International Journal of Machine Learning and Cybernetics (2021) 12:2443–2458
https://doi.org/10.1007/s13042-021-01312-w

ORIGINAL ARTICLE

Path-based estimation for link prediction

Guoshuai Ma[1,2] · Hongren Yan[1,2] · Yuhua Qian[1,2,3] · Lingfeng Wang[4] · Chuangyin Dang[5] · Zhongying Zhao[6]

Received: 23 September 2020 / Accepted: 18 March 2021 / Published online: 1 April 2021

Abstract
Link prediction has received a great deal of attention from researchers. Most of the existing researches are based on the network topology but ignore the importance of its preference; for aggregating multiple pieces of information, they normally sum up them directly. In this paper, a path-based probabilistic model is proposed to estimate the potential connectivity between any two nodes. It takes carefully the effective influence of nodes and the dependency among paths between two fixed nodes into account. Furthermore, we formulate the connectivity of two inner-community nodes and that of two inter-community nodes. The qualitative analysis shows that the links between inner-community nodes are more likely to be predicted by the proposed model. The performance is verified on both the multi-barbell network and Lesmis network. Considering the proposed model's practicability, we develop an algorithm that iterates over the adjacent matrix to simulate paths of different lengths, with the parameters automatically grid-searched. The results of the experiments show that the proposed model outperforms competitive methods.

Keywords Link prediction · Preferential attachment · Community structure

1 Introduction

Complex model has been widely used to represent complex systems, where nodes and edges represent the entities of the system and their connections, respectively. One of the important issues in analyzing such complex networks is link prediction, which studies how nodes potentially link to each other [26]. By means of link prediction, we may ultimately find out the reason and power of why links arise [31]; in practice, link prediction has been applied to personalized recommendation [50], community detection [7, 24, 55], web search [37], and so on.

Link prediction methods developed in the past decades are mostly topology-based and learning-based [45]. The representatives of topology-based methods are neighbor-based, path-based and random walk. Neighbor-based methods assume that neighboring information indirectly reflect user behavior or implicitly affect user's choices, they are simple

Guoshuai Ma and Hongren Yan contributed equally to this work.

✉ Yuhua Qian
jinchengqyh@126.com

Guoshuai Ma
maguoshuaixy@126.com

Hongren Yan
nochioce_zerg@yahoo.com

Lingfeng Wang
wang289@uwm.edu

Chuangyin Dang
mecdang@cityu.edu.hk

Zhongying Zhao
zzysuin@163.com

[1] Institute of Big Data Science and Industry, Shanxi University, Taiyuan 030006, Shanxi, China

[2] School of Computer and Information Technology, Shanxi University, Taiyuan 030006, Shanxi, China

[3] Key Laboratory of Computational Intelligence and Chinese Information Processing of Ministry of Education, Shanxi University, Taiyuan 030006, Shanxi, China

[4] Department of Electrical Engineering and Computer Science, University of Wisconsin-Milwaukee, Milwaukee, WI 53211, USA

[5] Department of Manufacture Engineering and Engineering Management, City University of Hong Kong, Kowloon Tong, Hong Kong

[6] The School of Computer Science and Engineering, Shandong University of Science and Technology, Qingdao 266590, China

but effective. However, This class of methods only capture the local topological information. In contrast, path-based or random walk methods are capable of taking the quasi-local information into account [53]; they are time consuming but accurate. Note that the random walk methods may be considered as the extension of the path-based. Other than the topology-based model, advanced machine learning techniques are applied in link prediction as well, to list a few, feature-based classification [30], matrix factorization [47], and deep learning [57]; it takes time to train these models.

In essence, path-based, random walk and GCN [11] depend on propagation: they aggregate the neighborhood features to represent that of the target nodes. When the information from multiple paths (or neighbors) is available, addition is the most regular aggregation mechanism, which leads to imprecise estimation. Taking a simple circuit diagram as an example. The circuit has a power source, one bulb on the bus, and three switches on three parallel lines. The probability of closing each switch is assumed to be 1/2, and if we want to switch on the bulb, one switch must be closed at least. So the probability of switching on the bulb is $1-(1-1/2)\times(1-1/2)\times(1-1/2)=7/8$ rather than $3/2$ $(1=1/2+1/2+1/2)$. The latter result, analogous to that of previous methods in calculating the connection probability in the case of multiple paths, is greater than 1 and cannot be a reasonable probability value, moreover the links predicted are generated by comparing all links.

In fact, this phenomenon is widespread in the world. For telecom networks, when a message is sent from the source to the target, there are many paths to choose, and only one path is valid for the task. The power network, when transmitting power to a terminal consumer, choose only one path to operate. Thus, we are motivated to explore a better method to avoid all of these deficiencies. Had these values been manipulated more carefully, the model would have achieved a higher accuracy. So we propose an estimation formula which pays attention to the dependency between paths, and study its preference in terms of the community structure. The paper also designs an algorithm to approximate the formula to boost the computational efficiency. Our main contributions are:

(1) Unlike the aggregation method in previous works, the probability of two nodes being never connected is used to estimate the connection probability of a pair nodes. And the final connection probability is a probability value, which range from 0 to 1. Two typical graph are used as examples to made a lucid explanation for the property of PEPS.
(2) Previous researches neglected to study the preference of the method, thus their studies may be more reasonable if they had considered this situation. In this paper, the connection probability of inner-community links and inter-community links are calculated respectively through theoretical analysis. In general, PEPS prefers to predict links of two nodes that belong to the same community; and it also leverages the effective influence of nodes to predict the probability of a node being connected to the same community. Moreover, PEPS provides an alternative method for the preference of prediction.
(3) A developed method Iterative PEPS (IPEPS) is proposed to reduce the computational burden, and the results of the experiments show that it outperforms the state-of-the-art methods.

The remainder of this paper is structured as follows: some related works are briefly reviewed in Sect. 2. In Sect. 3, we establish the estimation formula for connections between nodes and analyze the prediction preferences with respect to community structure. Then an algorithm for efficiently approximating the formula is given. Section 4 details the experiments, including the experimental results and discussions. At last, the conclusion is drawn in Sect. 5.

2 Related works

In this section, a briefly review of topological-based and learning-based link prediction methods are present.

2.1 Topological-based methods

The most fundamental methods of topology based methods are neighbor-based, which quantify the similarities of nodes based on their common neighbors, such as common neighbors (CN) [25], adamic-adar (AA) [1], resource allocation (RA) [51], preferential attachment (PA) [4]; but they are only quantifying the number of common neighbors. Liu et al. extended the RA index by considering all the resources being transferred through neighbors and the amount of resources transferred by different neighbors being different. With the important observation that nodes preferentially link to other nodes with weak clique structure, Ma et al. [28] proposed the local friend recommendation (FR) that predicts the missing links better. Guo et al. [14] took the clustering coefficient of neighbor nodes into account in similarity estimation. The indices mentioned above are simple and effective.

Some models focus on path similarity, such as the local path index [27] and the Katz index [10], which measure the path similarity by the number of paths connecting them. A similarity index [2] calculate the local

information of the fixed distance between any pair of nodes, and it combines the advantages of neighbor-based methods and path-based methods. However, they have to tune one or more parameters for weighing paths with different lengths and neglect the heterogeneity in paths [58]. To address the above problem, Zhu et al. [58] proposed significant path (SP) to leverage the effective influence of endpoints and strong connectivity in calculating the similarity value, where they summed up the discounted degrees of all intermediate nodes over all local paths connecting two non-adjacent nodes. By assuming the huge influence of the hub nodes, Yang et al. [40] proposed a model to combined influence of endpoints and the transmission capability of quasi-local path, it achieves excellent performances. Inspired by the resource-traffic flow mechanism on networks, Yao et al. [53] hold that the more the intermediate nodes of a path receive resources from nodes on short paths, the more contribution of this path is, a path-dependent link predictor based on the Resource receiving process from Short Paths (RSP) is proposed. To overcome the difficulty of finding all paths between two nodes, various efficient methods with shortest paths or top-k shortest paths as the similarity measure is applied to predict potential links [20].

Random walk can capture global topological information with less computational resources. The basic idea of random walk [8] is that one particle starts at an arbitrary node in the network and randomly moves to its neighbors. Klein and Randic [19] hold that the particle may jump to the start node, so they proposed random walk with restart (RWR) method. Mahalanobis distance [9], a dissimilarity measurement between two vectors, was taken into account. Brin and Page [6] asserted that the particle may return to the starting point in web search, their proposed Page-rank based method achieve higher accuracy on link prediction. PropFlow [22] is similar to Rooted PageRank, but it is more localized due to fixed breadth-first search step and no restart mechanism. Jeh and Widom [18] claimed that the similarity of two nodes can be interpreted as their neighbors' similarity. Liu et al. [23] considered the limited steps of random walk and proposed local random walk (LRW). Based on the LRW index, they integrated the results of LRW into the superposed random walk (SRW) index. the former index considers the process of the limited number of steps, while the latter emphasizes the importance of the nodes closest to the target node. Among the link prediction methods published, RWR is still one of the most accurate methods at the time of this writing.

