EURASIP Journal on Wireless Communications and Networking

RESEARCH Open Access

Weighted enclosing subgraph-based link prediction for complex network

Weiwei Yuan[1,2], Yun Han[1], Donghai Guan[1*], Guangjie Han[3], Yuan Tian[4], Abdullah Al-Dhelaan[5] and Mohammed Al-Dhelaan[5]

*Correspondence: dhguan@nuaa.edu.cn

[1] College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, China
[2] Collaborative Innovation Center of Novel Software Technology and Industrialization, Nanjing, China
[3] Department of Information and Communication Engineering, Hohai University, Nanjing, China
[4] School of Computer Engineering, Nanjing Institute of Technology, Nanjing, China
[5] Department of Computer Science, King Saud University, Riyadh, Saudi Arabia

Abstract

Link prediction is a fundamental research issue in complex network, which can reveal the potential relationships between users. Most of link prediction algorithms are heuristic and based on topology structure. Weisfeiler–Lehman Neural Machine (WLNM), regarded as a new-generation method, has shown promising performance and thus got attention in link prediction. WLNM extracts an enclosing subgraph of each target link and encodes the subgraph as an adjacency matrix. But it does not consider the relationship between other links of the enclosing subgraph and target links. Therefore, WLNM does not make full use of the topology information around the link, and the extracted enclosing subgraph can only partially represent the topological features around the target link. In this work, a novel approach is proposed, named weighted enclosing subgraph-based link prediction (WESLP). It incorporates the link weights in the enclosing subgraph to reflect their relationship with the target link, and the Katz index between nodes is used to measure the relationship between two links. The prediction models are trained by different classifiers based on these weighted enclosing subgraphs. Experiments show that our proposed method consistently performs well on different real-world datasets.

Keywords: Weighted subgraph, Graph coding, Link prediction, Complex network

1 Introduction

Link prediction is a fundamental research issue of complex network analysis [1]. It has been used in many applications, such as friend recommendation in social networks [2], product recommendation in e-commerce [3], completion of knowledge maps [4], interactions between proteins, and recovery of missing reactions in metabolic networks [5]. Existing works predict link via the similarities of nodes: The higher similarity two nodes have, the more probably there exists a link between them.

Node similarity has different measurements, and network topology-based measurements are most popular. For instance, the node similarity is measured via the relationship of their neighborhood in the network in a number of works. In [5], the node similarity of two nodes is measured via their common neighbors. It assumes that if two nodes have more common neighbors, they are more likely to have a link. AA [2], PA [6, 7], RA [6, 8] are the extension of [5], which calculate the similarity between nodes based

on first-order or second-order neighbors between two nodes. They perform well in practice and are more interpretable [1].

However, a significant limitation of traditional methods is that they fail to make full use of the network topology information. And they are also lack of generalization ability. For example, the idea of common neighbors performs well in social networks but bad in power grids and biological networks [5]. To overcome the shortcomings of traditional methods, Zhang and Chen [9] recently proposed the Weisfeiler–Lehman Neural Machine (WLNM), which learns topological features through enclosing subgraphs of links. By extracting subgraph patterns for each target link, this method learns the subgraph pattern that promotes the formation of a link and codes the subgraph as an adjacency matrix. WLNM trains neural networks on these adjacency matrices to learn the link prediction model. In WLNM, subgraph coding is an important step. Because the machine learning model reads data sequentially, stable ordering based on the role of the node structure is essential in order to lean the meaningful models [10]. Subgraph coding establishes a mapping from graph representation to matrix representation, which ensures the nodes with similar structural characteristics to be mapped to similar positions in adjacency matrix. Enclosing subgraph of the target link is coded by the Weisfeiler–Lehman (WL) algorithm [11], which determines vertex ordering based on network topology information.

Existing methods use node proximity to predict the link between nodes [1, 5, 12]. WLNM is different, and it learns the subgraph pattern that promotes the link by extracting the enclosing subgraph of the target link. Compared to the traditional link prediction algorithms, WLNM has better generalization ability and prediction performance in social networks, power grids, and biological networks. However, WLNM still has a limitation in link analysis in subgraphs. After extracting the enclosing subgraph of the target link, if the subgraph is directly mapped to the adjacency matrix, the obtained adjacency matrix cannot reflect the relationship between the links in the subgraph and the target link. Much topology information that affects the performance of link prediction will be missed. The links in the enclosing subgraph have different relationships to the target link. An example is shown in Fig. 1. It is the enclosing subgraph of the target link (a, b). It can be observed that although the links (b, c) and (b, g) are directly adjacent to the target link, the node c is directly connected to the two end nodes of the target link and forms a triangle. The node g is not directly connected to them and farther away from node a. This indicates that the link

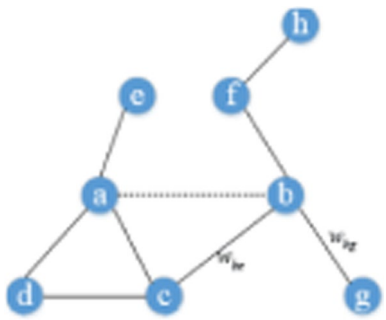

Fig. 1 An enclosing subgraph of the target link (a, b)

(b, c) has a closer relationship to the target link than (b, g). Adding weights to the links in the enclosing subgraph to reflect their relationship to the target link should be considered. For example, in Fig. 1, w_{bc} and w_{bg} are the weights of links (b, c) and (b, g), respectively, and w_{bc} is greater than w_{bg}.

