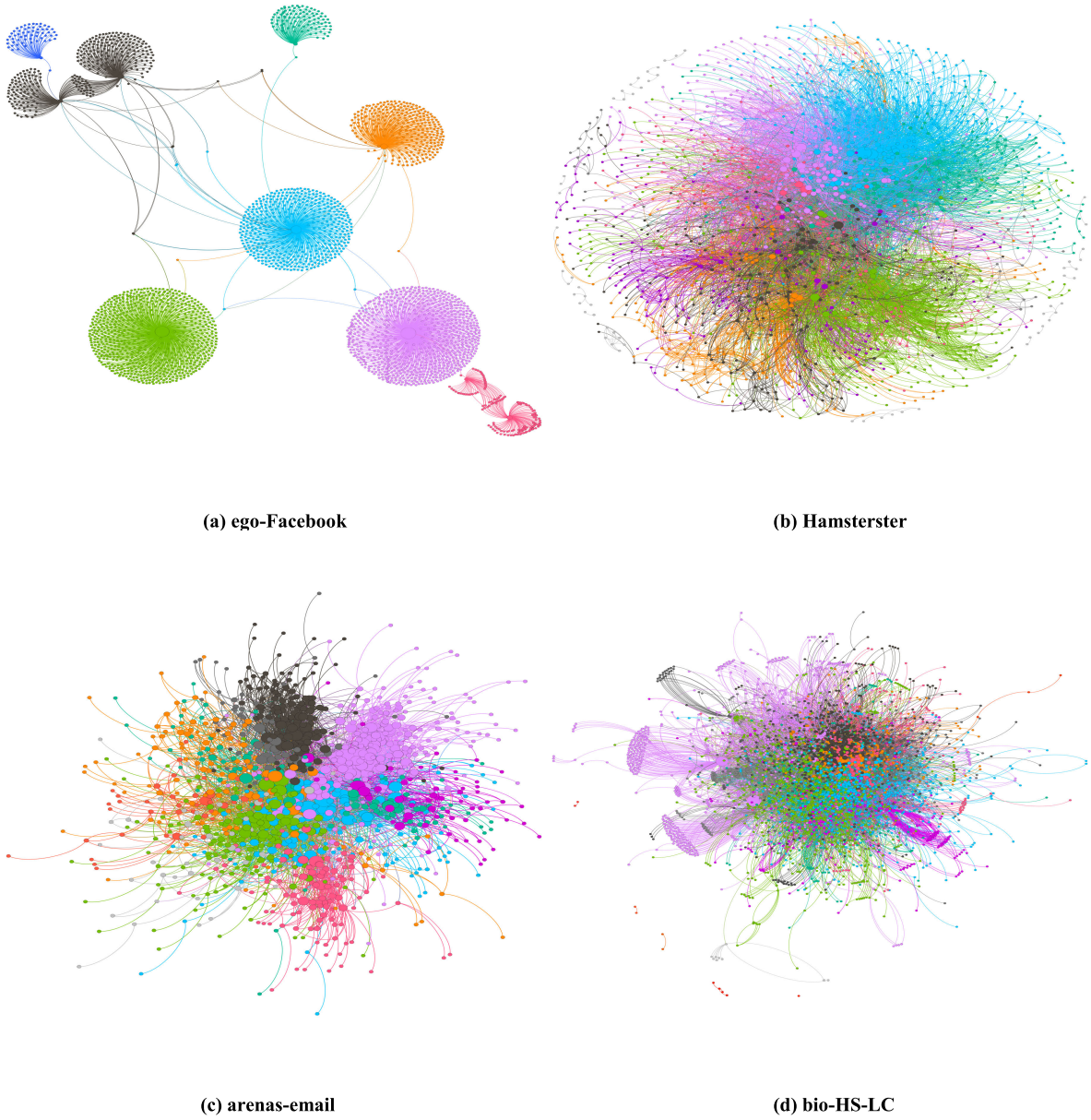
- RQ1: Compared to the traditional method for link prediction, does OC-GAE have a better prediction effect on the whole?
- RQ2: Compared to the classical network representation learning method, does OC-GAE have a better prediction effect on the whole?
- RQ3: What is the convergence performance of this method on each dataset?

In order to answer RQ1, 5 traditional methods for link prediction are compared to OC-GAE. For RQ2, 3 network representation learning methods are compared. And we analyze the convergence performance through the cost value (i.e. the loss value of the objective function) of each epoch in the training stage.