2.2 Learning-based methods

The link prediction can be treated as a supervised classification problem [16], which can be solved by classical learning models such as support vector machines [17], K-nearest neighbors [29] and logistic regression [21]. Furthermore, as the nodal features are intractable, that is, the adjacency matrix is too sparse and the additional attributes of the nodes cannot speak for the network topology, researchers studied the network representation learning to find low-dimensional vector representation of nodes automatically. Wang et al. [43] and Ou et al. [34] predicted links from the learned representations of nodes in publicly available collaborative social networks. However, when Goyal and Ferrara [12] compared a collection of latest network embedding methods of link prediction, such as Node2vec [13], HOPE [34], and SDNE [43], they found that the performance of such types depends more on the datasets and the dimension of embedding vectors. Zhang [54] proposed SEAL based on graph neural networks, and it achieved unprecedentedly competitive performance. Motif as the basic network blocks, it is applied in capture the higher order structures [44]. However, network representation learning aims to preserve the structure and inherent properties of the networks [46], it performs not so well in link prediction. Moreover these methods require a lot of training time.

3 Probability estimation of path similarity

We consider unweighted undirected network $G(V, E)$, where V is the set of nodes in the graph G, and E is the set of links. Multiple links and self-connections are not allowed in G. Then, i, j is the node of G, k_i is the degree of node i, d_{ij} is the distance between node i and j, Q_{ij} is the set of all paths between i and j, and Q_{ij}^d is the set of paths in which length is d from i to j, C_i is the community in which node i is located, q_{ij} and q_{ij}^d are the elements of Q_{ij} and Q_{ij}^d respectively. l_{ij} and l_{ij}^d are the lengths of the path q_{ij} and q_{ij}^d respectively.

Inspired by the connectivity in parallel circuits, a new link prediction method—path-based estimation on path similarity (PEPS) is proposed and its iterative algorithm (IPEPS) to predict link existence in complex networks.

3.1 Path-based estimation on path similarity (PEPS)

Definition 1 On an undirected unweighted network $G(V, E)$, based on random walk, the probability of a particle starting from i and reaching j through the path [58] q_{ij} ($q_{ij} = \{v_0 = i, v_1, \ldots, v_{l-1}, v_l = j\}$) is:

$$p_{ij}^{q_{ij}} = \prod_{m=1}^{l_{ij}-1} p(v_{m+1}|v_m) = \prod_{m=1}^{l_{ij}-1} \frac{1}{k_{v_m}}, \tag{1}$$

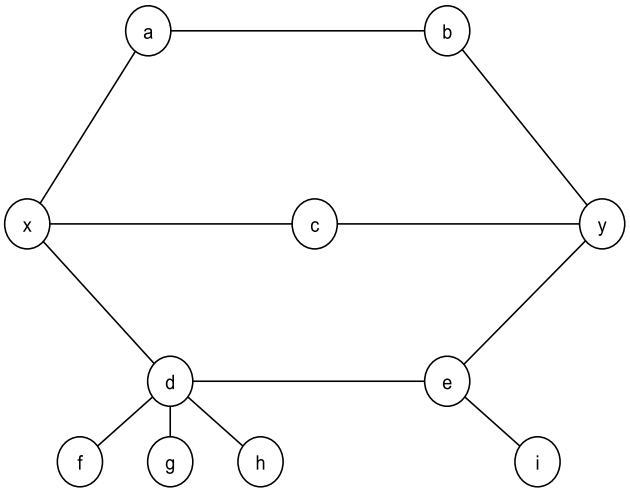


Fig. 1 An example network (colour figure online)

where $p(v_{m+1}|v_m) = \frac{1}{k_{v_m}}$ is the transfer probability from v_m to v_{m+1}, and k_{v_m} is the degree of v_m.

Then, we consider all paths between i and j and assume s is the longest path between i and j. The probability of a particle failing to travel from i to j is:

$$\overline{p_{ij}^{Q_{ij}}} = \prod_{d=1}^{s}\left(\prod_{q_{ij}^d \in Q_{ij}^d}\left(1 - \prod_{m=1}^{l_{ij}^d - 1}\frac{1}{k_{v_m}}\right)\right). \tag{2}$$

It is easy to see that the connection probability of i and j is $1 - \overline{p_{ij}^{Q_{ij}}}$.

Definition 2 On an undirected unweighted network $G(V, E)$, if all paths are counted, the connection probability of i and j is

$$p_{ij} = 1 - \prod_{d=1}^{s}\left(\prod_{q_{ij}^d \in Q_{ij}^d}\left(1 - \prod_{m=1}^{l_{ij}^d - 1}\frac{1}{k_{v_m}}\right)\right). \tag{3}$$

For example, in Fig. 1, there are three paths between x and y, $x-a-b-y$, $x-c-y$, and $x-d-e-y$. So the probabilities of particle moving from x to y through these paths are $1/2 \times 1/2 = 1/4$, $1/2$ and $1/5 \times 1/3 = 1/15$ respectively. Finally, the connection probability of x and y is $1 - (1-1/4) \times (1-1/2) \times (1-1/15) = 39/60$. However, in other link prediction indices, these probabilities are added directly, so the final connection probability is $1/4 + 1/2 + 1/15 = 49/60$. For general networks, we can fix a threshold valued between 0 and 1 in our model. The links with probabilities higher than the threshold are predicted to appear in the future. Furthermore, in this way, the effective influence [58] of nodes can be captured. And the pseudocode of PEPS is described in Algorithm 1.

Algorithm 1: PEPS

INPUT: Network $G(N, E)$, two nodes i and j.
OUTPUT: The connection probability of i and j p_{ij}.

1: Calculate the degree of each node.
2: Find all paths between i and j.
3: Calculate the connection probability of i and j by formula 3.
4: **return** p_{ij}

The index of PEPS is not only a probability value, but also can measure the influence of both the numbers and lengths of paths for each pair of nodes. Here, two special networks are used as examples to explain this characteristic. Firstly, in the path graph, except for the head and tail nodes, the degree of other nodes is 2, and with the increase of path length, the values of head and tail nodes should be smaller.

Property 1 *Let q_{ij} be a Path-graph and its length is l, and i and j are the head node and tail node respectively. Then, the connection probability of i and j is*:

$$p = 1 - \left(1 - \left(\frac{1}{2}\right)^{(l-1)}\right) = \frac{1}{2}^{(l-1)}. \tag{4}$$

This definition of connection probability has the following property:

$$\begin{cases} p \to 0 \;\; l \to \infty, \\ p = 1 \;\; l = 1. \end{cases}$$

So our model is able to reflect the decrease of connection probability as the path length increases. And in this case, if α is 1/2, PEPS yields same similarity between i and j as that measured by the Katz index (formulated as $S_{xy} = \alpha^{l-1}$).

As the number of nodes increases, the number of paths of each pair of nodes in the complete graph (a community) grows, so the connection probability of each pair of nodes should increase. And in a community, if all nodes are connected to each other, the more nodes this community contains, the higher the probability of connections between nodes will be.

Property 2 *In a complete graph with n nodes, the connection probability of each pair nodes i, j can be written as*

$$p_{ij} = 1 - \prod_{m=2}^{n-1}\left(1 - \frac{1}{(n-1)^{m-1}}\right)^{A_{n-2}^{m-1}}, \tag{5}$$

where A_{n-2}^{m-1} is the permutations, and n is the number of nodes in G ranging from 3.

Theorem 1 *Let $G(N, E)$ be a complete graph, and n is the number of nodes. If $n \to \infty$, then $p_{ij} \to 1$.*

Proof If we want to prove that the limit of Eq. (5) is 1, it is equivalent to verifying that the limit of the latter part of Eq. (5) is 0, which can be written as:

$$\begin{aligned} D &= \prod_{m=2}^{n-1}\left(1-\frac{1}{(n-1)^{m-1}}\right)^{A_{n-2}^{m-1}} \\ &= \prod_{m=1}^{n-2}\left(1-\frac{1}{(n-1)^{m}}\right)^{(n-1)^m\cdot\frac{A_{n-2}^{m}}{(n-1)^m}}. \end{aligned} \tag{6}$$

When $\varepsilon = \frac{1}{2e}$, there exists N_1, such that $n > N_1$, $|\left(1-\frac{1}{(n-1)^i}\right)^{(n-1)^i} - \frac{1}{e}| < \varepsilon$,

$$i.e., \frac{1}{2e} < \left(1-\frac{1}{(n-1)^i}\right)^{(n-1)^i} < \frac{3}{2e}, \tag{7}$$

so,

$$\left(\frac{1}{2e}\right)^{\sum_{i=1}^{n-2}\frac{A_{n-2}^i}{(n-1)^i}} < D < \left(\frac{3}{2e}\right)^{\sum_{i=1}^{n-2}\frac{A_{n-2}^i}{(n-1)^i}}. \tag{8}$$

Using Stirling formula, we show $D \to 0$, if $n \to \infty$. Setting $a = \frac{1}{2}$, there exists i such that $\frac{A_n^i}{n^i} > a = \frac{1}{2}$. According to the trial, when $i = \frac{n}{2}$, $\frac{A_n^i}{n^i} \sim (\frac{1}{e})^{\sqrt{n}}$ is calculated, so we guess i is much smaller than n—hence we assume $i \in o(n)$, then

$$\begin{aligned} &\frac{A_n^{i+1}}{n^{i+1}} = \frac{A_{n-1}^i}{n^i}, \\ &A_n^i = \frac{n!}{(n-i)!} \sim \frac{n^n e^{-n}\sqrt{2\pi n}}{(n-i)^{n-i}e^{-(n-i)}\sqrt{2\pi(n-i)}}, \\ &\frac{A_n^i}{n^i} \sim \left(1+\frac{i}{n-i}\right)^{n-i+\frac{1}{2}}\cdot e^{-i}. \end{aligned} \tag{9}$$