In order to solve the problem in the existing works, an improved link prediction method is proposed: weighted enclosing subgraph-based link prediction (WESLP). It adds weights to links in the enclosing subgraph according to their relationship to the target link. WESLP extracts the enclosing subgraph of each target link in the network and encodes the nodes of the extracted enclosing subgraph with the optimized WL algorithm. To reflect the relationship between the link in the enclosing subgraph and the target link, WESLP assigns weights to the links in the subgraph. We assume that the closer the relation between the links of the enclosing subgraph and the target link, the higher the weights of links and vice versa. WESLP uses the Katz index [13] between nodes to define the relationship between links. Then, WESLP adds weights to the non-target links in the subgraph based on their relationship with the target link. Finally, the weighted subgraph is mapped to a weight adjacency matrix as training data for the learning machine. Compared with the adjacency matrix of WLNM, the weight adjacency matrix generated by WESLP can better reflect the local characteristics around the target link.

The main contributions of this paper are: (1) WESLP uses Katz index between nodes to define weights to the links in the extracted enclosing subgraphs to reflect their relationship with the target link. (2) We optimize the WL algorithm and apply the optimized WL algorithm to our proposed method. (3) WESLP improves the link prediction performance, which has been verified through conducting extensive experiments on different datasets.

The rest of this paper is organized as follows: Sect. 2 introduces the related works; Sect. 3 presents the proposed link prediction method in details; Sect. 4 gives the experimental results of the proposed method; and Sect. 5 concludes this paper and points out the future directions.

2 Methods/experimental

In this paper, we study the link prediction problem based on enclosing weighted subgraphs of links. Specially, we extract the enclosing subgraph for the target link and use the graph coding algorithm to obtain the coding for the subgraph node and add weights according to the relationship between each link and the target link in the subgraph. The algorithms we use include improved graph coding algorithm and weighting methods that exploit the Katz index between nodes.

Our experimental environment is 64 bit Windows 7 system with Intel (R) Core i7-8700k CPU @ 3.70 GHz and the RAM of 32G. The programming language is Python on PyCharm. We execute experiments comparing several popular methods on several commonly used datasets. A large number of experiments have shown that our method can achieve better results in predicting links.

3 Related works

A network can be represented as a graph $G = (V, E)$, where $V = \{v_1, \ldots, v_n\}$ is the set of vertices and $E \subseteq V \times V$ is the set of links. e_{ij} represents the link between node v_i and v_j, $e_{ij} \in E$. An adjacency matrix A is used to represent G. If G is an unweighted network,

$a_{ij} = 1$ if there is a link connecting v_i and v_j, otherwise $a_{ij} = 0$. v_i and v_j are called to be adjacent if $a_{ij} = 1$. If G is a weighted network, $a_{ij} = w_{ij}$ if there is a link connecting v_i and v_j with a weight w_{ij}, otherwise $a_{ij} = 0$. $N_d(v_i)$ represents a set of vertices whose distance to v_i is no longer than d, in which d is an integer. For example, $N_1(v_i)$ is the set of vertices within 1-hop away from v_i in G. $d(v_i, v_j)$ represents the shortest path between v_i and v_j.

Link prediction is a basic task of network mining. For a target link e_{ij}, the link prediction task predicts a_{ij} for e_{ij}. Link prediction methods predict links via the similarity of the two end nodes of the target link [1, 5]. The higher similarities two nodes share, the more probably there exists a link connecting these two nodes. Some classical node similarity measurements used in link prediction models include Preferential Attachment (PA), Common neighbors (CN), Adamic–Adar (AA), and Resource Allocation (RA). PA [6, 7] measures the existence of a link by the product of its node pairs' degrees. If two end nodes of a link have higher degree product, i.e., these two nodes connect to more nodes in the network, it is more likely that there exists a link between these two nodes. PA is defined as:

$$PA(v_i, v_j) = |N_1(v_i)| \cdot \left|N_1(v_j)\right|, \tag{1}$$

where $N_1(v_i)$ and $N_1(v_j)$ are the neighbors of v_i and v_j, respectively, and $|\cdot|$ is the number of $\cdot$. CN [5] measures the existence of a link by the number of its node pairs' common neighbors. If two end nodes of a link share more common neighbors, they are regarded to have more topology similarity in the network, and it is more likely that there exists a link between these two nodes. CN is defined as:

$$CN(v_i, v_j) = \left|N_1(v_i) \cap N_1(v_j)\right|. \tag{2}$$

Unlike CN, if two end nodes of a link share more common neighbor, AA [2] regards; there is less likely to exist a link between these two nodes:

$$AA(v_i, v_j) = \sum\nolimits_{z \in N_1(v_i) \cap N_1(v_j)} \frac{1}{\log D(z)}, \tag{3}$$

where $D(z)$ is degree of the selected common neighbor. RA [6, 8] is similar as AA, while changes the degree on how the number of two end nodes' common neighbors influences the existence of the link:

$$RA(v_i, v_j) = \sum\nolimits_{z \in N_1(v_i) \cap N_1(v_j)} \frac{1}{D(z)}. \tag{4}$$

These link prediction methods based on first-order or second-order neighbors similarity between two nodes tend to lose a large amount of network topology information, resulting in poor prediction performance and poor scalability. A new approach to predicting links has been opened up with recent research on enclosing subgraphs of links. Specifically, the enclosing subgraph of the target link is first extracted, and then, the extracted enclosing subgraph is encoded. Encoding each enclosing subgraph is a key issue. The graph coding algorithm enables the machine learning model to read the data orderly [10]. A stable order based on the structural characteristics of the node is essential to the machine learning model.

A graph coding algorithm is a mapping f: $V \rightarrow C$ from vertices V to an ordered set C. C is usually a set of integer coding starting from 1. If f is injective, C can determine the vertex order in an adjacency matrix. The Weisfeiler–Lehman (WL) algorithm [11] is a newly proposed graph coding algorithm which has been verified to be effective in the real applications. The key idea of WL is to iteratively augment the vertex coding using their neighbor's coding and compress the augmented coding into new coding before convergence [11]. Algorithm 1 illustrates the details of WL.

