sustainability

*Article*

# A Two-Stage Deep-Learning Model for Link Prediction Based on Network Structure and Node Attributes

**Peng Liu, Liang Gui *, Huirong Wang and Muhammad Riaz**

School of Economics and Management, Jiangsu University of Science and Technology, Zhenjiang 212003, China
* Correspondence: guiliang0207@163.com

**Abstract:** Link prediction, which is used to identify the potential relationship between nodes, is an important issue in network science. In existing studies, the traditional methods based on the structural similarity of nodes make it challenging to complete the task of link prediction in large-scale or sparse networks. Although emerging methods based on deep learning can solve this problem, most of the work mainly completes the link prediction through the similarity of the representation vector of network structure information. Many empirical studies show that link formation is affected by node attributes, and similarity is not the only criterion for the formation of links in reality. Accordingly, this paper proposed a two-stage deep-learning model for link prediction (i.e, TDLP), where the node representation vector of the network structure and attributes was obtained in the first stage, while link prediction was realized through supervised learning in the second stage. The empirical results on real networks showed that our model significantly outperforms the traditional methods (e.g., CN and RA), as well as newly proposed deep-learning methods (e.g., GCN and VGAE). This study not only proposed a deep-learning framework for link prediction from the perspective of structure and attribute fusion and link distribution capture, but also lays a methodological foundation for practical applications based on link prediction.

**Keywords:** link prediction; deep learning; network structure; node attribute

**Citation:** Liu, P.; Gui, L.; Wang, H.; Riaz, M. A Two-Stage Deep-Learning Model for Link Prediction Based on Network Structure and Node Attributes. *Sustainability* **2022**, *14*, 16299. https://doi.org/10.3390/su142316299

Academic Editor: Manuel Fernandez-Veiga

Received: 13 November 2022
Accepted: 5 December 2022
Published: 6 December 2022

## 1. Introduction

Network models are often used to describe real systems in different domains, such as biology, social science, and transport systems [1–3]. Unlike random networks, these networks exhibit non-trivial structures (e.g., small world and community structures), and the formation of these structures is inseparable from the links that represent the interaction between individuals [4]. Correspondingly, predicting the future links between nodes in networks (i.e., link prediction) has become a hotspot in network science. At present, the link prediction method is widely used in practical tasks such as friend system recommendations [5,6] and knowledge graph construction [7].

For link prediction, the most widely used traditional methods are based on structural similarity, which consider that nodes with higher structural similarity are more likely to form links. For example, Zhou et al. [8] and Newman [9] used resource allocation (RA) and common neighbors (CN), respectively, to measure the similarity between nodes, which can capture potential links in the network. Traditional methods are simple and effective in some real networks, but in other cases (e.g., sparse networks) their performance is poor, and they find it especially difficult to handle high nonlinearity [10]. Fortunately, emergent graph representation learning methods provide a new opportunity to solve this problem. These methods convert the complex structure information into low-dimensional vectors to ensure that nodes with similar characteristics are closely connected, and the corresponding link prediction effect is improved [11,12].

However, the link prediction method based on deep learning is still worth further exploration. On the one hand, the existing methods mainly focus on the global or local structural information of the network. Many empirical studies show that link formation in networks is closely related to the attribute information of nodes [13–15]. For example, Wang et al. found that the similarity of individual attributes (i.e., the concept of homophily in social science) can explain 65% of the formation of links in the scientific collaboration network [14]. Therefore, the node attribute information should not be ignored in link prediction. Although some scholars have begun to explore link prediction methods that integrate network structure and node attributes, work in this area is still relatively insufficient [16–19]. On the other hand, most of the existing deep learning-based methods achieve link prediction through the similarity of representation vectors. Besides this similarity, there are many other factors, such as heterophily [20–24], that affect the link formation in real works. Accordingly, the similarity is not enough to capture the distribution of links in real networks.