Since $\frac{A_n^i}{n^i}$ is a value decreasing as m increases,

$$\begin{aligned} ln\left(\frac{A_n^i}{n^i}\right) &= ln\left(1+\frac{i}{n-i}\right)\cdot\left(n-i+\frac{1}{2}\right) - i \\ &\sim \left(\frac{i}{n-i} - \frac{i^2}{(n-i)^2}\right)\cdot\left(n-i+\frac{1}{2}\right) - i \\ &\sim \frac{1}{2}, \end{aligned} \tag{10}$$

in the approximation, $ln(1+x) \sim x$ is applied in Eq. (10), so

$$\frac{i^2}{n} \sim \ln 2, i \sim \sqrt{n\cdot ln2}. \tag{11}$$

The approximation in Eq. (6) is of the following form:

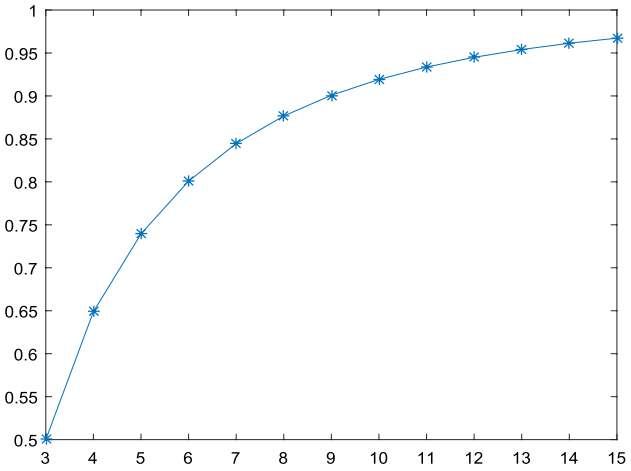


Fig. 2 The connection probability of K_n complete graph

$$\ln(1+x) \sim x - \frac{x^2}{2}. \tag{12}$$

So, $\sum_{i=2}^{n}\frac{A_n^i}{(n)^i} \geq \frac{\sqrt{(\ln 2)n}}{2}$, which converges to infinity when n tends to be infinity. Therefore, as $n \to \infty$, $D \to 0$, $p_{ij} \to 1$. □

Then we check the connection probability of any two nodes in the above complete graphs with the number of nodes ranging from 3 to 15 respectively, and the connection probability is shown in Fig. 2. In the 15-node complete graph, the connection probability is 0.9672, which is very close to 1.

3.2 The link prediction of PEPS

In this section, we investigate the relationship between link prediction preference of the proposed model and community [39], with the assumption [52, 56] that a community is a subgraph C, E_C of G in which each node is more densely linked to others than to the nodes in $N\backslash C, E\backslash E_C$. For the simplicity of the following analysis, we suppose that the community is approximately a complete subgraph of G. To check the validity of these two types of connections (properties 1&2), we assume node a in community C_1, with probability P of the connection to any other in the same community; and node B in another community C_2, with probability P'. If there is only one path l bridging C_1 and C_2, it has probability P_1 based on property 1, and the link always prefers inner-community (i.e., $P > PP_1P'$). If there is more than one path of property 1, we take two non-crossing paths connecting the two communities into account at first. The probability P_{Int} of A to $c_2 \in C_2$ is given by

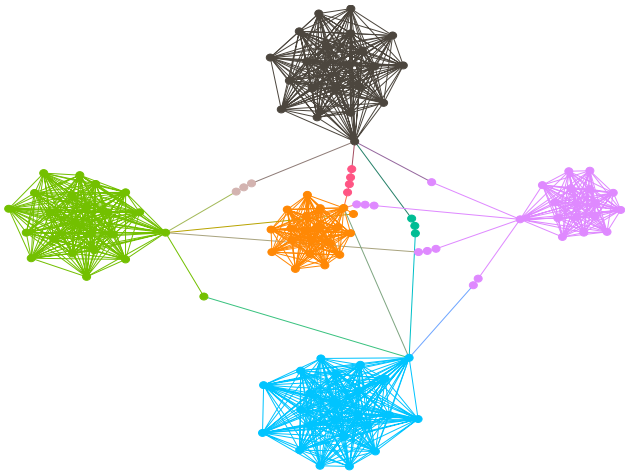


Fig. 3 A multi-barbell network contains five complete graphs, and each complete graph is connected to other complete subgraphs by a path graph, the length of the path graph and the nodes of each complete graph are random from 3 to 10

$$P_{Inter} = 1 - (1 - PP_2P')(1 - PP_1P'). \tag{13}$$

And P_{Inner} of A to $c_1 \in C_1$ by $P_{Inner} = 1 - (1 - P)(1 - PP_1P_2P')$, then

$$\begin{aligned}\Delta &= P_{Inner} - P_{Inter} \\ &= P(1 - P_2P')(1 - P_1P') > 0.\end{aligned} \tag{14}$$

The generalized formula for the k paths situation is complicated, so that we had better consider general k's paths (allowing for crossing each other) under property 2, and $P_{Inner} = 1 - (1 - P)(1 - PP_lP')$, while $P_{Inter} = PP_lP'$. In this case, $\Delta > 0$, for any P_l which has 1 as its peak when the graph of $G - E(C_1 \cup C_2)$ is a complete graph. For most real networks that contain community structure, P_l is much smaller than 1, meaning that the same community may have much higher link preferences.

For example, as shown in Fig. 3,[1] the barbell graph can be seen as a combination of multiple barbell graphs, and each pair of the barbell subgraphs is connected by a path graph. When two nodes are the endpoints of the bridge between two different complete subgraphs respectively, it is obvious that the similarity is smaller than two nodes in a complete subgraph. As long as the similarity of two endpoints on the bridge edge is less than 1, the similarity of two nodes in two different complete subgraphs is less than that of any two nodes in one complete subgraph. In this sense, we say the connection probability in one complete subgraph is stronger than that of two nodes from two different complete subgraphs. Our definition relies on a complete graph structure, so if there are communities that are close to a complete shape, our prediction will predict well, no matter how small the size of the community is.

When n approaches infinity, $\frac{A_n^2}{n^i} \approx \frac{1}{e}\sqrt{n}$, so $\prod_{i=\frac{n}{2}+1}^{i=n} \frac{A_n^i}{n^i} \leq \frac{n}{2}e\sqrt{n}$, and this indicates that in specific cases, complete connection may be relaxed to a sparser structure. For instance, we can ask for a community to be a bunch of nodes that connect to each other within a link order smaller than the average on the whole network.

3.3 Iterative approximation to PEPS (IPEPS)

Though we only consider the second-order or third-order paths, it is too slow to find all paths in each network using Eq. (3). In order to reduce the time of computation [35], we adapt Eq. (3) to be an iterative algorithm. It is assumed that each node is not affected by its neighbors, and it is not assumed that each path between two nodes must be independent in the network. Let a, b, and c be three nodes where b is a neighbor of c, L_{ab} is the set of paths from a to b, m is the distance from a to b. Therefore $m + 1$ is the shortest distance between a and c. Then the probability that a particle starts from a and arrives at c through b is

$$p(a \to b \to c) = p_{a\to b} \times p_{b\to c} = p_{a\to b} \times \frac{1}{k_b}. \tag{15}$$

So, the probability of a not passing through b or not arriving at c is

$$\overline{p_{a\to b\to c}} = 1 - p_{a\to b\to c}. \tag{16}$$

Let $N = \{b | l_{ab} = m, l_{bc} = 1, l_{ac} = m + 1\}$, so the probability of a having at least one path end to c is

$$\begin{aligned}p_{ac} &= 1 - \prod_{v\in N}(1 - px(a \to v \to c)) \\ &= 1 - \prod_{v\in N}\left(1 - p_{a\to v} \times \frac{1}{k_v}\right).\end{aligned} \tag{17}$$

If p_{ab} is known, as well all nodes in N, the probability of a reaching c can be calculated. Then, we model PEPS into an iterative form (IPEPS) depicted as follows:

$$\begin{aligned}p_{ij}^{(1)} &= 1 - \prod_{u=1}^{n}\left(1 - m_{iu} \times z_{uj}\right), \\ &\cdots \\ p_{ij}^{(t+1)} &= 1 - \prod_{k=1}^{n}\left(1 - p_{iu}^{(t)} \times z_{uj}\right),\end{aligned} \tag{18}$$

[1] All pictures of networks in this paper are drawn by Gephi which is a software for the visualization of graphs and networks (http://networkrepository.com/index.php). Fruchterman Reingold [32] is used to generate the network layout, and the nodes in a network are colored by their modularities.

where M is an identity matrix representing the initial state of the network, and Z is the state transition matrix. The iteration simulates the path length that the particle traverses on the network. Inspired by RWR, the restart mechanism is introduced to Eq. (18) accordingly, and the IPEPS becomes

$$p_{ij}^{(t+1)} = c\left(1 - \prod_{k=1}^{n}(1 - p_{ik}^{(t)} \times Z_{kj})\right) + (1-c)e_i, \tag{19}$$

where c denotes the probability of the particle returning to the initial position, and e_i is the value of the initial state. Because the random particle may be wiggling between any two nodes, this method is not exactly equivalent to PEPS. The pseudocode of IPEPS is described in Algorithm 2. In terms of algorithmic complexity, the computational complexity of IPEPS is $O\left(K * N^3\right)$, with K being the number of iterations. It can be seen that this algorithm is not complex at all, even compared to other simple ones.