Specifically, each vertex constitutes the signature string by connecting its own coding and its neighbor's sorted coding. Then, the vertices are sorted in an ascending order of the signature string, and assigned new coding 1, 2, 3, $\cdots$. Vertices with the same signature string get the same coding. Figure 2 gives an example of two iterations of the WL algorithm. The vertices in Fig. 2: (1) are initially coded 1. In each iteration, Step 1: Each vertex constitutes a signature string by connecting its own coding and its neighbor's sorted coding; Step 2: The coding of the vertex is updated according to the string order ascending in lexicographic order. Take the two vertices v_i and v_j as an example, vertex v_i has coding 1 and its neighbors have coding $\{1, 1\}$, respectively. Simultaneously, v_j has coding 1 and its neighbors have coding $\{1, 1, 1\}$. The signature strings for v_i and v_j are $1, 11$ and $1, 111$, respectively, because $1, 11$ is smaller than $1, 111$ lexicographically. v_j is assigned a smaller coding than v_i in the next iteration. We repeat Step1, 2 until the coding of the nodes converges, i.e., their coding stops changing.

One key structure-coding property of the WL algorithm is that vertices with the similar coding share the similar structural role across different graphs [11]. For example, if vertex v in G and vertex v' in G' have similar structural roles in their respective graphs, they will have similar relative positions in their respective rankings. Another graph is shown in Fig. 3 after executing Algorithm 1. Comparing Figs. 2 and 3, it shows that if vertices have similar structural roles in their respective graphs, they will have similar relative positions in their respective rankings. The structure-coding property of WL is

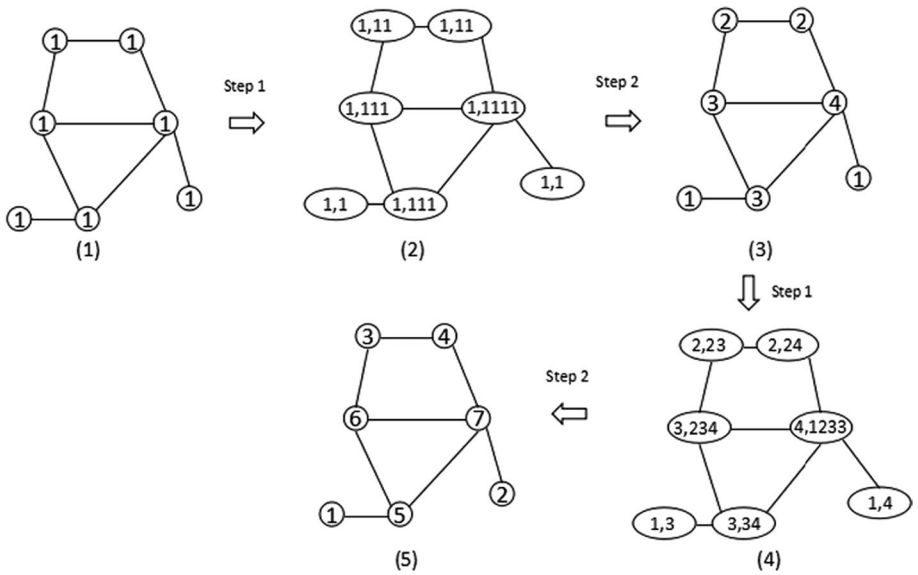

Fig. 2 The executive process of WL with two iterations

essential for its success in graph kernel design [11], which measures graph similarity by counting matching WL codings. According to the similarity between the subgraphs extracted by the known links in the network, the learning machine learns the subgraph pattern that constitutes the links through the matrices that subgraphs map and then constructs the link prediction model. The stable order based on structural role of node is critical to the machine learning model [10].

Algorithm 1 The Weisfeiler-Lehman Algorithm

Input: Initial coding $c_0(v) = 1$ for all nodes $v \in V$, graph $G = (V, E)$
Output: Final coding $c(v)$ for all $v \in V$
Let $c(v) = c_0(v)$ for all $v \in V$
while $c(v)$ has not converged **do**
  for each $v \in V$ **do**
    Assign a multiset-coding $M(v')$ which consists of its neighbors' coding $\{c(v')|v' \in N_1(v)\}$
    Sort elements in $M(v')$ in an ascending order and concatenate them into a string $s(v')$
    Add $c(v)$ as a prefix to $s(v')$ and call the resulting string $s(v) = [c(v), s(v')]$
  end for
  Resort all of the strings $s(v)$ for all v from $G = (V, E)$ in lexicographical ascending order
  Recoding all $s(v)$ to new coding 1,2,3,... sequentially; same strings will get the same coding
end while

4 The proposed model

The proposed method assigns weights to the links (except target link) in the k-enclosed subgraph of the extracted target links to reflect their relationship to the target links. Specifically, a k-enclosing subgraph is firstly extracted for each target link. The nodes of the k-enclosing subgraph are then encoded by the proposed optimized WL algorithm. Each link of the k-enclosing subgraph is assigned a weight reflecting its relationship with the target link. This weight is assigned according to the Katz index between nodes. The weighted k-enclosing subgraphs are then used to measure the similarity of target links for the link prediction task. The framework of the proposed method is given in Fig. 4, in which the rectangles represent the functional modules and parallelograms represent the data [14]. The input of the proposed model is adjacency matrix of the network and target links, and the outputs are the predicted ratings. The proposed method has four phases: Enclosing Subgraph Generation phase, which generates a k-enclosing subgraph for each target link; Enclosing Subgraph Coding phase, which codes each generated enclosing subgraph with the WL algorithm; Enclosing Subgraph Weighting phase, which assigns weights to links of each coded enclosing subgraph; and Link Classification phase, which

