The Deep Fusion of Topological Structure and Attribute Information for Link Prediction

MINGQIANG ZHOU, YIHAN KONG, SHENSHEN ZHANG, DAN LIU, AND HAIJIANG JIN

College of Computer Science, Chongqing University, Chongqing 400044, China
Chongqing Key Laboratory of Software Theory and Technology, Chongqing 400044, China
Corresponding author: Mingqiang Zhou (zmqmail@cqu.edu.cn)

This work was supported in part by the National Nature Science Foundation of China under Grant 61802360, in part by the Key Research and Development Projects on Artificial Intelligence Technological Innovation of Chongqing under Grant cstc2017rgzn-zdyf0118, and in part by the Chongqing Science and Technology Project under Grant CSTC2017jgunzdyf0040.

ABSTRACT The link prediction can be used to seek missing or future links in the network, so it has become a hot research topic. The network generally contains two types of information: the topological structure of network formed by the connection between nodes, and the attribute information of nodes. However, the existing topology-based link prediction algorithms consider little attribute information. In this paper, a novel algorithm called Network Embedding with Attribute Deep Fusion for Link Prediction (NEADF-LP) is proposed. We get the embedded vectors with topological structure and attribute information by structure encoder and attribute encoder respectively, and fuse two vectors deeply. Compared with mainstream baselines on CiteSeer and Cora datasets, the results show that the deep fusion of topological structure and attribute information improve the accuracy of link prediction effectively.

INDEX TERMS Complex network, deep fusion, link prediction, network embedding.

I. INTRODUCTION

Complex networks in the real world can usually be constructed with nodes and links, in which nodes represent entities and links represent the connections between these entities [1]. Link prediction is to predict the possibility that two nodes in a network have a link by utilizing the information of nodes and the topological structure of network, which can be applied in various scientific fields, such as social networks, bio-networks, recommendation systems, stock market forecasts, and so on [2]. Link prediction has become a research focus due to its important value of science and commerce.

To deal with link prediction, a large amount of work has been done, especially in the realm of defining structural similarity criteria on the basis of the topological structure and modeling attribution knowledge.

The topological structure of the network is more versatile than the attribute information of nodes. Using the same structural similarity metrics enables higher prediction accuracy on different networks with similar structures. We can divide the mainstream methods for link prediction which based on the topological structure similarity of the network into three categories [2]. The first class is proposed on the global structure of the network, such as Katz Index [3], which calculates the topological similarity of the nodes by global structural information. The second class is called local indices. Traditional local indices simulate the similarity by calculating the number of common neighbors(CN) [4], or penalizing the large-degree nodes by setting penalizing parameter, such as Hub Promoted Index [5]. Adamic-Adar Index(AA) [6] and Resource Allocation Index(RA) [7] punish the large-degree common neighbors based on CN index. Significant Influence(SI) [2] models the significant influence by distinguishing the strong influence from the weak. The third class concentrates on quasi-local structures of the network to get the compromise between complexity and performance. The Local Path Index(LP) [8] considers short paths within two or three hops but ignores the longer paths. In recent years, some methods based on these traditional indices above are proposed to improve prediction precision. Liu et al. [9] filter out the redundant links in the network to improve the accuracy of the k-shell method from the perspective of spreading dynamics. Ahmed et al. [10] present an algorithm based on random walks in temporal networks. Path-based similarity metrics utilize more topological information of

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see http://creativecommons.org/licenses/by/4.0/
network for link prediction. However, traversing more nodes of networks will lead to higher computational complexity. Zhang and Chen [11] study a heuristic learning paradigm for link prediction. They propose a method to learn heuristics from local subgraphs using a graph neural network based on the $\gamma$-decaying heuristic theory.

The similarity indices of unstructured information could be defined by the attributes of nodes. Compared with the structural information of the network, the attributes information of the nodes is difficult to express in a structured form. The attribute-based method mainly uses attributes to calculate the similarity between the pair of nodes and predicts whether the two nodes will establish the link based on the value. The more identical attributes two nodes contain, the more similar the two nodes are [10]. For example, in the citation network, Trouillon et al. [12] predict mutual references between authors by using the information of the authors, such as schools, countries, and educational backgrounds. Wang et al. [13] extract topological information firstly by a latent-feature representation model, establish the connections between topological and non-topological information by logistic model secondly, and calculate the possibility that existing users and cold-start users are finally connected. Xie et al. [14] propose a joint prediction feature model that the characteristics of users and the network structure be taken into account, and they transform the link prediction problem into a classification problem. Sheikh et al. [15] introduce the GAT2VEC framework that generates structural contexts by structural information, and generates attribute contexts by attributes, and employs a shallow neural network model to learn a joint representation from them. Most attribute-based methods are more efficient in terms of computation time [16]. As there are many different types of attribute information, it is difficult to give a general solution. Therefore, the current attribute-based prediction method is limited to a specific attribute field, and the uniform representation of attributes cannot be performed. The algorithm performance varies significantly in different networks.