Algorithm 2: IPEPS

INPUT: The adjacency matrix A of network $G(N, E)$, iterative times S, restart probability c, transition matrix Z.
OUTPUT: The connection probability matrix of any pair nodes P.

$s \leftarrow 0$, $P \leftarrow A$.
while $s < S$ **do**
  for $i = 1; i < n; i++$ **do**
    for $j = 1; j < n; j++$ **do**
      for $k = 1; k < n; k++$ **do**
        $temp = temp * (1 - p_{ik} * z_{kj})$
      end for
      $P_{ij} = c * (1 - temp) + (1 - c)$
    end for
  end for
end while
return P

4 Experiments

4.1 Data description

The proposed models PEPS/IPEPS are tested in eight real networks, and these networks have been converted into unweighted undirected networks, with loops and multi-links eliminated on the premise of network connectivity. These networks[2] are listed as below, and the detailed features are listed in Table 1:

Table 1 The basic feature of datasets

Datasets	$\|V\|$	$\|E\|$	D	$\langle k\rangle$	$\langle d\rangle$	C
Adjnoun	112	425	0.068	7.589	2.536	0.283
Celegansneural	297	2345	0.053	15	2.455	0.311
Chesapeake	35	118	0.198	6	2.508	0.339
Usair	332	2126	0.039	12.807	2.738	0.749
Yeast	2375	11,693	0.004	9.847	5.096	0.388
Power	4941	6594	0.001	1.335	2.8	0.04
Openflight	2939	30,501	0.004	10.378	4.145	0.435
Euroroad	1174	1417	0.002	2.414	18.371	0.02

$|V|$ denotes the number of nodes, $|E|$ is the number of edges, D is the graph density, $\langle k\rangle$ denotes the average degree, $\langle d\rangle$ denotes the average distance, and C represents the clustering coefficient [38]

(1) Adjnoun ([36]): contains the network of common adjectives and noun adjacencies in novel David Copperfield by Charles Dickens, as described by M. Newman.
(2) Celegansneural ([49]): Neural network of the nematode *C. elegans* Compiled by Duncan Watts and Steven Strogatz from original experimental data.
(3) Chesapeake ([3]): is a network of carbon flows among species living in the Chesapeake Bay. The data are collected in three main areas: lower, middle, and upper bay.
(4) Power ([48]): a network representing the Western States Power Grid of the United States, in which nodes are transformers or power relay points and two nodes are connected if a power line links them.
(5) Usair ([5]): the network of the US air transportation system.
(6) Yeast ([33]): Interaction detection methods have led to the discovery of thousands of interactions between proteins, and discerning relevance within large-scale data sets is important to present-day biology.
(7) Euroroad ([42]): the international E-road network, a road network located mostly in Europe. The network is undirected; nodes represent cities and an edge between two nodes indicates that they are connected by an E-road.
(8) Openflight ([36]) is downloaded from Openflights.org, and it contains ties between two non-US-based airports (Table 1).

[2] These real networks can be downloaded at http://networkrepository.com/index.php.

In the experiments, to ensure the network is connected, the training set E^T is constructed from G by randomly removing 10% edges in it. Note that in this construction process the connectedness of the graph has to be preserved. Moreover, the removed edges are added to the test set E^P. The prediction accuracy is evaluated by 100 times with independent random network division of the training set and the test set.

4.2 Evaluation metrics

AUC [15] (area under the ROC curve) can be interpreted as the probability that a randomly chosen missing link (a link in E^P) is given a higher score than a randomly chosen non-existent link (a link in U/E , where U denotes the universal link set). In the algorithmic implementation, we usually calculate the score of each non-observed link, then at each time a missing link is randomly picked and compared with the nonexistent link based on their scores. Among N independent comparisons, if there are n' times that the missing link has a higher score, and n'' times that they have the same score, AUC can be calculated as follows:

$$AUC = \frac{n' + 0.5n''}{n}. \tag{20}$$

AUC estimates the accuracy of the index globally, with the significance that if all scores are generated from independent and identical distribution the accuracy should be about 0.5. Therefore, the degree to which the accuracy exceeds 0.5 indicates how much the algorithm outperforms the pure chance one.

4.3 Baseline

In order to illustrate the performance of our model, it is compared with the eight topological methods: CN, AA, RA, FR of local-similarity based; LP and Katz the local-path based; ACT, RWR random-walk based; and two network representation methods: node2vec and SEAL. And they are listed as below.

(1) `Common neighbors (CN)` ([25]) index measures if two endpoints are similar (the more common neighbors they have, the higher CN value will be). The CN index can be calculated as follows:

$$S_{xy} = |\Gamma(x) \bigcap \Gamma(y)|, \tag{21}$$

where $\Gamma(x)$ is the set of nodes which are neighbors of endpoint x, and $\Gamma(x) \bigcap \Gamma(y)$ denotes the set of common neighbors of node x and y.

(2) `Adamic-Adar (AA)` ([1]) index punishes the common neighbors with high degrees by considering the logarithm of reciprocal of common neighbors degrees:

$$S_{xy}^{AA} = \sum_{z \in (\Gamma(x) \bigcap \Gamma(y))} \frac{1}{\log k(z)}. \tag{22}$$

$k_{(z)}$ is the degree of node z.

(3) `Resource Allocation (RA)` ([51]) index, similar to AA, punishes the common neighbors with big degrees just by considering the reciprocal of common neighbors degree:

$$S_{xy}^{RA} = \sum_{z \in (\Gamma(x) \bigcap \Gamma(y))} \frac{1}{k(z)}. \tag{23}$$

(4) `Local Path (LP)` ([27]) index counts the contribution of local paths with 3,

$$S_{xy}^{LP} = A^2 + \alpha A^3, \tag{24}$$

(5) `Katz` ([10]) index considers all paths in the network, and it can be expressed as:

$$\begin{aligned} S_{xy} &= \sum_{i=1}^{L} \alpha^i |paths_{x,y}^{\langle i \rangle}| \\ &= \alpha A_{xy} + \alpha^2 \left(A^2\right)_{xy} + \cdots + \alpha^L \left(A^L\right)_{xy}, \end{aligned} \tag{25}$$

where L is a constant indicating the longest length considered in the Katz index, and $|path_{xy}^{\langle i \rangle}|$ represents a collection of paths that connect vertices v_x and v_y, A is the adjacency matrix, and the parameter $\alpha \in (0, 1)$ is used to control the weight coefficient of path.

(6) `Average commute time (ACT)` ([19]) counts the average steps that a random walk particle takes to move from A to B, which can be defined as:

$$S_{xy}^{ACT} = \frac{1}{l_{xx}^{+} + l_{yy}^{+} - 2l_{xy}^{+}}, \tag{26}$$

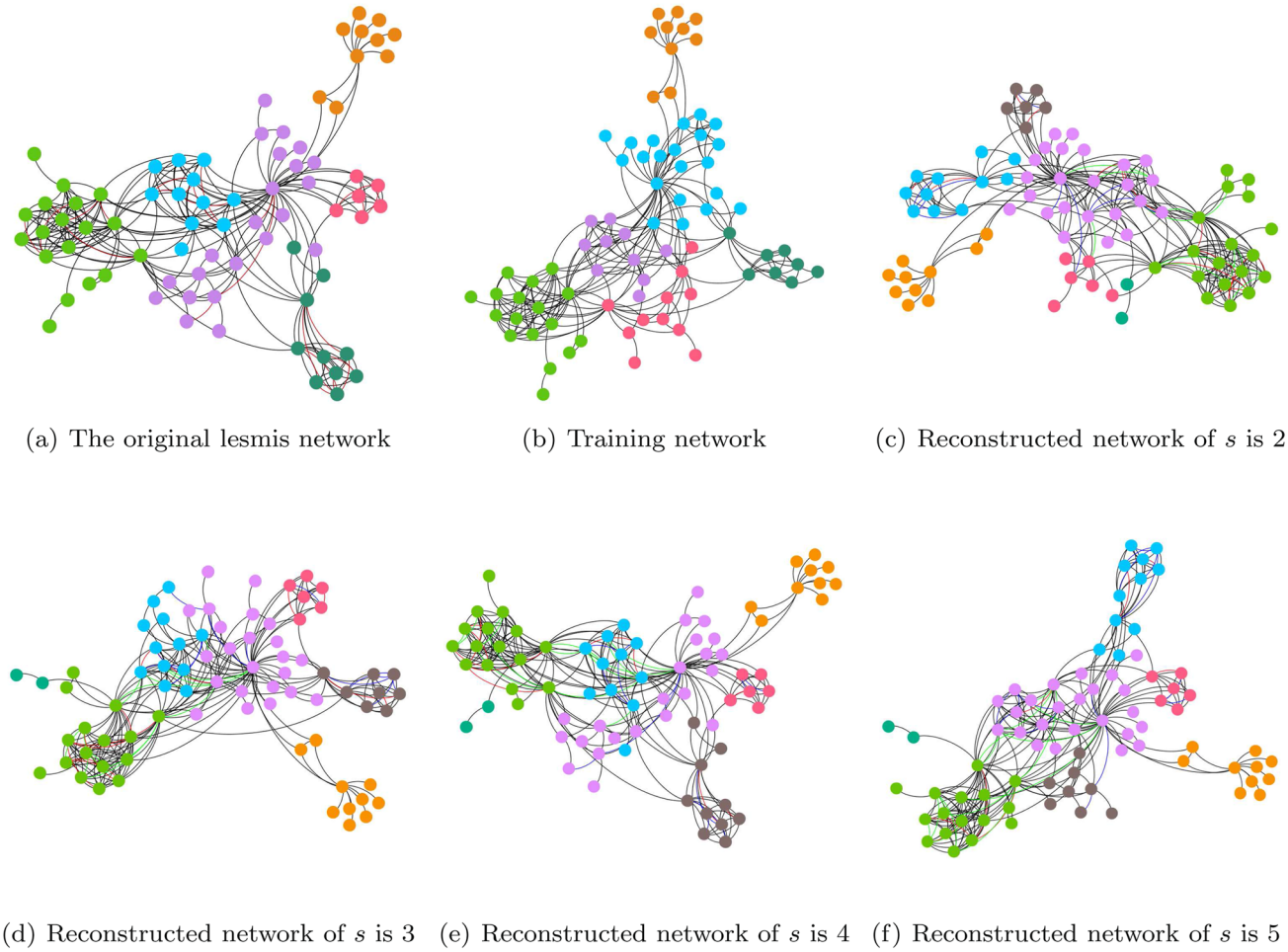
where l_{xx}^{+} is the element in L^{+}, which is the pseudo inverse of Laplacian matrix $L(L = D - A)$.