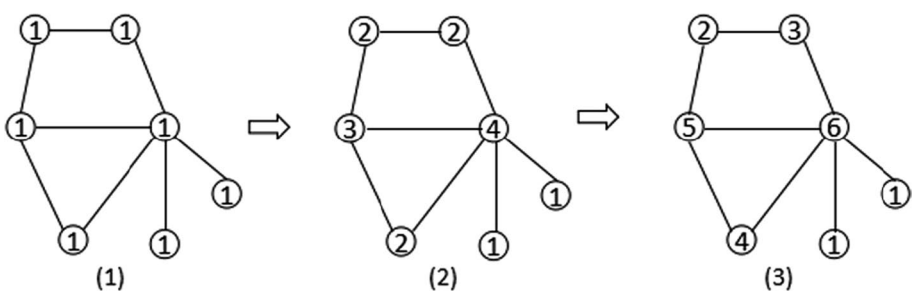

Fig. 3 The executive process of the WL algorithm for another similar graph

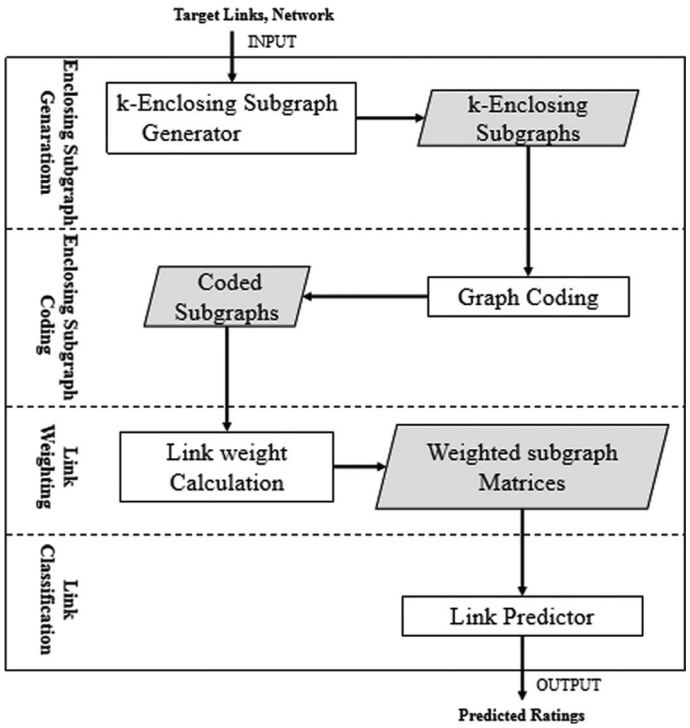

Fig. 4 The framework of the proposed method

classifies links with the weighted enclosing subgraph the each link. The details of the proposed method are given as follows.

4.1 k-enclosing subgraph generation

A k-enclosing subgraph is firstly generated for each target link in the proposed method. The k-enclosing subgraph of a link is a subgraph composed by the neighbor nodes of this link, and the size of the neighbor nodes is k. The enclosing subgraph describes the topology information surrounding the target link. It discovers the local patterns inducing the existence of links between nodes. For a link e_{ij}, its enclosing subgraph is represented as $SG(e_{ij})$ is generated by adding the neighborhood nodes iteratively, as shown in Algorithm 2: v_i and v_j are firstly added to $SG(e_{ij})$; the d-order neighbor nodes of v_i and v_j, which are represented by $N_d(v_i)$ and $N_d(v_j)$, $d \in N$, are then added to $SG(e_{ij})$ according to the ascending order of d until $\left|SG(e_{ij})\right| \geq k$. When $\left|SG(e_{ij})\right| > k$, nodes lastly added to $SG(e_{ij})$ are removed until $\left|SG(e_{ij})\right| = k$. The k-enclosing subgraph of e_{ij} is represented as $SG(e_{ij}^k)$.

4.2 k-enclosing subgraph coding

This phase codes the nodes of $SG(e_{ij}^k)$ by an optimized WL algorithm to a node sequence. This allows machine learning models to read vertices of k-enclosing subgraphs orderly in a stable order and makes each target link always in the $a_{1,2}$ in the adjacency matrix SA of its k-enclosing subgraph.

The node sequence generated directly by the existing WL algorithm cannot differentiate the two end nodes of the target link. The WL algorithm is an iterative graph coding

algorithm. It is coding-order preserving: Given any two vertices v_i and v_j of a network, if the coding of v_i is smaller than the coding of v_j in one iteration, then the coding of v_i is still smaller than the coding of v_j in the next iteration. Since the WL algorithm initializes nodes with the same value, the coding of the two end nodes of the target link cannot be differentiated. This will result in the target link being in training, making the trained machine learning model meaningless.

The proposed method optimizes the existing WL algorithm by differentiating the initial coding of nodes. The initial codes of nodes are generated based on the ascending order of their distance to the target link. The WL algorithm is then used to update the node coding. In this case, the two end nodes of each target link are distinctively identified.