Based on the above facts, in this study, we proposed a two-stage deep-learning model for link prediction, named TDLP. In the first stage, the representation vector of structural and attribute information for each node was obtained by early fusion. Then, the deep learning model was introduced in the second stage to capture the link distribution and realize the link prediction. The empirical results of real networks show that our model significantly outperformed traditional methods and newly proposed deep learning methods. This work's contribution can be summarized as follows. Theoretically, we proposed a deep-learning framework for link prediction from the perspective of structural and attributive information fusion and link distribution capture, which is not only a supplement to mainstream methods based on the similarity between structure representation vectors, but also the enrichment of methods considering node attributes. In addition, our work has also laid a foundation for practical applications based on link prediction, such as system recommendation and technology forecasting.

## 2. Related Work

### 2.1. Traditional Methods

In existing studies, similarity-based methods are widely used in traditional methods, including local similarity indices and global similarity indices [25–28]. For local similarity indices, Newman proposed the common neighbor (CN) index, emphasizing that the probability of link formation between two nodes depends on the number of common neighbors [9]. Zhou et al. improved the CN index and proposed the resource allocation (CA) index, which can suppress the influence of high-degree nodes on link prediction [8]. Subsequently, based on the idea of common neighbors, many local similarity indices have been proposed, such as Adamic/Adar Index [29], CAR-based Common Neighbor Index (CAR) [30], Node Clustering Coefficient (CCLP) [27], etc.

The global similarity indices often use the topological information of an entire network to complete the prediction task. Correspondingly, these methods have high computational complexity and are not feasible for large networks [17]. For example, the SimRank index, proposed by Jeh et al., argues that two nodes are similar if they are related to similar nodes; then, two nodes with high similarity are more likely to form a connection [31]. Tong et al. proposed a method named random walk with restart (RWR), which iteratively explores the overall structure of the network to estimate the similarity between two nodes [32].

In addition to the similarity approaches, many approaches have been developed to complete link prediction tasks, including the probabilistic and maximum likelihood approaches, matrix decomposition approaches, and clustering approaches. The probabilistic and maximum likelihood approaches optimize an object function based on existing link information, then use conditional probability to estimate the link probability between nodes [33–36]. The matrix decomposition approaches complete the link prediction by extracting the latent features of nodes and measuring latent features' similarity [37–39]. The clustering approaches employ the quantified models to capture the node-clustering pattern that affects the probability of the links' occurrence [40,41].

In short, the traditional method is widely used due to its simplicity, and is also more effective in some real networks, such as the musician collaboration network [17], the USAir network [8], the football games network, etc. These networks often have high average degree and network density. For example, the musician collaboration network contains 198 nodes and 2742 links, whose average degree and network density are 27.7 and 0.14, respectively. Correspondingly, the traditional method is more suitable for dense networks because its link prediction is based on a pairwise comparison of node structure information. In large-scale or sparse networks, the computational complexity of the traditional method will exponentially increase, and its accuracy will be reduced.

### *2.2. Deep Learning Methods*

The deep learning method maps nodes of the network from a high-dimensional space to a low-dimensional vector space; the two nodes are more likely to be linked if they are closer in the low-dimensional space. The widely used methods include methods based on random walk (e.g., DeepWalk [42] and Node2vec [43]) and the methods based on graph embedding (e.g., LINE [44], SDNE [45], GNN [12]). These methods mainly focus on the global or local structure information of networks. For example, Seongjun et al. proposed the neighborhood overlap-aware graph neural networks (Neo-GNNs) approach to complete link prediction through capturing the structure information of nodes [46]. Zhang and Chen proposed a novel graph neural networks (GNN) method that can learn the local subgraph information around each target link; the experimental results identify their method has an unprecedented performance regarding classical datasets of link prediction [47].

Besides the structural information, the attribute information of nodes also has a significant impact on the formation of links between nodes [14,20,21]. Therefore, some scholars began to explore the deep learning model, incorporating attribute information in link prediction tasks. Zhou et al. proposed a novel network embedding algorithm (NEADF-LP) to realize the combination of structure and attribute information, and this method performs better than mainstream baseline models on the CiteSeer and Cora datasets [18]. A modified deep walk-method, proposed by Kamal et al., shows stronger link prediction capability after adding information on node attributes [16]. Kipf and Welling proposed variational graph auto-encoders (VGAE) for link prediction, and the experimental results also showed that the performance of the method improved after considering the attribute information of nodes [48]. Gao et al. use graph convolution networks (GCN) to integrate the structure and attribute information and implement link prediction on matching networks [19].