(7) `FR index` ([28]) is a method which based on friend recommendation model, and it can be formula as :

$$S_{xy}^{FR} = \sum_{l \in \Gamma(x) \bigcap \Gamma(y)} \frac{1}{k(l) - 1 - S_{jl}^{CN}}, \tag{27}$$

where S_{jl}^{CN} is the number of the common neighbors of i and j, and $k(l)$ denotes the degree of l.

(8) `Random walk with restart (RWR)` ([41]) considers a random walk particle may go back to the initial location during its walk in network. The probability vector of a particle reaching every node from the initial position in $t + 1$ is:



(a) The original lesmis network (b) Training network (c) Reconstructed network of s is 2

(d) Reconstructed network of s is 3 (e) Reconstructed network of s is 4 (f) Reconstructed network of s is 5

Fig. 4 The result of PEPS prediction in Lesmis, and the nodes with different color denote that they are not in a same community. **a** Is the original network of Lesmis; **b** is the training network with 10prediction results by PEPS with s ranging from 2 to 5 respectively. And the red links are intersection of our predictions and the test links, the green links exclusively in the prediction and the blue links exclusively in the test links

$$\pi_x(t+1) = c \cdot P^T \pi_x(t) + (1-c)e_x, \tag{28}$$

where e_x denotes the initial state, and P^T is the probability transfer matrix. And in the steady state,

$$\pi_x = (1-c)(I - cP^T)^{-1} e_x. \tag{29}$$

And the similarity of RWR is:

$$S_{xy}^{RWR} = \pi_{xy} + \pi_{yx}. \tag{30}$$

(9) `Node2vec` ([13]) is an algorithmic framework for learning continuous feature representations for the nodes and it is a typical representational learning method on graphs.

(10) `SEAL` ([54]): Based on GNN, it can use the node's feature vector to construct the node information matrix to predict links, and the information matrix has three components: structural node labels, node embeddings and node attributes.

4.4 Results and discussion

4.4.1 The preference of the prediction

The preference of link prediction by PEPS is illustrated by Lesmis. As shown in Fig. 4a, ten percent of the links are randomly selected as a test set which are denoted as red lines; while the training network (plotted in Fig. 4b) is obtained by removing these red lines. Now, PEPS predicts the links most likely to appear with s ranging from 2 to 5, in which the number of the predicted links exactly matches the number of links in the test network as shown

Table 2 The topological features of reconstructed network based on Lesmis by PEPS

Datasets	M	D	C	$\langle d \rangle$
`Original net`	0.546	5	0.736	2.641
`train net`	0.532	6	0.669	2.739
`net of` $s = 2$	0.532	6	0.79	2.658
`net of` $s = 3$	0.52	6	0.772	2.674
`net of` $s = 4$	0.527	6	0.774	2.673
`net of` $s = 5$	0.522	6	0.756	2.663

|M| denotes the modularity of network, |D| is the diameter of network, and C represents the clustering coefficient of network, $\langle d \rangle$ denotes the average distance

in Fig. 4c–f. As shown in Fig. 4c–f, the red links are the intersection of our predictions and the test links, the green ones are exclusively in the prediction links and the blue ones are exclusively in the test links. Some topological features of these networks are listed in Table 2. It can be observed that the modularities of reconstructed networks are lower than the original network, and the longer the network's diameter is, the shorter the average path is, and the larger the clustering coefficient larger will be. It can be found here that PEPS prefers to predict the links that belong to the same community. Based on the reconstructed network of s ranging from 2 to 5, it is discovered that with the increase of the length of paths, the average length of paths becomes longer, and the clustering coefficient decreases. This indicates that PEPS can successfully predict shortcuts that can guarantee an adequately long path.

To display the predicted preferences of PEPS more clearly, a multi-barbell graph is created, which is comprised of 10 complete subgraphs connected to each other by a path graph with random lengths. To increase computational efficiency, we use IPEPS as a substitute for PEPS on this graph. As shown in Fig. 5, the more the deleted edges are, the less visible the network's community structure is, thus the accuracy of IPEPS decreases. This implies that IPEPS is more likely to predict the links which locate within the community. When the proportion of deletions in the network is sufficiently high (over 80%), the longer the step is, the higher the accuracy is. If the community structure is not obvious, or the network is very sparse, the step can be increased to predict shortcuts. Therefore, no matter how sparse the network or how long the length of a link to be predicted is, PEPS and IPEPS will maintain good performance.

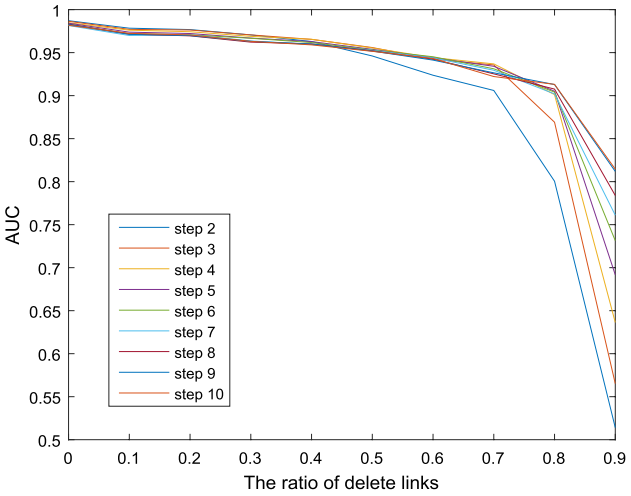


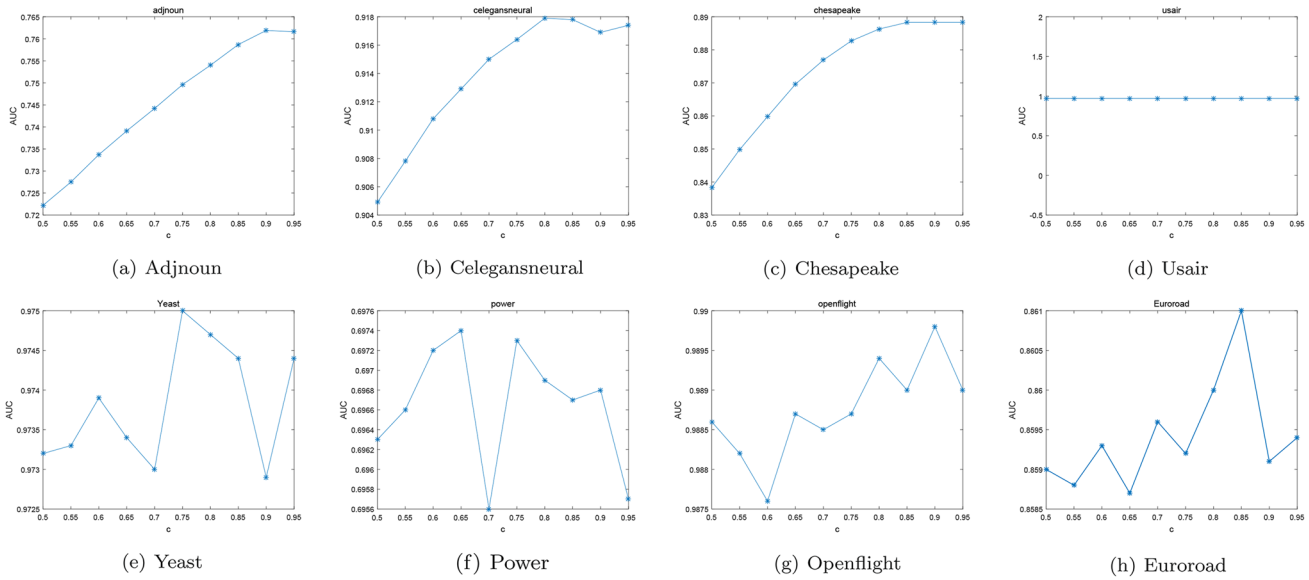
Fig. 5 The AUC of a multi-barbell network with different numbers of deleted edges, which is comprised of 10 complete subgraphs. The network structure changes through deletion operations—the more the deleted edges are, the sparser the network is

4.4.2 The analysis of parameter sensitivity

There are two parameters in this paper—restart probability c and steps s. c denotes the probability of the particle returning to the initial position. In other words, c is used to adjust the importance or quality of target nodes. Furthermore, s is the parameter to tune the weight of local topology information and global topology information, the small s means that the random walks tend to capture the local topology information and vice versa.