Three rules are formulated for nodes initialization: (1) The coding of the two end nodes v_i and v_j of the target link is set to be 1, i.e., $c(v_i) = c(v_j) = 1$; (2) the coding of a node v_x of the subgraph is the sum of its distance to v_i and v_j, i.e., $c(v_x) = d(v_x, v_i) + d(v_x, v_j)$, where $d(\cdot, \cdot)$ is the distance between two involved nodes; and (3) two nodes have the same coding if they have the same distance to the two end nodes of the target link, respectively, i.e., if $d(v_x, v_i) = d(v_y, v_i)$ and $d(v_x, v_i) = d(v_y, v_i)$, then $c(v_x) = c(v_y)$. Based on the above rules given, initialize the codings of the two end nodes v_i and v_j to 1; if a node v_x with $(d(v_x, v_i), d(v_x, v_j)) = (1, 1)$, set coding $c(v_x) = 2$; nodes with (1, 2) or (2, 1) get coding 3; nodes with (1, 3) or (3, 1) get coding 4; nodes with (2, 2) or (2, 2) get coding 5, so and so forth. The following hash function can be constructed to be used to calculate the coding fitting the above three rules:

$$c(v_x) = 1 + \min(d(v_x, v_i), d(v_x, v_j)) + IQ(d, 2)[IQ(d, 2) + R(d, 2) - 1], \tag{5}$$

where d is sum of $d(v_x, v_i)$ and $d(v_x, v_j)$, and $IQ(\cdot, \cdot)$ and $R(\cdot, \cdot)$ are the integer quotient and remainder of the involved two numbers, respectively.

Algorithm 2 Enclosing Subgraph Generation

Input: The adjacency matrix of G, the target link e_{ij}
Output: The enclosing subgraph $SG(e_{ij}^k)$ of e_{ij}
Parameter: Subgraph size k

$SG(e_{ij}^k) = \{v_i, v_j\}$
$border = \{v_i, v_j\}$
while $|SG(e_{ij}^k)| < k$ **do**
    $border = (\cup_{v \in border} N_1(v)) - SG(e_{ij}^k)$
    $SG(e_{ij}^k) = SG(e_{ij}^k) \cup border$
end while
return $SG(e_{ij})$

Figure 5 gives an example of how a 7-enclosing subgraph is coded by the optimized WL algorithm. The target link consists of two red nodes. The nodes in (1) are initialized with Equation 5. Step 1 and step 2 are the same as the original WL algorithm. In each iteration, Step1 contains the operation that each vertex constitutes a signature string by connecting its own coding and its neighbor's sorted coding. And Step 2 represents the operation that the coding of the vertex is updated according to the string order ascending in lexicographic

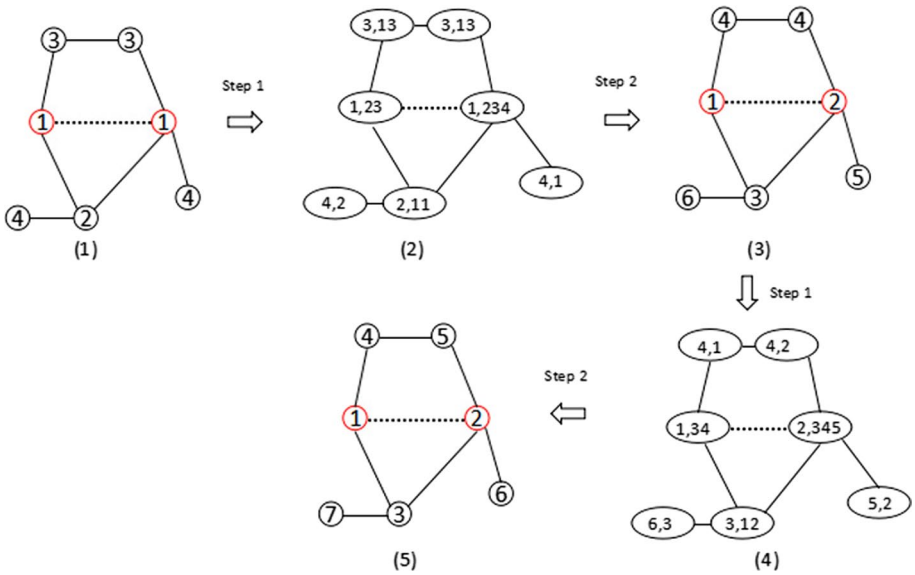

Fig. 5 Node coding with the optimized WL algorithm on an enclosing subgraph

order. We iteratively perform Steps 1 and 2 until the node's coding is converged, i.e., the node's coding stops changing. The proposed optimized WL algorithm guarantees that the two end nodes of the target link always have the smallest coding. The farther away from the target link, the higher the rank of a node; the closer to the target link, the lower the rank of a node. The target link is coded as $a_{1,2}$ in the adjacency matrix of the enclosing subgraph. This makes it easy for the machine learning model to ignore the label of the target link when reading the adjacency matrix. Otherwise, the trained model is meaningless.

4.3 Enclosing subgraph weighting

A weight is assigned to each link of the enclosing subgraphs to represent the relationship between this link and the target link. Let e_{xy} be a link of a k-hop enclosing subgraph $SG(e_{ij}^k)$, and e_{ij} is the target link, $e_{xy} \neq e_{ij}$. The weight of e_{xy} is assigned based on its distance to e_{ij}:

$$w_{xy} = dis(e_{xy}, e_{ij}) = S_{xi} + S_{xj} + S_{yi} + S_{yj}, \tag{6}$$

where $dis(\cdot,\cdot)$ is the distance of involved two links, and S_{xi}, S_{xj}, S_{yi}, S_{yj} are the Katz index of (v_x, v_i), (v_x, v_j), (v_y, v_i), (v_y, v_j), respectively. The Katz index reflects the integrated weight of all paths between two nodes. The Katz index of two nodes (v_a, v_b) is calculated as:

$$\begin{aligned} S_{ab} &= Katz(v_a, v_b) = \sum_{p=1}^{\infty} \alpha^p \left| paths^{<p>}(v_a, v_b) \right| \\ &= \alpha A_{ab} + \alpha^2 (A^2)_{ab} + \alpha^3 (A^3)_{ab} + \cdots, \end{aligned} \tag{7}$$