B. DATASETS

4 real attribute networks are selected, and their topologies are shown in Fig. 6.

- ego-Facebook [26]. This is a social network, named ego-Facebook. In the network, each node represents a user, and each edge represents a friend relationship between users.
- Hamsterster [26]. This is a friendship network. In the network, each node represents a user, and each edge represents a friendship between users of the hamsterster.com website.
- arenas-email [26]. This is a communication network, named arenas-email. In the network, each node represents a user, and each edge indicates that at least one message was sent between users.



(a) ego-Facebook

(b) Hamsterster

(c) arenas-email

(d) bio-HS-LC

FIGURE 6. Visualization of networks.

- bio-HS-LC [27]. This is a biological network. In the network, each node represents a gene, and each edge represents a correlation between genes.

Table 2 further gives specific information on these networks, where d is the average degree, D represents the maximum degree.

TABLE 2. Statistic characteristics of the networks.

Networks	$\|V\|$	$\|E\|$	d	D
ego-Facebook	2888	2981	2.0644	769
Hamsterster	1858	12534	13.492	272
arenas-email	1133	5451	9.6222	71
bio-HS-LC	4227	39484	18	397

C. BENCHMARK METHODS

When comparing to the traditional methods for link prediction, 5 methods based on local information are selected, including CN, AA, RA, Jaccard, and PA. This is partial because OC-GAE is also considering the local structure around the node from the perspective of the subgraph structure. More important, these 5 methods are generally adopted

by most of the current research on link prediction problems because they have good performance in many comparative experiments. DeepWalk and GraRep are selected to be comparison methods based on representation learning because they are the most popular methods. Besides, we also compare to Role2Vec, which is similar to our work, and also use orbits to learn the high-order structure. The following is a brief introduction to these methods.

- CN. The basic assumption is: if the number of common neighbors between two nodes is more, the probability of the existence of the connection between the two nodes is greater.
- AA. Improved based on CN. Uses the degree of common neighbors between nodes to calculate similarity, and uses the logarithm of degrees to distinguish the degree of contribution of common neighbors.
- RA. Similar to AA, the difference is that RA does not take the logarithm of node degrees of common neighbors.
- Jaccard. Besides the common neighbors among nodes, the union of neighbors among nodes is also considered.
- PA. This method considers that the similarity between nodes is only related to the degree of nodes.
- DeepWalk. Obtains node sequence by random walk and uses it as the input of the skip-gram for representation learning.
- GraRep. SVD is used to decompose the k-step logarithmic probability transfer matrix to k-step network representation, and the different k-step representation is spliced to obtain a higher-dimensional network representation.
- Role2Vec. Takes subgraph structure as a structural feature and maps it to the node type, so as to use node mapping and attributed random walks to learn the embeddings.

D. PARAMETERS SETTING

In order to carry out the task of link prediction, we randomly remove 10% of the existing node pairs as the positive example for the test set, 5% as the verification set, and the remaining 85% of the existing node pairs as the positive example for the training set. At the same time, we randomly sample the same number of non-connected node pairs as the negative example for test sets and negative examples for training set in the corresponding network. In addition, in order to improve the prediction performance, the feature matrix is standardized. 20 random repeated experiments were carried out on each dataset, and the final results were evaluated by taking the average value.

The experimental parameters are set as follows:

- DeepWalk: Random walk times $\gamma = 10$, length $l = 80$, window size $w = 10$, vector space dimension $d = 128$.
- GraRep: Maximum matrix transition step $k = 4$.
- Role2Vec: Weisfeiler-Lehman features are used as structural features, and the other parameters are set to the default values.

E. EVALUATION INDICATORS

The evaluation indicators adopted in this paper are AP (Average Precision) and AUC (Area Under Curve). AP corresponds to the area under the PR (Precision-Recall) curve, which can reflect the global performance of a method, whereas AUC can measure the accuracy of the method from the overall perspective. The value range of AUC is between [0, 1], the closer to 1, the better the prediction effect. If AUC is less than 0.5, the performance of the method is not as good as that of the random guess.

F. RESULT ANALYSIS

1) COMPARISON WITH TRADITIONAL METHODS OF LINK PREDICTION

In order to answer RQ1, we compare OC-GAE with 5 traditional methods for link prediction, shown in Table 3, where "Improve. (%)" denotes the percentage improvement of the OC-GAE compared to the best performing baseline, "Max-Improve. (%)" denotes the improvement compares OC-GAE with the worst competitor (wave lined), and the bold one represents the best effect on the dataset.