Then, the impact of the restart probability c is studied. In Fig. 6, c is increased from 0.50 to 0.95 with a step size of 0.05. It is found that as c increases, the accuracy in (a)–(c) tends to be higher; the accuracy is not affected in (d); the accuracy tends to decrease in (f); and in the last large network, the accuracy fluctuates in a certain range.

In small networks, the majority of missing links are short paths, and local links are sufficient for prediction. Therefore, the step on the four small real networks ranges from 2 to 8, and the performance is demonstrated



(a) Adjnoun (b) Celegansneural (c) Chesapeake (d) Usair

(e) Yeast (f) Power (g) Openflight (h) Euroroad

Fig. 6 The AUC of different c in all experimental networks

in Fig. 7. The reason for the fluctuation of accuracy is that the particle may wiggle between two nodes.

On the contrary, the links to be predicted in large networks are usually shortcuts. Since short paths are too short to be used to predict shortcuts, longer paths are needed. With the observations from Table 4, the global methods outperform the local ones. When the step is gradually increased to 40 in Fig. 8, it can be seen that the values of AUC tend to rise, despite fluctuation occurs. In Figs. 7 and 8, it is observed that the accuracy of the odd-numbered steps is higher than that of the even-numbered ones—this is because when the iteration number is even, the random particle may eventually return to the starting position.

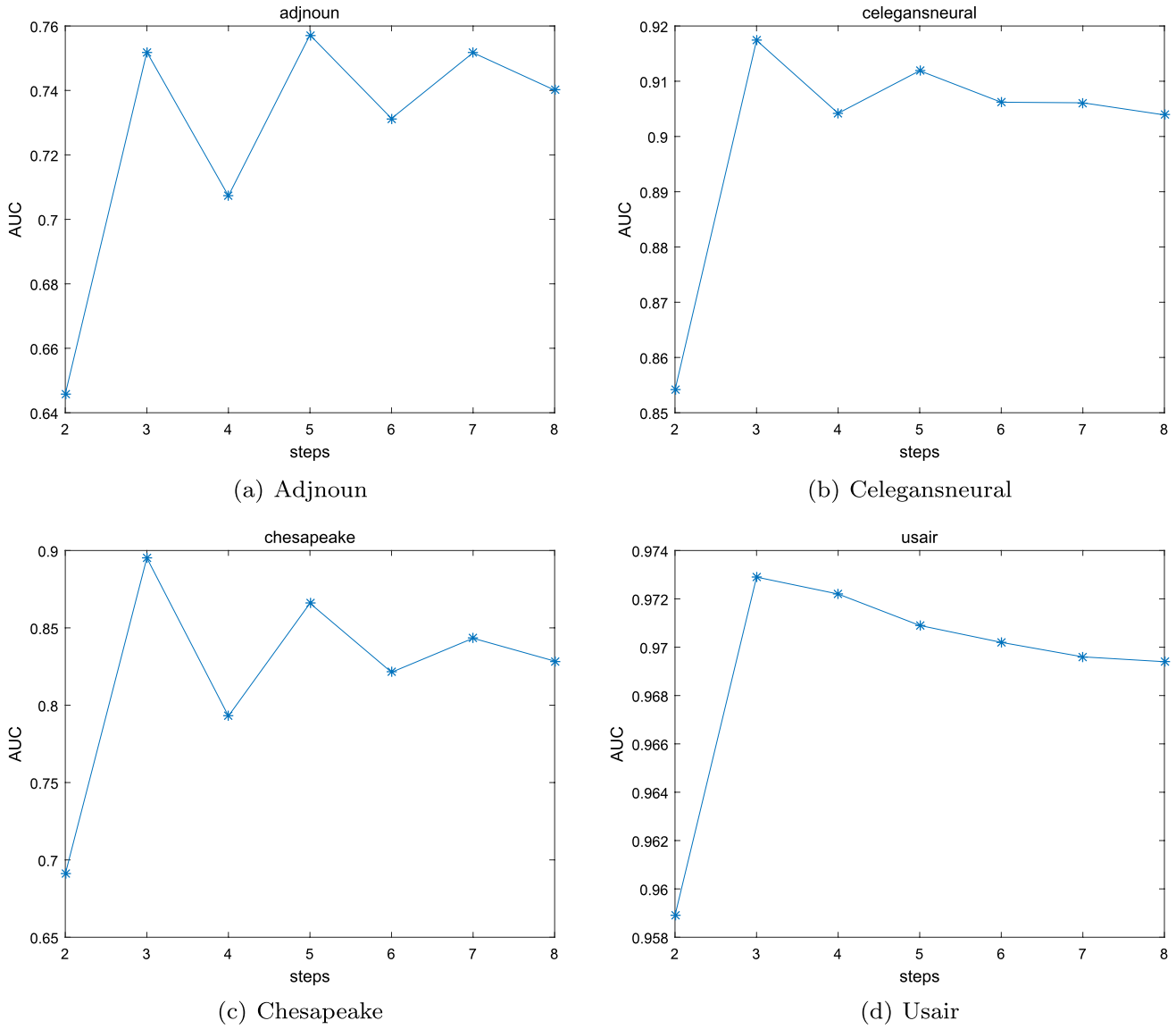
4.4.3 Performance

In order to examine the validity of the IPEPS, as shown in Table 3, IPEPS and PEPS are used in the experiments based on the eight networks with fixed path lengths of 2 and 3. From Table 3, there are two findings: (1) By using more topological information, both IPEPS and PEPS of order 3 are more precise than those of order 2; (2) the accuracy of IPEPS is similar to that of PEPS for most studied networks, and for certain networks, IPEPS is slightly more precise than PEPS.

Lastly, the grid parametric search is embedded in the IPEPS algorithm (GS-IPEPS). In the grid search process, a few links (E_T) are added to the development set, and the remaining links are kept in the training set. The parameters that yield the highest prediction accuracy in the development set are automatically selected. A comparison with other indices is shown in Table 4, and T test is utilized for method comparison. in Table 5. There are two hypotheses in the T test. H_0 is $\mu_1 - \mu_2 \leq 0$ ($\alpha \geq 0.05$), H_a is $\mu_1 - \mu_2 > 0$, where μ_1 denotes PEPS or IPEPS, μ_2 refers the other methods. Table 5 shows that H_0 should be rejected. Moreover, it can be seen that for most datasets from Table 4, IPEPS outperforms other indices on all networks but Euroroad and Openflight, on which the accuracy is still very close to the best ones. In addition, the average performance of IPEPS and GS-IPEPS are top-2 in all algorithms, and the average performance of both algorithms is better than that of SEAL in most data sets. On four small networks, IPEPS and GS-IPEPS perform excellently. Although IPEPS works subtly better than GS-IPEPS on some datasets, GS-IPEPS reduces the uncertainty of manual parametric selection and is considered a more stable approach.

5 Conclusion

We have proposed a new link prediction model—the path-based estimation on path similarity (PEPS). This model has three properties: with the increase of the number of nodes, the connection probability of each pair of nodes will approach 0 in the path graph and approach



(a) Adjnoun (b) Celegansneural

(c) Chesapeake (d) Usair

Fig. 7 The impact of different steps in four small networks

1 in the complete graph; moreover, our model prefers to predict links of two nodes that belong to the same community; and it also leverages the effective influence of nodes to predict the probability of a node being connected to the same community. In practice, since PEPS computes all paths of each pair of nodes inefficiently, the approximation algorithm (IPEPS) and its automatic-parametric-selection version termed GS-IPEPS are proposed as two efficient surrogates. Then the algorithms are tested in eight real networks and achieved good performance.