There are two main ways to improve the accuracy of link prediction, representing the network structure more reasonably and making full use of the node attribute information. With the rapid development in deep learning, word2vec [17] is proposed that is an extraordinary word embedding framework. The embedding vectors derived from the model preserve the syntactic and semantic relations between words under simple linear operations [18]. Word embedding provides a new concept in the study of social networks because it performs well to the representation requirement in large-scale social networks [19].

In recent years, many researchers have focused on network topological structure and presented various structure preserved graph embedding methods. These methods have significant effects on the representation of network topological structure. Graph embedding encodes the original graph into a low dimensional embedding space [20], and embedding vectors can be easily exploited by link prediction [21]. Applying random walk, DeepWalk [22] generates node sequences on graph firstly, and learns the node vectors with the help of the Skip-Gram [23] model. Both first-order and second-order proximities of nodes preserved, LINE [24] calculates node vectors with the two proximities respectively, and concatenates them directly. Node2vec [25] obtains node sequences via the biased random walk, which can perform BFS-like or DFS-like random walking to explore structures in different types of graphs. At present, there are many methods based on deep learning. SDNE [26], a semi-supervised deep model, exploits the first-order proximity and second-order proximity to characterize the local and global network structure. Xie et al. [27] present a network embedding framework, Sim2vec, the framework encodes more comprehensive node similarities among different nodes of the network into unified latent spaces.

Although we can obtain tolerable prediction accuracy via current link prediction algorithms, considering the insufficient combination of topological structure and attribute information, it is worth seeking a link prediction model that yields better link prediction accuracy. Besides the basic topological structure, the attribute information is abundant in network, such as user profile information in social networks, and writer information of article in the citation network. As analysis [28] shows, the performance of an algorithm can be effectively enhanced by taking external information into account, such as the attributes of nodes. The more of the same attributes two people have, such as age, gender, education, or job, the more possibility that they share the same interests and tastes. Therefore, attribute-based methods are preferable in some aspects. Song et al. [29] present a combined approach based on discriminative feature combinations that direct link prediction by the attribute information of nodes and topological structure. However, it is difficult to predetermined the weights of structure and attributes [30]. Bu et al. [31] propose Graph K-means which formulates the downstream task as a multi-objective optimization problem in a discrete-time dynamical system, and it does not need to pre-determine the weights of structure and attributes.

In this paper, we propose a novel link prediction method which considers both the topological structure and the attribute information of nodes, namely Network Embedding with Attribute Deep Fusion for Link Prediction, or abbreviated as NEADF-LP. In order to verify the performance of NEADF-LP, a series of experiments compared with state-of-the-art methods have been completed on two typical citation network datasets Cora and CiteSeer. The results show our method performs better compared with state-of-the-art methods. Moreover, NEADF-LP is still effective while a few links missed.

Our contributions are summarized as follows:

- We embed the attributes (discrete attributes and continuous attributes) of nodes in the network into low-dimensional vectors for subsequent fusion.
- We build a tower-structure model, in which the number of neurons in the current layer is halved from the
previous layer, so nonlinear fusion between structural features and attribute features could be achieved.

- We propose NEADF-LP (Network Embedding with Attribute Deep Fusion for Link Prediction), a novel link prediction method which takes both the topological structure and the attribute information of nodes into consideration.

The rest of this paper is organized as follows. Section II defines the problem description and the metric for evaluating the result. The novel approach NEADF-LP is presented in Section III. Experimental results are given and discussed in Section IV using standard datasets and metrics. The conclusion is drawn in Section V.

II. PROBLEM DESCRIPTION AND EVALUATION METRIC

Consider an undirected network $G = (V, E)$ where $V$ represents the nodes set, and $E$ represents the links set. Let $U$ denote the universal set which contains all $|V|(|V| - 1)/2$ possible node pairs in network $G$, where $|V|$ represents the number of nodes in $V$. The purpose of link prediction is to find out the missing links or future links in $U - E$, where $U - E$ denotes the non-observed or nonexistent links.

To evaluate the capability of a link prediction method, the link set $E$ is randomly partitioned into two parts: a training set $E^{\text{train}}$ and a test set $E^{\text{test}}$, such that $E^{\text{train}} \cup E^{\text{test}} = E$ and $E^{\text{train}} \cap E^{\text{test}} = \emptyset$ [28]. We use AUC, a standard metric, to evaluate the prediction performance of our proposed method. AUC is the probability that the score of a link randomly chosen in $E^{\text{test}}$ is higher than that in $U - E$. That is, we each time select a link from the test set $E^{\text{test}}$ randomly, and then from the nonexistent link set, $U - E$, randomly select another link. If the score of link in $E^{\text{test}}$ is higher than that in $U - E$, then we get 1. If two scores are equal, then we get 0.5.