where A is the adjacency matrix of the network, $\left| paths^{<p>}(v_a, v_b) \right|$ counts the number of length p paths between v_a and v_b, α is the decay parameter which determines the path weight's decay ratio with the growth of the path length, and $(A^p)_{ab}$ is the number of paths having a path length p between v_a and v_b. By integrating different paths between v_a and

v_b, the Katz index assigns a higher weight to a shorter path. When formula (7) converges, the value of the variable parameter α is less than the reciprocal of the largest eigenvalue of the adjacency matrix [15], and the Katz index can be expressed as:

$$S = (I - \alpha A)^{-1} - A, \tag{8}$$

where I is a unit matrix. By assigning a weight to each link of $SG(e_{ij}^k)$, a weighted enclosing subgraph with k nodes is generated for e_{ij}. We name this weighted enclosing subgraph with k nodes as $SG^w(e_{ij}^k)$. A weighted adjacency matrix $SA^w(e_{ij}^k)$ is then generated to represent $SG^w(e_{ij}^k)$ for link prediction. Since the symmetric matrix is constructed for the undirected graph, we only use the upper triangular matrix.

4.4 Classifier training

Let f be a classifier and use a training set $D = \{(SA^w(e_1^k), y_1), (SA^w(e_2^k), y_2), \cdots, (SA^w(e_n^k), y_n)\}$ to train f to get a prediction model of the WESLP, in which $SA^w(e_i^k)$ is the weighted adjacency matrix of the weighted enclosing subgraph $SG^w(e_i^k)$, $i = 1, 2, \cdots, n$, e_i represent the i^{th} link of the training set D, and y_i is the label of e_i in D. $y_i \in \{0, 1\}$, in which 1 means e_i exists in the network, while 0 means e_i does not exist in the network. For a target link e_{ij}, its label is predicted as:

$$y(e_{ij}) = f(SA^w(e_{ij}^k)), \tag{9}$$

where $y(e_{ij}) \in \{0, 1\}$, and f is the link classifier trained by D (Fig. 6).

5 Results and Discussion

Experiments are held on several real-world networks to measure the performances of the proposed method. These real-world networks include: (1) Air traffic network [16] (ATN): nodes represent airports or service centers, and links represent routes between airports or service centers; (2) Power network [17] (PWN): nodes represent generators, transformators, or substations of the power grid, and links represent power supply lines between nodes; (3) C.ele network [17] (CEN): nodes are metabolites (e.g., proteins) of the roundworm caenorhabditis elegans, and links are interactions between nodes; (4) Router network [18] (RTN): nodes represent routers of the Internet connected autonomous system, and links represent the communications between routers; (5) Political blog network [19] (PBN): nodes represent US political blogs, and links represent hyperlinks between blogs; and (6) Network science paper network [20] (NSN): nodes represent researchers who publish papers on network science, and links represent collaborations between these researchers. The detailed information of these experimental networks is given in Table 1.

The performances of the proposed WLNM model are compared with some of the most popular link prediction methods that have been introduced in Sect. 2. These methods measure the neighborhoods of links for link prediction. The neighborhood measured in these methods is the subset of the subgraph used in the proposed methods. These methods include Preferential Attachment (PA), Common neighbors (CN), Adamic–Adar (AA), and Resource Allocation (RA). The classifiers used for link prediction are the same for all the above-mentioned methods, and the key differences are

Table 1 The detailed information of the experimental networks

	ATN	PWN	CEN	RTN	PBN	NSN
Number of nodes	1226	4941	297	5022	1222	379
Number of edges	2408	6594	2148	6258	16714	914

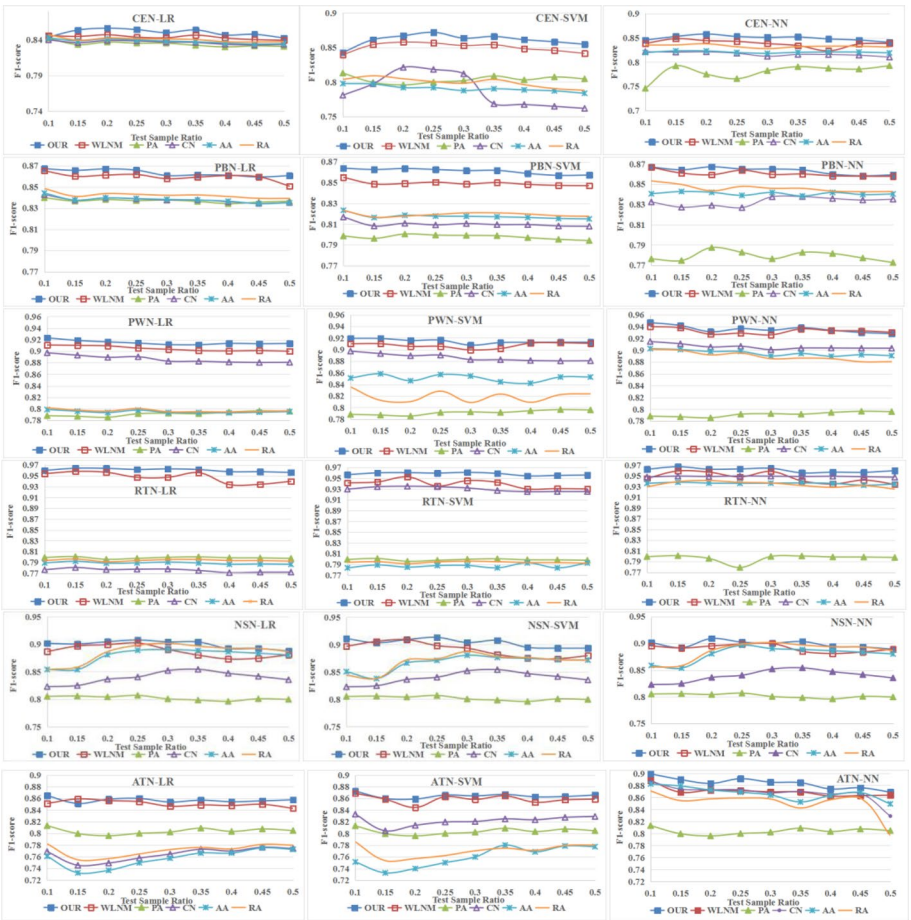

Fig. 6 Link prediction performances with logistic regression (LR), support vector machines (SVM), and neural network (NN) on the experimental datasets

how they measure the proximity between nodes. F1-score is used to evaluate the link prediction performances of these methods.