In addition, one of the potential applications of the proposed model is community detection. Although in Sect. 2.1, Property 2 is derived for complete graphs, the following experiments indicate that this property can be beneficial in accurately finding the expected links in real networks. These links found that PEPS or IPEPS tend to be connected nodes in the same "community" based on the analysis of prediction preferences. In other

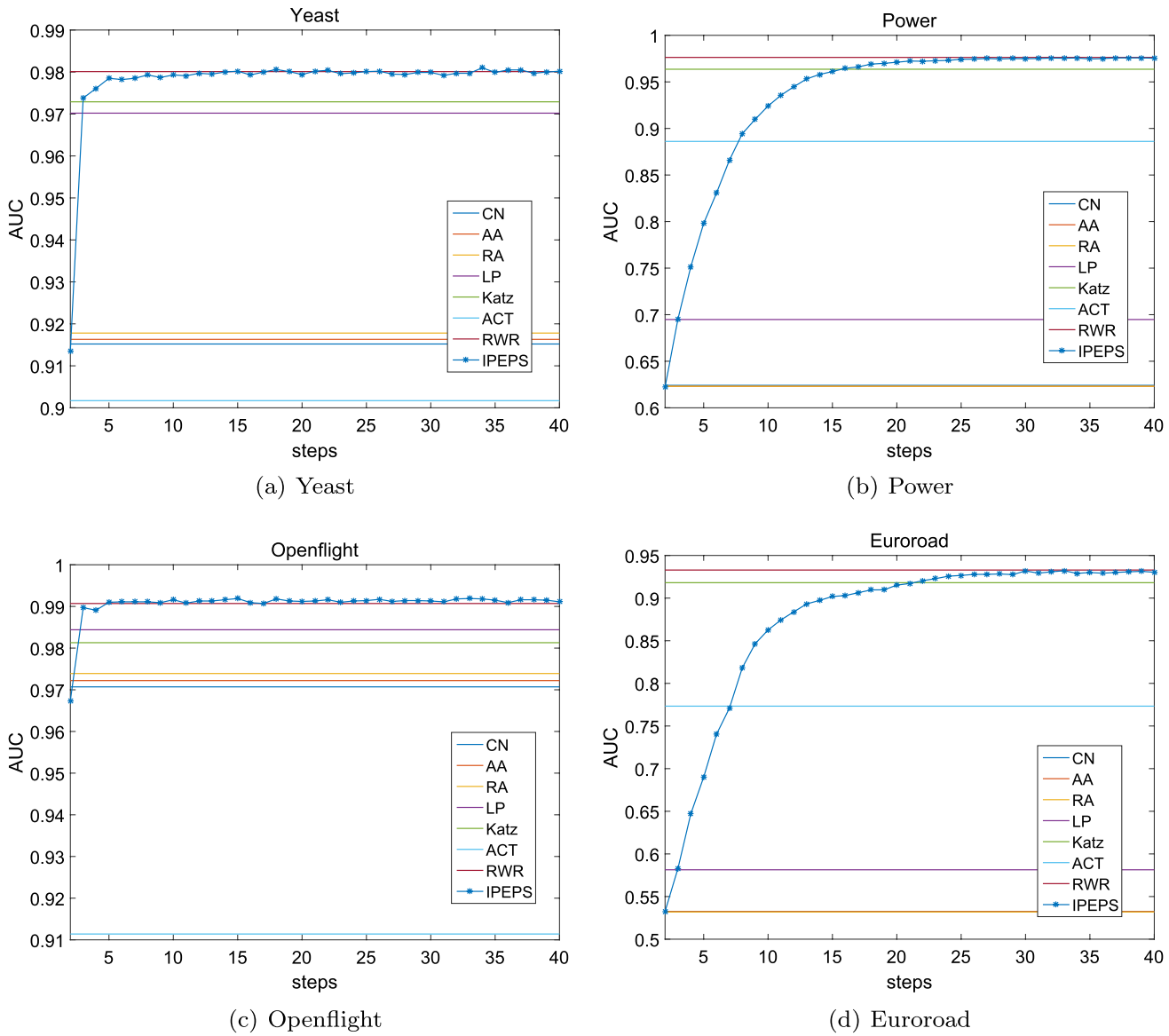


Fig. 8 The impact of different steps in four large networks

Table 3 Experiments of PEPS and IPEPS in four small networks

Dataset	PEPS		IPEPS	
	Order 2	Order 3	Order 2	Order 3
`Adjnoun`	0.6729	0.7471	0.6458	0.7519
`Celegansneural`	0.8695	0.9071	0.8541	0.9174
`Chesapeake`	0.7169	0.8188	0.6909	0.8952
`Usair`	0.97	0.9772	0.9589	0.9729

words, PEPS can detect whether two nodes are contained within one community. Although PEPS is only tested on unweight undirected networks, it is not limited to these types of networks. In the future, PEPS will be generalized to directed or weighted networks. Our proposed model is an extension of link prediction with respect to dynamic networks where nodes may disappear, a formidable task few algorithms are adept at so far. In addition, to achieve an accurate link prediction in practice, our future study will make use of both topological information and network attributes (for example, the account information in social networks).

Table 4 The AUC of IPEPS in eight real networks

Methods	Datasets								
	Adjnoun	Celegansneural	Chesapeake	Usair	Yeast	Power	Euroroad	Openflight	Average
CN	0.6761 ± 0.001475	0.8473 ± 0.0001943	0.6417 ± 0.004253	0.9551 ± 0.00003441	0.9143 ± 0.00003188	0.6241 ± 0.00003185	0.5394 ± 0.00005976	0.9692 ± 0.00001037	0.7709
AA	0.675 ± 0.00154	0.8642 ± 0.0001607	0.694 ± 0.004005	0.9673 ± 0.00003279	0.9147 ± 0.00003137	0.6231 ± 0.00005055	0.5398 ± 0.00005665	0.9716 ± 0.00001086	0.7812125
RA	0.6727 ± 0.001544	0.8698 ± 0.0001449	0.7246 ± 0.003695	0.9735 ± 0.0000321	0.9151 ± 0.0000257	0.6231 ± 0.00003889	0.5395 ± 0.00005211	0.9726 ± 0.00001022	0.7863625
LP	0.7299 ± 0.001442	0.866 ± 0.0001522	0.6961 ± 0.004108	0.9531 ± 0.00004112	0.9701 ± 0.000009274	0.6948 ± 0.000136	0.5893 ± 0.000211	0.984 ± 0.000003505	0.8104125
Katz	0.7291 ± 0.001366	0.8642 ± 0.0001619	0.6952 ± 0.004158	0.951 ± 0.00004457	0.972 ± 0.00001205	0.9636 ± 0.00001872	0.9258 ± 0.00006059	0.9811 ± 0.000003311	0.88525
ACT	0.7415 ± 0.001434	0.7456 ± 0.0002578	0.7164 ± 0.004802	0.9022 ± 0.00015	0.8989 ± 0.00002967	0.8861 ± 0.0001311	0.776 ± 0.0001535	0.9082 ± 0.00001826	0.8218625
FR	0.6753 ± 0.001558	0.8713 ± 0.0001774	0.7425 ± 0.003931	0.9734 ± 0.00002947	0.9156 ± 0.00003216	0.6255 ± 0.00005307	0.5407 ± 0.00007197	0.9724 ± 0.00000912	0.7895875
RWR	0.7431 ± 0.001513	0.8991 ± 0.00008304	0.828 ± 0.002302	0.969 ± 0.00001591	0.9791 ± 0.000006892	0.9763 ± 0.00001394	**0.9389 ± 0.00003561**	0.9912 ± 0.000001085	0.9155875
node-2vec	0.3988 ± 0.036506	0.7617 ± 0.007497	0.4659 ± 0.128306	0.7035 ± 0.012801	0.8925 ± 0.002315	0.8752 ± 0.006061	0.8296 ± 0.014767	0.7191 ± 0.003197	0.7057875
SEAL	0.7032 ± 0.05433537	0.8942 ± 0.010711266	0.7524 ± 0.097225185	0.9674 ± 0.003129945	0.9656 ± 0.002457493	0.7711 ± 0.013305678	0.7655 ± 0.021161448	0.9911 ± 0.001076092	0.8513125
IPEPS	0.7501 ± 0.001589	**0.9192 ± 0.00006689**	**0.8884 ± 0.001881**	**0.9742 ± 0.00001257**	**0.9794 ± 0.000008136**	0.9759 ± 0.000009786	0.9381 (35)	0.9915 ± 0.000001359	**0.9271**
GS-IPEPS	**0.7721 ± 0.0005124**	**0.9143 ± 0.0001562**	**0.8949 ± 0.001124**	**0.9758 ± 0.000006231**	0.9789 ± 0.00001484	**0.9764 ± 0.000006611**	0.937 ± 0.0000615	**0.9916 ± 0.000001096**	**0.930125**

The value of last column is the average AUC in all data sets, and the best performance are emphasized by bold fonts in each dataset

Table 5 The P value of T test to compare the competitive methods with $\alpha \geq 0.05$

Methods	vs. CN	vs. AA	vs. RA	vs. LP	vs. Katz	vs. ACT	vs. FR	vs. RWR	vs. node2vec	vs. SEAL
PEPS	1.5553E−14	3.5146E−13	1.4868E−12	2.7947E−11	2.6483E−07	1.4913E−23	4.2484E−13	1.6625E−03	1.3666E−25	1.9800E−10
IPEPS	1.3817E−15	2.7935E−14	1.2172E−13	1.5520E−12	6.5784E−09	5.1703E−30	1.2736E−13	4.4504E−05	1.7386E−24	4.0119E−12

Acknowledgements The authors would like to thank Yayu Zhang, Furong Lu, Honghong Cheng, Jieting Wang and Junjie Ma for their insightful discussions. This work was supported by National Natural Science Foundation of China (nos. 61672332, 61322211, 61432011, 61872226 and U1435212), the Young Scientists Fund of the National Natural Science Foundation of China (Grant no. 61802238). Program for New Century Excellent Talents in University (no. NCET-12-1031), Program for the Outstanding Innovative Teams of Higher Learning Institutions of Shanxi, and Program for the Young San Jin Scholars of Shanxi, the Natural Science Foundation of Shanxi Province (no. 201701D121052), the Research Project Supported by Shanxi Scholarship Council of China (no. 2017023).