After independent comparisons for $n$ times, if there are $n'$ times that the links in $E^{\text{test}}$ have higher scores and $n''$ times they are the same [2], the AUC can be presented as follow:

$$AUC = \frac{n' + 0.5n''}{n} \quad (1)$$

III. PROPOSED METHODOLOGY

A. FRAMEWORK

The framework of NEADF-LP is shown in Fig.1. Our method consists of three essential components, which we will introduce in the following parts in detail:

- Structure Encoder: the structure encoder is designed to represent each node as an embedded vector with topological structure by DeepWalk;
- Attribute Encoder: the function of the attribute encoder is to encode the discrete and continuous attributes of the node into a uniform and real-valued attribute feature vector representation of the node;
- Deep Fusion: the deep fusion utilizes the tower-structure model to deeply fuse the topological structure vector and the attribute information vector of the network to perform link prediction.

FIGURE 1. The framework of NEADF-LP. The structural and attribute information of original network is input to the structure encoder and the attribute encoder respectively, and the results that embedded vectors with topological structure and attribute feature vector are weighted input to a tower-structural model. The link probability is the output of the model.

B. STRUCTURE ENCODER

Considering the operational efficiency on large-scale data sets and the parallelism and scalability of the algorithms, we utilize DeepWalk algorithm as structure encoder to represent each node as an embedded vector with topological structure in this work.

DeepWalk learns the potential vector representation of a node by truncated random walks [22]. Comparing the word embedding model Word2Vec in the NLP domain, the basic processing element of the word embedding model is a word, but the processing element of network embedding is a node correspondingly; the word embedding is to analyze the sequence of words constituting the sentence, whereas network embedding is to analyze the path of a node which truncated random walks pass.

The algorithm mainly consists of three parts, as shown in Fig.2:

- Node sequence sampling
  The random walker samples the node sequence in the network, which selects a node $v_i$ from $G = (V, E)$ as the root node, randomly walks from the root node, and randomly selects a neighbor node of the last visited node. When the maximum path length $t$ is reached, the path through which the random walker passes is taken as a sample sequence $W_{v_i}$ and $|W_{v_i}| = t$. The sampling is repeated $\lambda$ times at node $v_i$.

FIGURE 2. The process of the structure encoding.
- **Node mapping to a vector representation**
  The sampled node sequences are treated as sentences which are input to the Skip-Gram model for training with the window size of \( w \). In the network representation learning task, a window of size \( w \) is set for the sampled node sequence \( W_{v_j} \). For each \( v_j \) mapping to the current vector space \( \Phi(v_j) \), and for each representation of the given node \( v_j \), maximize the probability of its appearance in the window \( w \) of the node sequence \( W_{v_j} \):
  \[
  \Pr \left( \{v_{i-w}, \ldots, v_{i+w} \} | \Phi(v_j) \right) = \prod_{k=i-w}^{i+w} \Pr \left( v_k | \Phi(v_j) \right)
  \]

2) **CONTINUOUS ATTRIBUTE**
Continuous attributes are ubiquitous in social networks, such as a blog with text, image, and video. These types of attributes cannot be directly compared with each other. However, by manual processing, information such as text and picture can also be converted into continuous vector representations using existing methods. TF-IDF is a weighting method to represent text data as comparable values. Term Frequency (TF) refers to the number of times a given word appears in the file. Inverse Document Frequency (IDF) is a measure of the universal importance of a word. TF-IDF is the product of the two, which can filter out common words and retain important words in the document. In the field of text information processing, Bag-of-words is usually used to obtain a vector representation of text, and TF-IDF is used to reduce noise to obtain a real-valued text vector representation. We process text attributes by TF-IDF in this work.

The main steps to calculate text features using TF-IDF are summarized as follows:
- Calculate TF. In order to facilitate the comparison of texts of different lengths, normalize TF;
- Calculate IDF;
- Calculate TF-IDF, using the product of TF and IDF;
- Splice the TF-IDF value of each word in the document to form the feature vector of the text attribute.

After the continuous attribute is represented as a real value vector, all attribute feature vectors of the nodes are stitched together, and we obtain the final attribute feature vector of node in the network. Next, the attribute information of each node is processed into feature vectors with the same dimensions. A node that lacks a certain type of attribute fills in 0 with the corresponding attribute feature position. Fig. 3 illustrates a diagram of attribute vector.

**C. ATTRIBUTE ENCODER**
Many real social network data contain a wealth of attribute information, which may be varied. In order to facilitate the combination of attribute information and topological structure, we propose an attribute encoder, which encodes the discrete attributes and continuous attributes of a node to obtain a uniform and real-valued attribute feature vector.