In reality, network nodes are often identified as having attributes besides structural information. However, most existing studies only consider the structural information, while neglecting the attribute information of nodes [16]. Furthermore, in the methods considering node attributes, the probability of link formation is often measured by the similarity of representation vectors, which is not enough to reflect the complexity of the formation of real relationships, such as diversity and heterophily [13,20,24]. Based on a realization of network structure embedding representation and attribute information fusion, the TDLP method in the present study captures the link formation rules in the network through supervised learning and then completes link prediction, which is not only a supplement to the mainstream methods based on the similarity of structure representation vectors, but also enriches the methods considering node attributes. Nevertheless, our method has some shortcomings. First, the node representation vectors obtained through early fusion methods may cause data redundancy, which correspondingly requires dimension reduction methods to reduce the redundancy, and this increases the complexity of the method. Second, the link prediction obtained through supervised learning in the model needs enough data to ensure that it can perform, and its accuracy may decline in cases with less data.

## 3. Methodology

As shown in Figure 1, the framework of TDLP contains two stages. In the first stage, the representation vector of node structure and attribute features is obtained by early fusion, and then any pair of representation vectors is labeled according to whether there is a link between the corresponding nodes. In the second stage, a deep neural network (DNN) model used for link prediction is trained and tested by the labeled vector pairs.

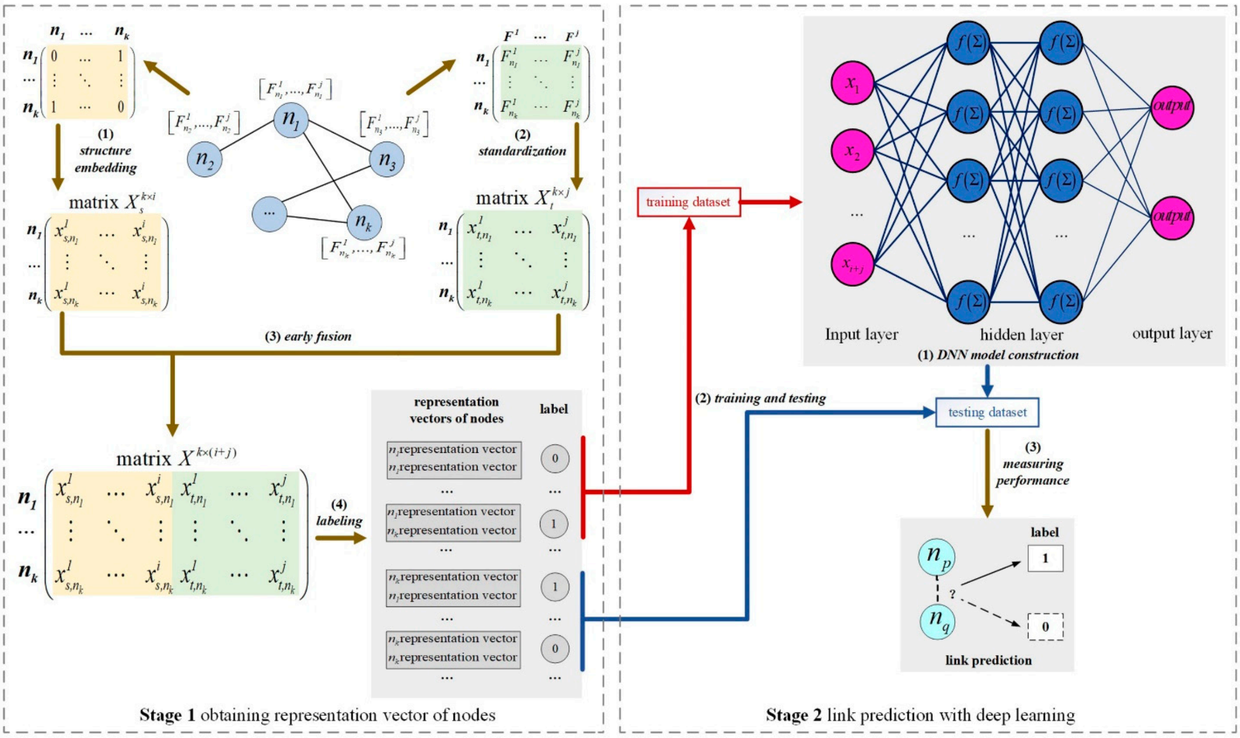
(1) structure embedding
(2) standardization
(3) early fusion
(4) labeling
matrix $X_s^{k\times i}$
matrix $X_t^{k\times j}$
matrix $X^{k\times(i+j)}$
representation vectors of nodes
label
$n_1$ representation vector
$n_2$ representation vector
$n_k$ representation vector
training dataset
testing dataset
Input layer
hidden layer
output layer
output
(1) DNN model construction
(2) training and testing
(3) measuring performance
link prediction
Stage 1 obtaining representation vector of nodes
Stage 2 link prediction with deep learning