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Adamic L, Adar E (2003) Friends and neighbors on the web. Soc Netw 25(3):211–230
2. Aziz F, Gul H, Muhammad I, Uddin I (2020) Link prediction using node information on local paths. Phys Stat Mech Appl 557:124980
3. Baird D, Ulanowicz R (1989) The seasonal dynamics of the Chesapeake bay ecosystem. Ecol Monogr 59(4):329–364
4. Barabäsi A, Albert R (2009) Emergence and scaling in random networks. Science 106(52):22073–22078
5. Batagelj V, Mrvar A (2000) Some analyses of Erdös collaboration graph. Soc Netw 22(2):173–186
6. Brin S, Page L (1998) The anatomy of a large-scale hypertextual web search engine. Comput Netw ISDN Syst 30(1):107–117
7. Cheng H, Ning Y, Yin Z, Yan C, Liu X, Zhang Z (2018) Community detection in complex networks using link prediction. Mod Phys Lett B 32(3):1850004
8. Curado M (2020) Return random walks for link prediction. Inf Sci 510:99–107
9. Fouss F, Pirotte A, Renders J, Saerens M (2007) Random-walk computation of similarities between nodes of a graph a graph with application to collaborative recommendation. IEEE Trans Knowl Data Eng 19(3):355–369
10. Freeman L (1977) A set of measures of centrality based on betweenness. Sociometry 40(1):35–41
11. Gao H, Ji S (2019) Graph u-nets. In: International conference on machine learning, pp 2083–2092
12. Goyal P, Ferrara E (2017) Graph embedding techniques and applications and and performance: a survey. Knowl Based Syst 151:78
13. Grover A, Leskovec J (2016) node2vec: Scalable feature learning for networks. In: Proceedings of the 22nd ACM International conference on knowledge discovery and data mining
14. Guo J, Shi L, Liu L (2019) Node degree and neighbourhood tightness based link prediction in social networks. In: 2019 9th International conference on information science and technology (ICIST), IEEE, pp 135–140
15. Hanely J, McNeil B (1982) The meaning and use of the area under a receiver operating characteristic (ROC) curve. Radiology 143:29–36
16. Hasan MA, Zaki M (2011) A survey of link prediction in social networks. In Social network data analytics. Springer, New York, pp 243–275
17. Jalili M, Orouskhani Y, Asgari M, Alipourfard N, Perc M (2017) Link prediction in multiplex online social networks. R Soc Open Sci 4(2):160863
18. Jeh G, Widom J (2002) Simrank: a measure of structural-context similarity. In: Eighth ACM SIGKDD International conference on knowledge discovery and data mining, pp 538–543
19. Klein D, Randic M (1993) Resistance distance. J Math Chem 12(1):81–95
20. Lebedev A, Lee J, Rivera V, Mazzara M (2017) Link prediction using top-k shortest distances. In: British International conference on databases, Springer, pp 101–105
21. Li Y, Luo P, Fan Z, Chen K, Liu J (2017) A utility-based link prediction method in social networks. Eur J Oper Res 260(2):693–705
22. Lichtenwalter R, Lussier J, Chawla N (2010) New perspectives and methods in link prediction. In: Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining, pp 243–252
23. Liu W, Lü L (2010) Link prediction based on local random walk. Europhys Lett 89(5):58007
24. Liu W, Gong M, Wang S, Ma L (2018) A two-level learning strategy based memetic algorithm for enhancing community robustness of networks. Inf Sci 422:290–304
25. Lorrain F, White H (1977) Structural equivalence of individuals in social networks. Soc Netw 1(1):67–98
26. Lü L, Zhou T (2011) Link prediction in complex networks: a survey. Phys A Stat Mech Appl 390(6):1150–1170
27. Lü L, Jin C, Zhou T (2009) Similarity index based on local path for link prediction of complex networks. Phys Rev E 80(4):046122
28. Ma C, Zhou T, Zhang H (2016) Playing the role of weak clique property in link prediction: a friend recommendation model. Sci Rep 6:1
29. Mallek S, Boukhris I, Elouedi Z, Lefevre E (2017) Evidential k-nn for link prediction. In: European conference on symbolic and quantitative approaches to reasoning and uncertainty, Springer, pp 201–211
30. Manshad M, Meybodi M, Salajegheh A (2020) A new irregular cellular learning automata-based evolutionary computation for time series link prediction in social networks. Appl Intell 51:1–14
31. Martínez V, Berzal F, Cubero J (2016) A survey of link prediction in complex networks. ACM Comput Surv (CSUR) 49(4):1–33
32. Mennens R, Scheepens R, Westenberg MA (2019) A stable graph layout algorithm for processes. Comput Graphics Forum 38(3):725–737
33. Mering CV, Krause R, Snel B, Cornell M, Stephen GO, Fields S, Bork P (2002) Comparative assessment of large-scale data sets of protein–protein interactions. Nature 417:399–403
34. Ou M, Peng C, Jian P, Zhang Z, Zhu W (2016) Asymmetric transitivity preserving graph embedding. In: The 22nd ACM SIGKDD International conference
35. Qian Y, Jiye L, Witold P, Dang C (2010) Positive approximation: an accelerator for attribute reduction in rough set theory. Artif Intell 174(9–10):597–618
36. Rossi R, Ahmed N (2015) The network data repository with interactive graph analytics and visualization. In: Proceedings of the twenty-ninth AAAI conference on artificial intelligence. http://networkrepository.com
37. Sarukkai R (2000) Link prediction and path analysis using Markov chains. Comput Netw 33(1–6):377–386
38. Soffer S, Vazquez A (2005) Network clustering coefficient without degree-correlation biases. Phys Rev E 71(5):057101
39. Sun B, Shen H, Gao J, Ouyang W, Cheng X (2017) A non-negative symmetric encoder–decoder approach for community detection. In: Proceedings of the 2017 ACM on conference on information and knowledge management. https://doi.org/10.1145/3132847.3132902

40. Tian Y, Li H, Zhu X, Tian H (2019) Link prediction based on combined influence and effective path. Int J Mod Phys B 33(22):1950249
41. Tong H, Faloutsos C, Pan J (2006) Fast random walk with restart and its applications. In: Sixth international conference on data mining (ICDM'06), IEEE, pp 613–622
42. Šubelj L, Bajec M (2011) Robust network community detection using balanced propagation. Eur Phys J B 81(3):353–362
43. Wang D, Peng C, Zhu W (2016a) Structural deep network embedding. In: the 22nd ACM SIGKDD International conference
44. Wang L, Ren J, Xu B, Li J, Luo W, Xia F (2020) Model: Motif-based deep feature learning for link prediction. IEEE Trans Comput Soc Syst 7(2):503–516
45. Wang P, Xu B, Wu Y, Zhou X (2015) Link prediction in social networks: the state-of-the-art. Sci China Inf Sci 58(1):1–38
46. Wang X, Cui P, Wang J, Pei J, Zhu W, Yang S (2017) Community preserving network embedding. In: Thirty-first AAAI conference on artificial intelligence
47. Wang Z, Liang J, Li R, Qian Y (2016b) An approach to cold-start link prediction: establishing connections between non-topological and topological information. IEEE Trans Knowl Data Eng 28(11):2857–2870
48. Watts D, Steven H (1998) Collective dynamics of small world networks. Nature 393:440–442
49. White J, Southgate E, Thomson J et al (1976) The structure of the ventral nerve cord of *Caenorhabditis elegans*. Philos Trans R Soc Lond 275(938):327
50. Xie F, Chen Z, Shang J, Feng X, Li J (2015) A link prediction approach for item recommendation with complex number. Knowl Based Syst 81(C):148–158
51. Xie Y, Zhou T, Wang B (2008) Scale-free networks without growth. Phys A Stat Mech Appl 387(7):1683
52. Xie Y, Gong M, Wang S, Yu B (2018) Community discovery in networks with deep sparse filtering. Pattern Recognit 81:50–59
53. Yao Y, Zhang R, Yang F, Tang J, Yuan Y, Hu R (2018) Link prediction in complex networks based on the interactions among paths. Phys A Stat Mech Appl 510:52–67
54. Zhang M, Chen Y (2018) Link prediction based on graph neural networks. In: Advances in Neural information processing systems, pp 5165–5175
55. Zhao Z, Zheng S, Li C, Sun J, Chang L, Francisco C (2018) A comparative study on community detection methods in complex networks. J Intell Fuzzy Syst 35(1):1077–1086
56. Zhao Z, Li C, Zhang X, Chiclana F, Enrique HV (2019) An incremental method to detect communities in dynamic evolving social networks. Knowl Based Syst 163:404–415
57. Zhong Z, Zhang Y, Pang J (2020) Neulp: An end-to-end deep-learning model for link prediction. In: International conference on web information systems engineering, Springer, pp 96–108
58. Zhu X, Tian H, Cai S (2014) Predicting missing links via effective paths. Phys A Stat Mech Appl 413(11):515–522