1) **DISCRETE ATTRIBUTE**
Discrete attributes, more commonly categorical attributes, such as gender, age, can be one-hot encoded to a continuous vector representation. This method performs one-hot encoding of all discrete attributes of the node. Finally, the coding of each type of attribute is spliced to obtain the final discrete attribute feature vector. For example, the gender attribute has two values {male, female}, and a vector {0, 1} can describe the user gender. The first element in the vector takes a value of 0 for the gender male, and the second element takes 1 for the gender female.

**D. DEEP FUSION**
As shown in Fig. 4, we build a tower-structure model to achieve nonlinear fusion between structural feature and attribute feature. The link prediction is the object-oriented, and a neural network is established to fuse the features.
We first define $f$ as a function of the similarity between $v_i$ and $v_j$, and the probability that $v_i$ and $v_j$ are connected is computed by:

$$p(v_j | v_i) = \frac{\exp(f(v_j, v_i))}{\sum_{j' = 1}^{|V|} \exp(f(v_j, v_{j'}))}$$  (5)

The degree of structural similarity between nodes needs to be extracted from the neighbors. Based on the assumption of conditional independence, the probability between node $v_i$ and the set of its neighbor nodes $N_i$ is computed by:

$$p(N_i | v_i) = \prod_{v_j \in N_i} p(v_j | v_i)$$  (6)

Then the likelihood function $L$ of the model can be defined as:

$$L = \prod_{i = 1}^{|V|} p(N_i | v_i) = \prod_{i = 1}^{|V|} \prod_{v_j \in N_i} p(v_j | v_i)$$  (7)

Next, the structure of the model is introduced in detail:

**Input layer**

The structure vector representation of the node, denoted as $u$, contains the structural information of the node; the attribute vector representation of the node is denoted as $u'$, which contains the attribute information of the node.

**Hidden layers**

The structural feature vector $u$ and the attribute feature vector $u'$ are weighted, and being input into a multi-layer perceptron, and the representation of each hidden layer is denoted as $h^0, h^1, \ldots, h^n$. The representation of the hidden layers is defined as follows:

$$h^0 = \left\{ \frac{u}{\gamma u'} \right\}$$

$$h^k = \delta_k \left( W^k h^{k-1} + b^k \right), \quad k = 1, 2, \ldots, n$$  (8)

where $\gamma$ represents the weight of the attribute feature, $\delta_k$ represents the activation function, and $n$ represents the number of the hidden layers.

Stacking multiple nonlinear hidden layers has been shown to learn better data representation. We build a tower-structure model shown in Fig.4 in the hidden layer: the number of neurons in the current layer is halved from the previous layer. This kind of tower structure can learn more abstract features, which have achieved good results in the recommendation tasks. $u$ and $u'$ need to be adjusted by the weight coefficient $\gamma$ when being input to the hidden layer.

**Output layers**

The output of hidden layer $h$ is converted into a probability vector $o$, which contains the probability that input node $v_i$ is connected to other nodes, as follow:

$$o = [p(v_1 | v_i), p(v_2 | v_i), \ldots, p(v_{|V|} | v_i)]$$  (9)

The corresponding row in the weight matrix of the hidden layer is taken as an abstract representation of node $v_i$, denoted as $u_i$. The similarity function of nodes $v_i$ and $v_j$ is defined as:

$$f(v_i, v_j) = u_j \ast h^0_i$$  (10)

Substituting (10) into (5), the connection probability in the vector $o$ is calculated by:

$$p(v_j | v_i) = \frac{\exp(u_j \ast h^0_i)}{\sum_{j' = 1}^{|V|} \exp(u_{j'} \ast h^0_i)}$$  (11)

Then the optimization objective function of the model is recorded as $\Theta$:

$$\log \Theta = \max \sum_{v_i \in V} \sum_{v_j \in N_i} \log p(v_j | v_i)$$  (12)

Substitute (11) into formula (12):

$$\log \Theta = \max \sum_{v_i \in V} \sum_{v_j \in N_i} \log \frac{\exp(u_j \ast h^0_i)}{\sum_{j' = 1}^{|V|} \exp(u_{j'} \ast h^0_i)}$$  (13)

The purpose of this optimization objective function is to make the similarity between neighbor nodes larger, and the similarity between non-neighbor nodes smaller.

Consider the gradient calculation in (12), which mainly consists of two parts:

$$\nabla \log p(v_j | v_i) = \nabla f(v_i, v_j) - \sum_{j' \in |V|} p(v_{j'} | v_i) \nabla f(v_i, v_{j'})$$  (14)

Since the second half of (14) is computationally intensive, we approximate the calculation by sampling a part of the nodes by means of negative sampling.

We summarize our algorithm in Algorithm 1.
Algorithm 1 NEADF-LP

Input: network $G = (V, E)$, attribute set $A$, embedding dimension $d$, learning rate $\eta$, negative samples $L$, attribute weight $\gamma$.