From Table 3, OC-GAE has the best effect on both AUC and AP compared with 5 traditional methods.

In both AUC and AP, CN and Jaccard have similar performance because the two methods both consider the number of common neighbors between nodes; AA and RA have similar performance because the feature taken by the two methods is both the degree of common neighbors between nodes. On bio-HS-LC, PA achieves the optimal result, and OC-GAE is the suboptimal method. Bio-HS-LC is a dense network, and the average degree of nodes is 18. For PA, the similarity between nodes is proportional to the degree of nodes, so the method has better performance in a dense network, so it achieves the best AUC on bio-HS-LC. But OC-GAE is only the suboptimal method because the error of subgraph recognition in a dense network increases relatively. in addition, it is found that CN, jaccard, AA, and RA have achieved poor results on ego-Facebook. The main reason is that these four methods have higher requirements for the density of the network. Generally speaking, OC-GAE has the best prediction effect because of considering a more complex network structure.

2) COMPARISON WITH NETWORK REPRESENTATION LEARNING METHOD

For RQ2, we compare OC-GAE with 3 different network representation learning methods, shown in Table 4.

From Table 4, it can be seen that OC-GAE performed the best overall. Its AUC increases by 31.57%, 25.38%, 11.35% and 12.28% respectively in different networks, and its AP increases by 24.79%, 31.11%, 15.33% and 15.75% respectively. The effect of DeepWalk is the worst because it adopts the random walk method to sample the structural features of the network, so the learned representation vector retains relatively less information. GraRep achieves better results in the datasets of ego-Facebook and bio-HS-LC, mainly because

TABLE 3. Comparison of link prediction performance with traditional methods.

Method	ego-Facebook		Hamsterster		arenas-email		bio-HS-LC	
	AUC	AP	AUC	AP	AUC	AP	AUC	AP
CN	0.4598	0.4997	0.8281	0.8182	0.8658	0.8602	0.8872	0.8868
Jaccard	0.4516	0.4911	0.8094	0.7625	0.8624	0.8589	0.8729	0.8521
AA	0.4673	0.5444	0.8308	0.8296	0.8678	0.8704	0.8898	0.8947
PA	0.9966	0.9942	0.9017	0.8979	0.8309	0.8167	**0.9219**	0.9187
RA	0.4673	0.5444	0.8311	0.8301	0.8672	0.8690	0.8906	0.8961
OC-GAE	**0.9981**	**0.9981**	**0.9054**	**0.9190**	**0.8691**	**0.8831**	0.9059	**0.9244**
Improve. (%)	+0.15%	+0.39%	+0.41%	+2.35%	+0.15%	+1.46%	-	+0.62%
Max-Improve. (%)	+121.01%	+103.23%	+11.86%	+20.52%	+4.59%	+8.01%	+3.78%	+8.48%

TABLE 4. Comparison of link prediction performance with representation learning methods.

Method	ego-Facebook		Hamsterster		arenas-email		bio-HS-LC	
	AUC	AP	AUC	AP	AUC	AP	AUC	AP
DeepWalk	0.9193	0.9027	0.8483	0.8423	0.7805	0.7657	0.8777	0.8820
GraRep	**0.9981**	0.9971	0.8886	0.8842	0.7994	0.7903	**0.9105**	0.9007
Role2Vec	0.7586	0.7998	0.7221	0.7009	0.8439	0.8406	0.8068	0.7986
OC-GAE	**0.9981**	**0.9981**	**0.9054**	**0.9190**	**0.8691**	**0.8831**	0.9059	**0.9244**
Improve. (%)	+0.00%	+0.10%	+1.89%	+3.94%	+2.99%	+5.06%	-	+2.63%
Max-Improve. (%)	+31.57%	+24.79%	+25.38%	+31.11%	+11.35%	+15.33%	+12.28%	+15.75%

GraRep reduces the dimension of global information through matrix decomposition, and its expressiveness is much better than DeepWalk. The reason why OC-GAE achieves the suboptimal effect on the bio-HS-LC dataset is still related to the high density of the network. OC-GAE outperformed Role2Vec in all four datasets. The reason is that Role2Vec obtains the topology information of nodes through random walk, while OC-GAE obtains more expressive embedding by aggregating high-order neighborhood information of nodes by GNN.