To compare the performances of the proposed method and the baseline methods, three of the most popular classifiers, including logistic regression (LR), support vector machines (SVM), and neural network (NN), are used to classify links according to their characteristics measured by subgraphs or neighbors. The WLNM algorithm is limited to the neural network, that is, the link prediction performance on the NN performs better, while the performance on the LR does not perform well. In order to highlight the robustness of our proposed method, in addition to use NN as classifiers, we also add SVM and LR in the experiments. In the experiments, LR is implemented

by LibLinear and L1-regularization, and SVM uses linear kernel and optimal parameters. NN uses three fully connected hidden layers with 50, 50, and 16 hidden neurons, respectively, and uses a softmax layer as the output layer. Logistic is adopted as the activation function for all hidden layers of NN. Adam is used to update the optimization rules with a learning rate of 0.001. The size of the mini-batch is set to be 128. The number of training epochs is set to be 100.

First, the performances of the proposed method are measured with the variety of test sample ratio. The size of the subgraph is set to be 10, i.e., $k = 10$, and the decay ratio α of the Katz index calculation is set to be the value that achieves the optimal solution. The experimental results are given in Fig. 7. It is shown that the proposed method has better link prediction performances on all experimental datasets with LR, SVM, and NN, respectively. Varying the test sample ratio in different experimental datasets, the proposed method still has better link prediction performances compared with the related works. Specifically, subgraph preserving link prediction models, including the proposed method and WLNM, have better link prediction performances compared with the popular neighborhood preserving link prediction models including PA, CN, AA, and RA. By measuring the weights of links in the subgraph, the proposed method has better link performances than WLNM, which predicts links with subgraphs without weights of links.

Second, the parameter sensitivity is further analyzed for the proposed model. Specifically, it is verified how the decay parameter α affects the link prediction performances. For the converge of the Katz index, α is set to be less than the reciprocal of the adjacency matrix. We take the value of α as m ($m = 0.9, 0.7, 0.5, 0.3, 0.1$) times the reciprocal of the adjacency matrix. The experimental results on the experimental datasets with LR are given in Fig. 7. It is shown that the link prediction performances of the proposed method tend to be best with a small value of m: The link prediction performances are the best with m equal to 0.1 on the PBN dataset; the link prediction performances are the best with m equal to 0.3 on the ATN dataset; the link prediction performances are the best with m equal to 0.1 or 0.3 on the PWN dataset; the link prediction performances are the best with m equal to 0.3 or 0.5 on the RTN dataset; and the link prediction performances are the best with m equal to 0.5 on the CEN and NSN datasets.

From Fig. 7, we can observe the following conclusions. WESLP generally performs much better than other baselines in terms of F1-score. The proposed method can

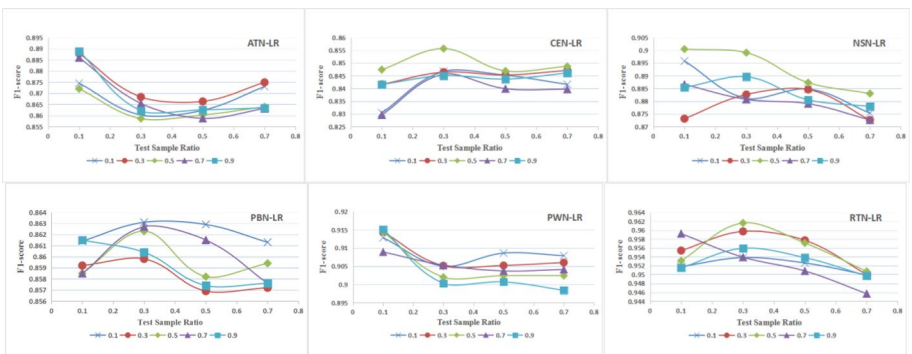

Fig. 7 Link prediction performances with LR on the experimental datasets with different m value

obtain good prediction results on different datasets under different classifiers and shows the robustness of the method. It is worth noting that many of the baselines often have poor performance, but our method has also achieved good results in the case of two sparse datasets PTN and RTN. The result show that our method can learn topology information that other baselines cannot express. Through the experimental analysis of the parameter decay ratio α in the method, the results show that different datasets have different alpha values, which makes WESLP achieve the best link prediction performance.

6 Conclusions and future works

In this paper, we proposed a weighted enclosing subgraph-based link prediction (WESLP) method, which utilizes topological features in the network by extracting links' k-enclosing subgraphs, encodes the extracted subgraph by the optimized WL algorithm, and adds weights to links in the enclosing subgraphs of the target link. We use the Katz index between nodes to make the link closer to target link, the greater the weight. WESLP makes full use of topology information around the target link. To assess the performance of our proposed method, we performed extensive experiments of link prediction on six real-world networks through three different classifiers. Comparisons between our method and four conventional methods suggest that our method performs better than AA, CN, RA, and PA on all tested network by different classifiers. Comparisons our method and WLNM validate that, in most real-world networks, our method has overall better F1-score than WLNM by different classifiers. The experimental results show that compared with the five baselines, our method is very robust; that is, the performance of different classifiers on different networks is better. In the future work, we will further evaluate our proposed method on larger datasets and do more work on the representation of the extracted subgraph.