Output: the link probability with other nodes for each node.

1: Initialize all the model parameters $\varphi$.
2: Generate structural embedding vector $u$ for each node $v \in V$ using DeepWalk.
3: Generate attribute vector $u'$ by attribute information in $A$ for each node.
4: while not converged do
5:    for each $(u, u')$ do
6:        calculate the probability vector $\theta$ using (8)
7:        Sample $L$ negative samples and calculate objective function $\Theta$ using (12)
8:        Update model parameters $\varphi$ by $\frac{\partial \Theta}{\partial \varphi}$
9:    end for
10: end while

| Dataset | V  | E  | A  | AD | ACC |
|---------|----|----|----|----|-----|
| CiteSeer| 3327 | 4677 | 3704 | 1.421 | 0.08 |
| Cora    | 2708 | 5429 | 1433 | 3.891 | 0.293 |

IV. EXPERIMENT AND RESULT

A. DATASET AND EXPERIMENTAL ENVIRONMENT

This work utilizes two network datasets with attribute information, CiteSeer\(^1\) and Cora\(^2\), to study the performance of our approach. CiteSeer is a multi-disciplinary dataset consisting of papers from 10 research fields. Cora is a dataset based on citations between scientific papers. The detailed information about these datasets is described in Table 1, in which $V$ and $E$ are the numbers of nodes and links respectively; $A$ represents the dimension of the preprocessed attribute vector; AD means the average degree and ACC represents the average clustering coefficient.

The experiment is conducted on the two datasets using a single Windows server with Intel(R) Core(TM) i5-4200U CPU @2.3GHz, 512G RAM and NVIDIA GeForce 840M. The codes of our proposed models are implemented with TensorFlow 1.12 in Python 3.6.

B. PROCEDURE

The processes of our experiment are as follows:

Dataset dividing: remove the existing edges in the network by 5%, 15%, 25%, 35%, and 45% as the positive samples of the test set, and select the same number of non-existing edges as the negative samples of the test set. Train the model with the remaining existing edges as the train set.

Representation of structural feature vectors: obtain the representation of a node with topological features using the algorithm described in Section III-B.

Representation of attribute feature vectors: the discrete attribute and the continuous attribute of the node are respectively encoded using the algorithm described in Section III-C, and they are stitched together as the attribute feature vector of the node.

Training of the deep fusion model: the structural feature vectors and the attribute feature vectors are input into the deep neural network for training, and the hidden layer of the tower structure are used to fuse the structural features and the attribute features nonlinearly. The connection probability value of between one node and the other nodes are shown in the output layer.

Link prediction: calculate the AUC value using the probabilistic method according to the connection result in the previous step.

C. HYPER-PARAMETERS

According to test experiments, we set the context window size $k$, the walk length $l$ and the number of walks per node $t$ are set to 10, 80 and 16 respectively, and the dimensions of vector $d$ is 128. After data preprocessing, the initial attribute dimensions of CiteSeer and Cora are 3707 and 1433, respectively. In order to facilitate the input of attribute feature vectors into the deep learning model, the vector of attribute is also processed into 128-dimensional feature vectors. Besides, the weight of attribute feature vector $\gamma$ is set as 0.2 and 0.4 on CiteSeer and Cora respectively.

D. BASELINE METHODS

Five link prediction algorithms available in the recent literatures, including Common Neighbor (CN) [4], Local Path criterion (LP) [8], DeepWalk [22], Node2vec [25] and GraphGAN [32] algorithms are used to prove the effectiveness and robustness of our proposed method. Besides, all the compared baseline methods and the proposed NEADF-LP method are evaluated on the same criteria.

Common Neighbor (CN) [4] formulates the similarity between any two nodes by counting the number of their common friends.

Local Path criterion (LP) [8] considers the effects of second-order neighbors and third-order neighbors of nodes to develop the concept of resource allocation, where parameter $\alpha$ in LP is set as 0.01.

DeepWalk [22] is an approach to represent a node of the network as a $d$-dimension vector with the help of the truncated random walks over the graph. After that, it processes the sequences by the Skip-Gram model with hierarchical softmax. In the experiment, we set the embedding dimension as 128 and set window size $w$ as 10, random walk length as 80 and walks per node as 10.

Node2vec [25] is a generalized version of DeepWalk, which uses the biased random walks based on BFS and DFS sampling. Introducing two hyper-parameters $p$ and $q$,
it exploits structural equivalence. The parameter setting is same as DeepWalk.

GraphGAN [32] combines both generative thinking and discriminative thinking into network embedding. As a result, we can get the low-dimensional representation vectors of nodes. We set all parameters in the experiment to be optimal.