G. CONVERGENCE ANALYSIS

To answer RQ3, we analyze the convergence of OC-GAE by plotting the cost value of the objective function in each epoch (i.e. the loss value of the objective function), as shown in Fig. 7. Adam optimizer is used to optimize parameters during training.

It can be seen that after about 20 epochs, the value of the objective function of the other 3 datasets, except for ego-Facebook, declines fast, and then gradually begins to converge. For these 3 datasets, about 200 epochs are needed for convergence. For ego-Facebook, with the increase of epoch, the cost value of the objective function has been decreasing, and it has not yet started to converge at 200 epochs. The reason is that the initial learning rate is set relatively low (1e-6), so the parameter update on this dataset is relatively slow. We tried to increase the initial learning rate to reach the convergence state early, but it is shown that the final prediction effect is not ideal, and the initial low learning rate (1e-6) can get better prediction performance. Therefore, combined with the above model convergence analysis, we set the number of model training iterations to 200.

H. COMPEXITY ANALYSIS

To analyze the complexity of OC-GAE, we recorded the time consumption of all methods used in the comparison. Take arenas-email as an example, Fig. 8 shows one running time of all methods in seconds.

From Fig. 8, the traditional methods are based on the statistical characteristics of the local structure of nodes to predict the link directly, the calculation is simple, and so the running time is short. Because the representation learning methods need to calculate the node representation first, and then make link prediction, so take a long time. Among them, DeepWalk and Role2Vec based on random walk have the longest running time because they need to repeat random walk many times and then calculate a large number of paths between nodes. GraRep obtains the node representation

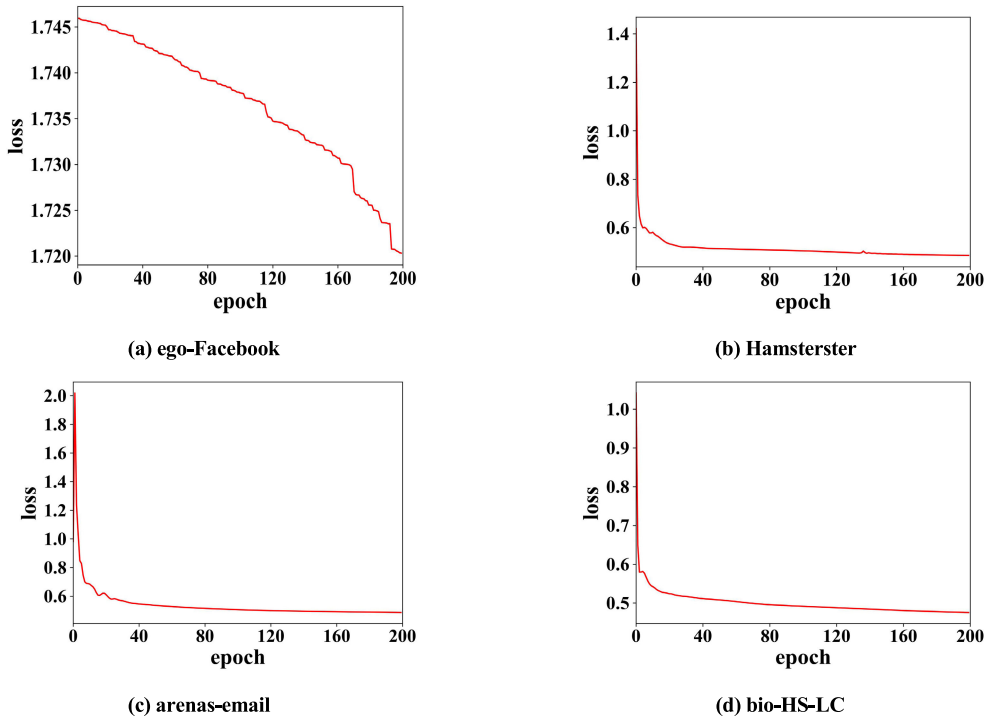


FIGURE 7. Convergence of objective function on each dataset.