**Figure 1.** The framework of TDLP.

### 3.1. Obtaining Representation Vector of Nodes

The representation vector of nodes is obtained through the following four steps.

(1) Representation vector of structure information

For the embedded representation of structure information, the TDLP uses the Node2vec model, which is a widely used baseline network structure embedding method. The Node2vec method obtains the representation vector through random walk, and contains four main parameters, i.e., hyper-parameter $p$ and $q$ (which are used to control the strategy of random walk), walk length $l$ and number of walks $r$. Then the matrix formed by the representation vector of structure information of all nodes is denoted as $X_s^{k\times i}$, where $k$ represents the number of nodes, and $i$ represents the dimension of the representation vector.

(2) Representation vector of attribute information

For each node, the representation vector of attribute information is obtained through the following sub-steps. First, the attributes involved in all network nodes are extracted. Second, we count the attribute status of each node and build the attribute matrix of all nodes (denoted as $A_t^{k\times j}$, where $j$ represents the dimensions of representation vectors). Finally, the matrix formed by the representation vector of the attribute information of all nodes ($X_t^{k\times j}$) is obtained through the standardization of $A_t^{k\times j}$. The standardization process is shown in Formula (1), where $x_t^{m,n}$ represents the normalized value of attribute $n$ for node $m$, and $a_t^{m,n}$ refers to the value that has not been standardized.

$$x_t^{m,n} = (a_t^{m,n} - min(A_t^{k,n}))/(max(A_t^{k,n}) - min(A_t^{k,n})) \quad (1 \le m \le k,\ 1 \le n \le j) \quad (1)$$

(3) Early fusion of representation vectors

Since the structure and attribute information of nodes have been vectorized, the early-fusion method is adopted to construct the node characteristic matrix (denoted as $X^{k\times(i+j)}$), as shown in Figure 2. Based on the matrix $X_s^{k\times i}$ and $X_t^{k\times j}$, the structure representation vector and the attribute representation vector are spliced at the node level. For example, the structure and attribute representation vectors of node $n1$ are $x^i_{s,n1}$ and $x^j_{t,n1}$, respectively. Then, these two vectors are directly concatenated to form the characteristic vector of node $n1$ (i.e., $x^{i+j}_{n1}$).

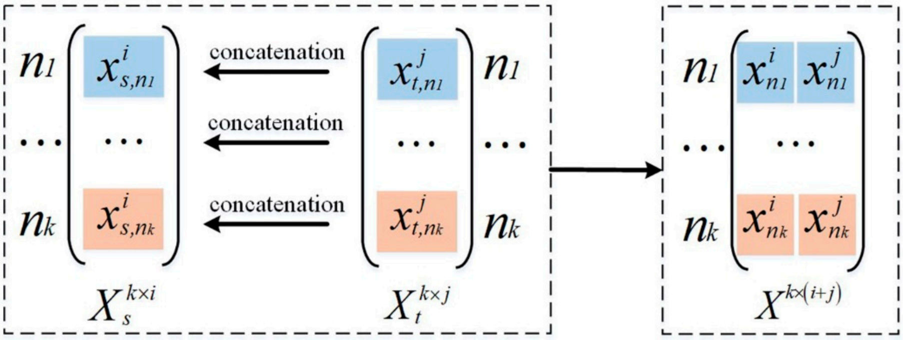

**Figure 2.** Illustration of early fusion.

(4) Data labeling

After obtaining the node characteristic matrix ($X^{k\times(i+j)}$), we label any pair of representation vectors according to whether there is a link between the corresponding nodes. If there is link between the corresponding nodes, the label of the vector pair is 1; otherwise, the label is 0 if there is no link between the corresponding nodes. Correspondingly, the link prediction can be transformed into a binary classification task based on supervised learning. Then, we select all positive samples (i.e., data labeled 1) and randomly select negative samples (i.e., data labeled 0) with 5 times of the number of positive samples to construct the dataset.