E. RESULT AND DISCUSS

As shown in Fig.5, our proposed method is compared with five baseline methods on CiteSeer and Cora real-world network. The x-axis represents the five proportions of the remaining training set, and the AUC value is shown on the y-axis. From the Fig.5, Table 2 and Table 3, we can see that NEADF-LP significantly outperforms all baselines, which indicates that this model combining topological structure and attribute information has much better predicting power on unobserved links. In addition, when the proportion of the training set is 0.55, the AUC value of NEADF-LP is about 0.8. It mainly because the attribute information of node is helpful for link prediction while the topological structure information missed.

The effect of CN is not good because it only focuses on the first order proximities. The method of CN performs better on the dataset of Cora, owing to the relatively higher average degree and clustering coefficient, which indicates that relationship between nodes are relatively close. As the proportion of the training set increases to a certain extent (0.75 on CiteSeer, 0.65 on Cora), the effect of LP method increases dramatically. The effects of DeepWalk and Node2vec are not as good as we expected, especially in the CiteSeer dataset, because the network representations, through a multitude of random walks and the shallow Skip-Gram model, is unable to capture highly nonlinear features in the underlying network. Besides, compared with DeepWalk, Node2vec performs better. That is because DeepWalk samples nodes randomly, which may result in different learned feature representations. While Node2vec designs a flexible objective that is not tied to a particular sampling strategy and provides parameters to tune the explored search space, the trouble of inflexible sampling of nodes can be solved. Although Node2vec can capture more general information from the graph, it cannot capture the structural similarities of links. Among the baselines, the performance of GraphGAN is the best in most instances (all proportions of the training set on Cora and proportions above 0.75 on CiteSeer). This might because the accessibility similarity is not encoded by GraphGAN explicitly, although it optimizes the connectivity distribution on the edges by generator and discriminator explicitly.

The link prediction methods based on topological structure have limited space for improvement. We fuse the attribute information of the node deeply on the basis of making full use of the topological structure. Experimental results show that it could improve the link prediction accuracy significantly.

F. PARAMETER SENSITIVITY

In this section, we study how the weight of attribute feature vector affect the performance of NEADF-LP. The effect of different parameter $\gamma$ is shown in Fig.6 (80% edges remained as training set).

![FIGURE 5. AUC scores of six methods for link prediction.](image)

| TABLE 2. AUC of six methods for link prediction on citeseer. |
|------------------|-----|-----|-----|-----|-----|
| Training set     | 55% | 65% | 75% | 85% | 95% |
| CN               | 0.5406 | 0.5742 | 0.6113 | 0.6833 | 0.7696 |
| LP               | 0.5769 | 0.6323 | 0.6959 | 0.8441 | 0.8917 |
| DeepWalk         | 0.6249 | 0.6469 | 0.6771 | 0.7034 | 0.7408 |
| Node2vec         | 0.7403 | 0.7724 | 0.7926 | 0.8179 | 0.8008 |
| GraphGAN         | 0.7199 | 0.7297 | 0.7647 | 0.8606 | 0.8855 |
| NEADF-LP(ours)   | 0.8013 | 0.8596 | 0.9051 | 0.9416 | 0.9435 |

| TABLE 3. AUC of six methods for link prediction on cora. |
|------------------|-----|-----|-----|-----|-----|
| Training set     | 55% | 65% | 75% | 85% | 95% |
| CN               | 0.5909 | 0.5992 | 0.6396 | 0.7006 | 0.7507 |
| LP               | 0.5968 | 0.6218 | 0.7502 | 0.8453 | 0.8770 |
| DeepWalk         | 0.6990 | 0.7394 | 0.7906 | 0.8149 | 0.8834 |
| Node2vec         | 0.7388 | 0.7825 | 0.7935 | 0.8239 | 0.8131 |
| GraphGAN         | 0.7756 | 0.8104 | 0.8493 | 0.8765 | 0.9124 |
| NEADF-LP(ours)   | 0.8110 | 0.8586 | 0.8919 | 0.9308 | 0.9519 |
As shown in Fig.6, the best weight of attribute feature vector on CiteSeer is 0.2 while on Cora is 0.4. When the parameter is set as 0, that is, without attribute information, the AUC value on both CiteSeer and Cora is lowest, which indicates that the attribute information helps a lot for link prediction. Besides, due to the large differences of attributes in different networks, the parameter of the weight of attribute information in different datasets should be determined by more specific experiments.

V. CONCLUSION AND FUTURE WORK

Link prediction has attracted more attention in various area. The existing link prediction algorithms mainly consider the structure information but ignore the attributes of the node. To improve the accuracy of link prediction, we proposed a novel method which utilizes the combination of the topological structure and the attribute information by deeply fusing them. First, we use DeepWalk algorithm as structure encoder to represent each node as an embedded vector with topological structure. Secondly, we encode the discrete attributes and continuous attributes of the node to obtain a uniform and real-valued attribute feature vector. At last, we propose the tower-structure model to deeply fuse the vector with topological structure and the vector with attribute information of the network to perform link prediction. The results of experiment demonstrate that the method proposed in this paper has better prediction power than the common benchmark algorithm, and attribute information is helpful for link prediction.