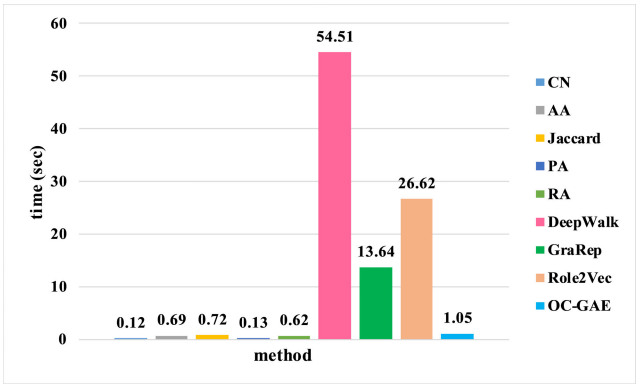


FIGURE 8. One runtime of each method on arena-email.

by matrix decomposition, and its computational complexity is also very high. On the basis of the orca algorithm, the OC-GAE calculates the node representation by GAE with few layers, which makes the running time much shorter than other representation learning methods.

To sum up, compared with the traditional link prediction methods or the typical network representation learning methods, the performance of OC-GAE is the best overall.

V. CONCLUSION

Aiming at the problem that the existing link prediction method only considers the simple structural characteristics around the nodes, this paper proposes a fusion subgraph orbit counting and GAE method OC-GAE for link prediction. The experimental results show that the prediction effect of OC-GAE is better than the existing methods, it is shown that the high-order structure features are more conducive to predicting the link formation than the low-order ones.

Still, there are some shortcomings in OC-GAE, such as the lack of adaptability in the high-density network. Therefore, the following possible research directions include the feature construction of nodes, such as increasing weight attenuation and considering the features of edge structure. By fusing the features of node structure and edge structure, more comprehensive attribute information can be obtained, so that the prediction performance can be further improved.

REFERENCES

[1] P. Xu, W. Hu, J. Wu, and B. Du, "Link prediction with signed latent factors in signed social networks," in *Proc. SIGKDD*, Anchorage, AK, USA, Jul. 2019, pp. 1046–1054.

[2] Z. Su, X. Zheng, J. Ai, Y. Shen, and X. Zhang, "Link prediction in recommender systems based on vector similarity," *Phys. A, Stat. Mech. Appl.*, vol. 560, Dec. 2020, Art. no. 125154, doi: 10.1016/j.physa.2020.125154.

[3] J. Zhang, X. Shi, S. Zhao, and I. King, "STAR-GCN: Stacked and reconstructed graph convolutional networks for recommender systems," in *Proc. IJCAI*, Macao, China, Aug. 2019, pp. 4264–4270.

[4] Y. Lu, Y. Guo, and A. Korhonen, "Link prediction in drug-target interactions network using similarity indices," *BMC Bioinf.*, vol. 18, no. 1, p. 39, Jan. 2017, doi: 10.1186/s12859-017-1460-z.

[5] A. Kumar, S. S. Singh, K. Singh, and B. Biswas, "Link prediction techniques, applications, and performance: A survey," *Phys. A, Stat. Mech. Appl.*, vol. 553, Sep. 2020, Art. no. 124289, doi: 10.1016/j.physa.2020.124289.

[6] J. Zhang, J. Zheng, J. Chen, and Q. Xuan, "Hyper-substructure enhanced link predictor," in *Proc. 29th ACM Int. Conf. Inf. Knowl. Manage.*, Virtual Event, Ireland, Oct. 2020, pp. 2305–2308.

[7] B. Perozzi, R. Al-Rfou, and S. Skiena, "DeepWalk: Online learning of social representations," in *Proc. SIGKDD*, New York, NY, USA, 2014, pp. 701–710.