### *3.2. Link Prediction Based on Deep Learning*

This section describes the stage of link prediction in TDLP, which includes model construction, training, and testing, and measuring model performance.

(1) Model construction

In the TDLP, we employed the Deep Neural Network (DNN) model to realize link prediction. The DNN model generally consists of three parts, i.e., an input layer, hidden layer, and output layer. Its prediction ability is realized by constantly updating the weight parameters between different layers with training data. As shown in Formula (2), during the training process, the output vector $z^h$ of layer $h$ depends on the input vector of layer $(h-1)$ and its own weight matrix ($W^h$), where $b^h$ is the bias vector.

$$z^h = \sigma\left(W^h z^{h-1} + b^h\right) \tag{2}$$

In addition, since the TDLP transforms the link prediction into a binary classification between representation vector pairs, the number of neurons in the input layer is twice that of the representation vector dimension, and the number of neurons in the output layer was fixed at 2. For the hidden layer, we observed the model performance when the number of layers and neurons gradually increased, and the parameter setting was adopted when the model performance tended to be stable.

(2) Training and testing

For the dataset constructed in the first stage, we randomly selected 80% positive and negative samples as the training dataset and used the rest as the testing dataset. The general scale ratio of training dataset to testing dataset was 8 to 2.

(3) Measuring performance for the model

We used three metrics to evaluate the performance of the model. The first metric, precision ($P$), reflects the proportion of actual positive samples in all the predicted positive samples, as shown in Formula (3), where $TP$ represents the number of positive samples correctly predicted as positive samples, while $FP$ represents the number of negative samples incorrectly predicted as positive samples. The second metric, recall ($R$), refers to the proportion of correctly predicted positive samples in all true positive samples, as shown in Formula (4), where $FN$ is the number of positive samples incorrectly predicted as negative samples. The third metric ($F1$) is the harmonic mean of precision and recall, which comprehensively reflects the model performance, as shown in Formula (5).

$$P = TP/(TP + FP) \tag{3}$$

$$R = TP/(TP + FN) \tag{4}$$

$$F1 = 2PR/(P + R) \tag{5}$$

## 4. Experiments

We performed experiments on four real networks and compared our TDLP method with relevant methods to validate its effectiveness.

### *4.1. Datasets*

All the experimental networks were social interaction networks from different social groups, including developers, scholars, inventors, and college football teams. Accordingly, the means of social interaction differed, and included emailing, face-to-face contact, and so on. Detailed descriptions of these datasets are listed below.

Developer Collaboration Network (DCN): This dataset was collected from the Angular OSS community and contained 250,423 commitment records during June 2013~August 2019. Each record contained the email address of the developer, the project to which the code submission belonged, and the documents involved in this commitment. Since the software is released in the form of versions, each version can be regarded as a knowledge product completed by all developers in the version cycle. Therefore, the developer's email address was treated as the node. There was a relationship between two developers if they submitted commitments for the same file in the same version cycle, and the corresponding relationship was abstracted as a link. On this basis, we counted each developer's submissions to different projects to construct their attribute vectors.

Inventor Collaboration Network (ICN): this dataset contains 5000 patent records (2015~2021) in the field of "digital information transmission" (IPC classification number is H04L) through the incoPat database. The inventors of each patent are abstracted as nodes, and the co-inventors are regarded as cooperative relations and abstracted as edges to build an inventor cooperation network. Based upon the above argument, the authors counted the number of patents invented by each inventor in the 14 subfields (IPC Main Group) under the "digital information transmission" field, and then constructed a numerical vector to describe the attribute characteristics of the inventor.