There are still some challenges ahead. Our algorithm considers a unweighted network. However, the type of nodes in the real world are more variable, and the relationship between nodes is more changeable. It makes sense to extend this work to heterogeneous networks. Besides, the random walk method is used to sample the node sequence when learning the topology of network. Sampling the node sequence in some specific way and in parallel might greatly improve the effect and efficiency of the algorithm.

REFERENCES

[1] W. Wang, L. Wu, Y. Huang, H. Wang, and R. Zhu, “Link prediction based on deep convolutional neural network,” Information, vol. 10, no. 5, p. 172, May 2019.
[2] Y. Yang, J. Zhang, X. Zhu, and L. Tian, “Link prediction via significant influence,” Phys. A, Stat. Mech. Appl., vol. 492, pp. 1523–1530, Feb. 2018.
[3] L. Katz, “A new status index derived from sociometric analysis,” Psychometrika, vol. 18, no. 1, pp. 39–43, Mar. 1953. [Online]. Available: http://leonidzhukov.net/hse/2015/socialnetworks/papers/katz-1953.pdf
[4] M. E. J. Newman, “Clustering and preferential attachment in growing networks,” Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top., vol. 64, no. 2, Jul. 2001, Art. no. 025102. [Online]. Available: http://core.ac.uk/download/pdf/22875494.pdf
[5] E. Ravasz, “Hierarchical organization of modularity in metabolic networks,” Science, vol. 297, no. 5586, pp. 1551–1555, Aug. 2002.
[6] L. A. Adamic and E. Adar, “Friends and neighbors on the Web,” Social Netw., vol. 25, no. 3, pp. 211–230, Jul. 2003. [Online]. Available: http://cond.org/fan.pdf.
[7] T. Zhou, L. Lu, and Y.-C. Zhang, “Predicting missing links via local information,” 2009, arXiv:0901.0553. [Online]. Available: http://arxiv.org/abs/0901.0553
[8] L. Lue, C.-H. Jin, and T. Zhou, “Similarity index based on local paths for link prediction of complex networks,” Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top., vol. 80, no. 4, Nov. 2009. [Online]. Available: http://doc.rero.ch/record/13217/files/zhou_sib.pdf
[9] Y. Liu, M. Tang, T. Zhou, and Y. Do, “Improving the accuracy of the k-shell method by removing redundant links-from a perspective of spreading dynamics,” 2015, arXiv:1505.07354. [Online]. Available: http://arxiv.org/abs/1505.07354
[10] N. M. Ahmed, L. Chen, Y. Wang, B. Li, Y. Li, and W. Liu, “Sampling-based algorithm for link prediction in temporal networks,” Inf. Sci., vol. 374, pp. 1–14, 2016.
[11] M. Zhang and Y. Chen, “Link prediction based on graph neural networks,” Feb. 2018, arXiv:1802.09691. [Online]. Available: https://arxiv.org/abs/1802.09691
[12] T. Trouillon, J. Welbl, S. Riedel, E. Gaussier, and G. Bouchard, “Complex embeddings for simple link prediction,” in Proc. Int. Conf. Mach. Learn., 2016, pp. 1–10.
[13] Z. Wang, J. Liang, R. Li, and Y. Qian, “An approach to cold-start link prediction: Establishing connections between non-topological and topological information,” IEEE Trans. Knowl. Data Eng., vol. 28, no. 11, pp. 2857–2870, Nov. 2016.
[14] X. Xie, Y. Li, Z. Zhang, H. Shuai, and H. Pan, “A joint link prediction method for social network,” Ifip Adv. Inf. Commun. Technol., vol. 503, pp. 56–64, Jan. 2015.
[15] N. Sheikh, Z. Kefato, and A. Montresor, “Gat2vec: Representation learning for attributed graphs,” Computing, vol. 101, no. 3, pp. 187–209, May 2019.
[16] Z. Hao, “Link prediction in online social networks based on the unsupervised marginalized denoising model,” IEEE Access, vol. 7, pp. 54133–54143, 2019.
[17] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, “Distributed representations of words and phrases and their compositionality,” in Proc. Adv. Neural Inf. Process. Syst., vol. 26, 2013, pp. 3111–3119.
[18] J. Li, Z. Liu, X. Chen, F. Xiha, X. Tan, and D. S. Song, “L-EncDB: A lightweight framework for privacy-preserving data queries in cloud computing,” Knowl.-Based Syst., vol. 79, pp. 18–26, May 2015.
[19] B. Hu, H. Wang, X. Yu, W. Yuan, and T. He, “Sparse network embedding for community detection and sign prediction in signed social networks,” J. Ambient Intell. Humanized Comput., vol. 10, no. 1, pp. 175–186, Jan. 2019.
[20] P. Cui, X. Wang, J. Pei, and W. Zhu, “A survey on network embedding,” IEEE Trans. Knowl. Data Eng., vol. 31, no. 5, pp. 833–852, May 2019.
[21] J. Lin, L. Zhang, M. He, H. Zhang, G. Liu, X. Chen, and Z. Chen, “Multipath relationship preserved social network embedding,” IEEE Access, vol. 7, pp. 26507–26618, 2019.
[22] B. Perozzi, R. Al-Rfou, and S. Skiena, “DeepWalk: Online learning of social representations,” in Proc. 20th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD), 2014, pp. 701–710.
[23] W. Shi, L. Huang, C.-D. Wang, J.-H. Li, Y. Tang, and C. Fu, “Network embedding via community based variational autoencoder,” IEEE Access, vol. 7, pp. 25323–25333, 2019.
[24] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, “LINE: Large-scale Information Network Embedding,” 2015, arXiv:1503.03578. [Online]. Available: http://arxiv.org/abs/1503.03578