[8] A. Grover and J. Leskovec, "Node2vec: Scalable feature learning for networks," in *Proc. SIGKDD*, San Francisco, CA, USA, 2016, pp. 855–864.
[9] S. S. Cao, W. Lu, and Q. K. Xu, "Grarep: Learning graph representations with global structural information," in *Proc. CIKM*, Melbourne, VIC, Australia, 2015, pp. 891–900.
[10] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. ICLR*, Toulon, France, 2017, pp. 1–14.
[11] M. H. Zhang and Y. X. Chen, "Link prediction based on graph neural networks," in *Proc. NIPS*, Montreal, QC, Canada, 2018, pp. 5171–5181.
[12] M. Schlichtkrull, T. N. Kipf, P. Bloem, R. Berg, I. Titov, and M. Welling, "Modeling relational data with graph convolutional networks," in *Proc. ESWC*, Heraklion, Greece, 2018, pp. 593–607.
[13] T. N. Kipf and M. Welling, "Variational graph auto-encoders," Nov. 2016, *arXiv:1611.07308*. [Online]. Available: http://arxiv.org/abs/1611.07308
[14] T. Hočevar and J. Demčar, "Computation of graphlet orbits for nodes and edges in sparse graphs," *J. Stat. Softw.*, vol. 71, no. 10, pp. 1–24, 2016, doi: 10.18637/jss.v071.i10.
[15] W. Liu and L. Lü, "Link prediction based on local random walk," *Europhys. Lett.*, vol. 89, no. 5, p. 58007, Mar. 2010, doi: 10.1209/0295-5075/89/58007.
[16] L. Xu, L. Huang, and C. D. Wang, "Motif-preserving network representation learning," *J. Frontiers Comput. Sci. Technol.*, vol. 13, no. 8, pp. 1261–1271, Nov. 2018, doi: 10.3778/j.issn.1673-9418.1807041.
[17] G. AbuOda, G. D. F. Morales, and A. Aboulnaga, "Link prediction via higher-order motif features," in *Proc. ECML PKDD*, Würzburg, Germany, 2019, pp. 412–429.
[18] R. A. Rossi, N. K. Ahmed, and E. Koh, "Higher-order network representation learning," in *Proc. WWW*, Geneva, Switzerland, 2018, pp. 3–4.
[19] N. Ahmed, R. A. Rossi, J. Lee, T. Willke, R. Zhou, X. Kong, and H. Eldardiry, "Role-based graph embeddings," *IEEE Trans. Knowl. Data Eng.*, early access, Jun. 2, 2020, doi: 10.1109/TKDE.2020.3006475.
[20] F. Aghabozorgi and M. R. Khayyambashi, "A new similarity measure for link prediction based on local structures in social networks," *Phys. A, Stat. Mech. Appl.*, vol. 501, pp. 12–23, Jul. 2018, doi: 10.1016/j.physa.2018.02.010.
[21] Z. Cao, L. L. Wang, and G. D. Melo, "Link prediction via subgraph embedding-based convex matrix completion," in *Proc. AAAI*, New Orleans, LA, USA, 2018, pp. 2803–2810.
[22] N. Przulj, D. G. Corneil, and I. Jurisica, "Modeling interactome: Scale-free or geometric?" *Bioinformatics*, vol. 20, no. 18, pp. 3508–3515, Dec. 2004, doi: 10.1093/bioinformatics/bth436.
[23] R. A. Rossi, R. Zhou, and N. K. Ahmed, "Deep inductive graph representation learning," *IEEE Trans. Knowl. Data Eng.*, vol. 32, no. 3, pp. 438–452, Mar. 2020, doi: 10.1109/TKDE.2018.2878247.
[24] T. Kloks, D. Kratsch, and H. Müller, "Finding and counting small induced subgraphs efficiently," *Inf. Process. Lett*, vol. 74, nos. 3–4, pp. 115–121, May 2000, doi: 10.1016/S0020-0190(00)00047-8.
[25] M. Kowaluk, A. Lingas, and E.-M. Lundell, "Counting and detecting small subgraphs via equations," *SIAM J. Discrete Math.*, vol. 27, no. 2, pp. 892–909, Jan. 2013, doi: 10.1137/110859798.
[26] J. Kunegis, "KONECT: The Koblenz network collection," in *Proc. WOW*, Rio de Janeiro, Brazil, 2013, pp. 1343–1350.
[27] R. A. Rossi and N. K. Ahmed, "The network data repository with interactive graph analytics and visualization," in *Proc. AAAI*, Austin, TX, USA, 2015, pp. 4292–4293.

JIAN FENG was born in Xi'an, Shaanxi, China, in 1973. She received the Ph.D. degree in computer software and theory from Northwest University, Xi'an, China, in 2008. Since 2010, she has been an Assistant Professor with the College of Computer Science and Technology, Xi'an University of Science and Technology, Xi'an, China. She is the author of two books and more than 30 articles. Her research interests include computer network and communication, network security, and distributed computing. She holds three patents.

SHAOJIAN CHEN was born in Putian, Fujian, China, in 1997. He received the B.S. degree in network engineering from the Xian University of Science and Technology, Xi'an, China, in 2019, where he is currently pursuing the M.S. degree. His current research interests include graph machine learning and link prediction.