Scientific Collaboration Network (SCN): The authors of this study selected the research dataset of the literature [49]. This dataset is the scientific collaboration network in the research field of "cooperative evolution", which not only contains the cooperative relationship between scholars, but also the keywords used by each scholar to publish his/her research articles. For all the keywords, the authors of this study carried out a unified treatment (that is, unifying the keywords of different forms). Therefore, the authors of this study clustered the keywords of all the papers and expressed each scholar's attribute characteristics by calculating the number of research articles published by each scholar on different clustered topics.

College Football Network (CFN): The CFN dataset is a real network dataset created by to the American College Football League. The network consists of 115 nodes and 616 links. The nodes in the network represent the football teams, and the link represents a game between the two football teams. The 115 football teams were divided into 12 leagues, and each league can be expressed as the attribute characteristics of the football team.

The basic information of the above four networks is shown in Table 1, including the number of nodes, the number of links, the network density and attribute dimensions of each node. It can be seen that the three networks (i.e., DCN, ICN, and SCN) are sparse (the network density is no more than 0.07), and the CFN has relative density. In addition, these networks have obvious differences in network size (i.e., the number of nodes) and attribute dimensions. The characteristics of the above data can more comprehensively test the performance of our method. On the one hand, we can examine whether TDLP performs well in networks with different densities. On the other hand, we can analyze the stability of TDLP performance in scenarios of varying network sizes and attribute dimensions.

**Table 1.** The basic information of experimental networks.

| **Network** | **Node** | **Link** | **Density** | **Dimension** |
|---|---|---|---|---|
| DCN | 1439 | 5165 | 0.005 | 6 |
| ICN | 923 | 2069 | 0.007 | 14 |
| SCN | 1127 | 3011 | 0.005 | 30 |
| CFN | 115 | 613 | 0.094 | 12 |

## *4.2. Parameter Setting*

The TDLP method consists of the Node2vec model in the first stage and the DNN model in the second stage; the parameters of these two models may influence the TDLP performance. Therefore, we examined the performance of TDLP under different parameter settings.

(1) Parameter settings of Node2vec model

For the Node2vec model, there were five parameters, including hyper-parameter $p$ and $q$, walk length $l$, number of walks $r$, and embedding dimensions $d$. Table 2 shows the TDLP performance when the hyper-parameters change and the other parameters remain consistent, where the strategy of random walk is breadth-first sampling when $(p = 0.5,\ q = 1)$, and depth-first sampling when $(\mathrm{p} = 0\ .5,\ \mathrm{q} = 2)$. It can be seen that the model performance metrics did not significantly change when $(p,\ q)$ takes two different sets of values. This indicates that the strategy of random walk in the Node2vec model has little effect on the TDLP performance. Thus, in the subsequent parameter-setting test, we fixed the value of $(\mathrm{p},\ \mathrm{q})$ at (0.5, 2).

**Table 2.** Parameter $p$ and $q$ of Node2vec.

| **Network** | **(*p*, *q*)** | ***P*** | ***R*** | ***F*1** |
|---|---|---|---|---|
| DCN | 0.5, 1 | 0.702 | 0.735 | 0.718 |
| | 0.5, 2 | 0.702 | 0.735 | 0.718 |
| ICN | 0.5, 1 | 0.741 | 0.762 | 0.751 |
| | 0.5, 2 | 0.741 | 0.762 | 0.751 |
| SCN | 0.5, 1 | 0.862 | 0.912 | 0.886 |
| | 0.5, 2 | 0.862 | 0.912 | 0.886 |
| CFN | 0.5, 1 | 0.904 | 0.667 | 0.768 |
| | 0.5, 2 | 0.904 | 0.667 | 0.768 |

Drawing on the suggestions of Ref. [26], we conducted two different groups of tests on the values of parameter $l$ and $r$ according to the network size. In the first group of tests, the value of parameter $r$ is fixed and the value of parameter $l$ changes, as shown in Table 3. In the experimental networks, with the increase in $l$ value, the model performance metrics first increase and then decrease, indicating that there is a $(l,\ r)$ combination with the optimal TDLP performance. For example, in the DCN dataset, the values of metric $P$, $R$, and $F1$ under $(l = 60,\ r = 10)$ are higher than those of other values of $(l,\ r)$. Table 4 shows the performance of TDLP under fixed $l$ and varying $r$. The model achieved optimal performance under a specific combination of $(l,\ r)$. According to the above results, in the four networks (i.e., DCN, ICN, SCN, and CFN), the optimal $(l,\ r)$ combinations are (60, 15), (50, 10), (60, 10), and (15, 10).