[25] A. Grover and J. Leskovec, “Node2vec: Scalable feature learning for networks,” in Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD), 2016, pp. 855–864.

[26] J. Wang, B. Gong, H. Liu, and S. Li, “Model and algorithm for heterogeneous scheduling integrated with energy-efficiency awareness,” Trans. Inst. Meas. Control, vol. 38, no. 4, pp. 452–462, Apr. 2016.

[27] Y. Xie, M. Gong, S. Wang, W. Liu, and B. Yu, “Sim2vec: Node similarity preserving network embedding,” Inf. Sci., vol. 495, pp. 37–51, Aug. 2019.

[28] L. Lu and T. Zhou, “Link prediction in complex networks: A survey,” Phys. A, Stat. Mech. Appl., vol. 390, no. 6, pp. 1150–1170, Mar. 2011.

[29] A. Song, Y. Liu, Z. Wu, M. Zhai, and J. Luo, “A local random walk model for complex networks based on discriminative feature combinations,” Expert Syst. Appl., vol. 118, pp. 329–339, Mar. 2019.

[30] J. Cao, Z. Bu, Y. Wang, H. Yang, J. Jiang, and H.-J. Li, “Detecting prosumer-community groups in smart grids from the multiagent perspective,” IEEE Trans. Syst., Man, Cybern., Syst., vol. 49, no. 8, pp. 1652–1664, Aug. 2019.

[31] Z. Bu, H.-J. Li, C. Zhang, J. Cao, A. Li, and Y. Shi, “Graph K-means based on leader identification, dynamic game and opinion dynamics,” IEEE Trans. Knowl. Data Eng., to be published.

[32] H. Wang, J. Wang, J. Wang, M. Zhao, W. Zhang, F. Zhang, X. Xie, and M. Gao, “GraphGAN: Graph representation learning with generative adversarial nets,” 2017, arXiv:1711.08267. [Online]. Available: http://arxiv.org/abs/1711.08267

MINGQIANG ZHOU was born in Lanxi, Zhejiang, China, in 1977. He received the Ph.D. degree from Computer Application Technology, Chongqing University, China, in 2010. He is currently an ACM Member and an Associate Professor with the Chongqing Key Laboratory of Software Theory and Technology, College of Computer Science, Chongqing University, Chongqing, China. He has authored or coauthored more than 60 refereed publications and holds seven patents. His research interests include the different aspects of service computing, artificial intelligence, and complex networks.

YIHAN KONG was born in Chongqing, China, in 1995. He received the bachelor’s degree from the College of Computer Science, Chongqing University, China, in 2018, where he is currently a Graduate Student with the College of Computer Science. His current research interests include machine learning, service-oriented architecture, and complex networks.

SHENSHEN ZHANG was born in Jiangsu, China, in 1994. He received the bachelor’s degree from the College of Computer Science and Electronic Engineering, Hunan University, Changsha, China, in 2015, and the master’s degree with the College of Computer Science, Chongqing University, China, in 2018. His main research interests include the big data, deep learning, and complex networks.

DAN LIU was born in Neijiang, Sichuan, China, in 1995. She received the bachelor’s degree from the School of Computer Science and Technology, Donghua University, China, in 2018. She is currently a Graduate Student with the College of Computer Science, Chongqing University, China. Her main current research interests include natural language processing, deep learning, and complex networks.

HAIJUANG JIN was born in Zhangjiakou, Hebei, China, in 1995. He received the bachelor’s degree from the College of Computer Science, Chongqing University, China, in 2018, where he is currently a Graduate Student with the College of Computer Science. His current research interests include service computing, deep learning, and complex networks.