Abbreviations

A	An adjacency matrix of graph or network
e_{ij}	The link or target link between v_i and v_j
a_{ij}	The value of the ith row and the jith column of the matrix A
w_{ij}	The weight of e_{ij}
$N_d(v_i)$	A set of vertices whose distance to v_i is no longer than d, where d is an integer
$d(v_i, v_j)$	The shortest path between v_i and v_j
$c(v)$	The coding of v
$SG(e_{ij})$	The enclosing subgraph of a target link e_{ij}
$SG(e_{ij}^k)$	The k-enclosing subgraph of a target link e_{ij}
SA	An adjacency matrix of subgraph
$IQ(\cdot,\cdot)$	The integer quotient of the involved two numbers
$R(\cdot,\cdot)$	The integer remainder of the involved two numbers
$dis(\cdot,\cdot)$	The distance of involved two links
S_{ab}	The Katz index between v_i and v_j
$\lvert paths^{<p>}(v_a, v_b)\rvert$	The number of length p paths between v_a and v_b
$(A^p)_{ab}$	The number of paths having a path length p between v_a and v_b
I	The unit matrix
$SG^w(e_{ij}^k)$	The weighted k-enclosing subgraph of a target link e_{ij}
e_i	The i^{th} link of the training set D

Acknowledgements

This research was supported by National Natural Science Foundation of China (Grant No. 61672284), Natural Science Foundation of Jiangsu Province (Grant No. BK20171418), China Postdoctoral Science Foundation (Grant No.

2016M591841), Jiangsu Planned Projects for Postdoctoral Research Funds (No. 1601225C). The authors extend their appreciation to the Deanship of Scientific Research at King Saud University for funding this work through research group no. RGP-VPP-264.

Author contributions
WY and YH proposed the initial idea. DG and GH refined the idea and designed the experiments. YT, AA, and MA conducted the experiments and analyzed the results. All authors read and approved the final manuscript.

Availability of data and materials
The datasets used and/or analyzed during the current study are available from the corresponding author on reasonable request.

Declarations

Competing interests
The authors declare that they have no competing interests.

Received: 7 May 2019 Accepted: 5 July 2022
Published online: 23 July 2022

References

1. David Liben-Nowell, Jon Kleinberg, The link-prediction problem for social networks. J. Am. Soc. Inf. Sci. Technol. **58**(7), 1019–1031 (2007)
2. Lada A. Adamic, Eytan Adar, Friends and neighbors on the web. Soc. Netw. **25**(3), 211–230 (2003)
3. A. Menon, C. Elkan. Link prediction via matrix factorization. In *ECML/ PKDD*, (2011)
4. Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. Learning entity and relation embeddings for knowledge graph completion. In Proceedings of AAAI, (2015)
5. L. Lü, T. Zhou, Link prediction in complex networks: A survey. Phys. A Stat. Mech. Appl. **390**, 1150–1170 (2011)
6. Tao Zhou, Linyuan Lü, Yi-Cheng. Zhang, Predicting missing links via local information. Eur. Phys. J. B **71**(4), 623–630 (2009)
7. M.E. Newman, Clustering and preferential attachment in growing networks. Phys. Rev. E **64**, 025102 (2001)
8. Albert-László. Barabási, Réka. Albert, Emergence of scaling in random networks. Science **286**(5439), 509–512 (1999)
9. Muhan Zhang, Yixin Chen. Weisfeiler-lehman neural machine for link prediction. In Proceedings of the 23rd ACM SIGKDD International Conference on knowledge discovery and data mining, pages 575–583. ACM, (2017)
10. Mathias Niepert, Mohamed Ahmed, Konstantin Kutzkov. Learning convolutional neural networks for graphs. In *Proceedings of the 33rd annual inter- national conference on machine learning. ACM*, (2016)
11. Shervashidze, Nino, Schweitzer, Pascal, Van Leeuwen, Erik Jan, Mehlhorn, Kurt, and Borg- wardt, Karsten M. Weisfeiler-lehman graph kernels. *The Journal of Machine Learning Research*, 12:2539–2561, (2011)
12. M. Hasan, V. Chaoji, S. Salem, and M. Zaki. Link prediction using supervised learning. In *Proceedings of the Workshop on Link Discovery: Issues, approaches and applications*, (2005)
13. Leo Katz, A new status index derived from sociometric analysis. Psychometrika **18**(1), 39–43 (1953)
14. W. Yuan, K. He, D. Guan, L. Zhou, C. Li. Graph kernel based link prediction for signed social networks. Inf. Fusion461–10, (2019)
15. M. Ou, P. Cui, J. Pei, Z. Zhang, and W. Zhu. Asymmetric transitivity preserving graph em- bedding. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*. ACM, pp. 672–681, (2016)
16. S. Wilkinson, S. Dunn, S. Ma, The vulnerability of the European air traffic network to spatial hazards. Nat. Hazards **60**(3), 1027–36 (2012)
17. Duncan J. Watts, Steven H. Strogatz, Collective dynamics of small-world networks. Nature **393**(6684), 440–442 (1998)
18. Neil Spring, Ratul Mahajan, David Wetherall, Thomas Anderson, Measuring ISP topo- logies with rocketfuel. IEEE/ ACM Trans. Netw. **12**(1), 2–16 (2004)
19. S. Meraz, Using time series analysis to measure intermedia agenda-setting influence in traditional media and political blog networks. Journal. Mass Commun. Q. **88**(1), 176–194 (2011)
20. Mark EJ. Newman, Finding community structure in networks using the eigenvectors of matrices. Phys. Rev. E **74**(3), 036104 (2006)

Publisher's Note