**Table 3.** Parameter *l* and *r* of Node2vec (fixed *r*, varying *l*).

| Network | (*l*, *r*) | *P* | *R* | *F*1 |
|---|---|---|---|---|
| DCN | 40, 10 | 0.672 | 0.702 | 0.687 |
| | 50, 10 | 0.695 | 0.721 | 0.708 |
| | 60, 10 | 0.704 | 0.732 | 0.718 |
| | 70, 10 | 0.702 | 0.693 | 0.697 |
| | 80, 10 | 0.682 | 0.665 | 0.673 |
| ICN | 40, 10 | 0.752 | 0.745 | 0.748 |
| | 50, 10 | 0.757 | 0.772 | 0.764 |
| | 60, 10 | 0.732 | 0.761 | 0.746 |
| | 70, 10 | 0.701 | 0.722 | 0.711 |
| | 80, 10 | 0.685 | 0.712 | 0.698 |
| SCN | 40, 10 | 0.912 | 0.914 | 0.913 |
| | 50, 10 | 0.915 | 0.935 | 0.925 |
| | 60, 10 | 0.918 | 0.935 | 0.926 |
| | 70, 10 | 0.902 | 0.934 | 0.918 |
| | 80, 10 | 0.906 | 0.935 | 0.920 |
| CFN | 5, 5 | 0.852 | 0.605 | 0.708 |
| | 10, 5 | 0.872 | 0.653 | 0.747 |
| | 15, 5 | 0.883 | 0.672 | 0.763 |
| | 20, 5 | 0.865 | 0.631 | 0.730 |
| | 25, 5 | 0.862 | 0.615 | 0.718 |

**Table 4.** Parameter *l* and *r* of Node2vec (fixed *l*, varying *r*).

| Network | (*l*, *r*) | *P* | *R* | *F*1 |
|---|---|---|---|---|
| DCN | 60, 5 | 0.701 | 0.722 | 0.711 |
| | 60, 10 | 0.704 | 0.732 | 0.718 |
| | 60, 15 | 0.706 | 0.732 | 0.719 |
| | 60, 20 | 0.697 | 0.721 | 0.709 |
| | 60, 25 | 0.672 | 0.715 | 0.693 |
| ICN | 50, 5 | 0.702 | 0.734 | 0.718 |
| | 50, 10 | 0.751 | 0.773 | 0.762 |
| | 50, 15 | 0.732 | 0.755 | 0.743 |
| | 50, 20 | 0.735 | 0.727 | 0.731 |
| | 50, 25 | 0.668 | 0.714 | 0.690 |
| SCN | 60, 5 | 0.902 | 0.906 | 0.904 |
| | 60, 10 | 0.918 | 0.932 | 0.925 |
| | 60, 15 | 0.907 | 0.934 | 0.920 |
| | 60, 20 | 0.903 | 0.921 | 0.912 |
| | 60, 25 | 0.874 | 0.912 | 0.893 |
| CFN | 15, 5 | 0.883 | 0.674 | 0.764 |
| | 15, 10 | 0.904 | 0.665 | 0.766 |
| | 15, 15 | 0.893 | 0.625 | 0.735 |
| | 15, 20 | 0.835 | 0.621 | 0.712 |
| | 15,25 | 0.821 | 0.637 | 0.717 |

For the last parameter $d$, we gradually increased its value from 2 to 10, and observed the change in TDLP performance, as shown in Table 5. The model showed an optimal performance in the experimental networks when $d = 4$. This indicates that the network structure information is well expressed.

**Table 5.** Parameter $d$ of Node2vec.

| **Network** | ***d*** | ***P*** | ***R*** | ***F*1** |
|---|---|---|---|---|
| DCN | 2 | 0.703 | 0.712 | 0.707 |
| | 4 | 0.703 | 0.736 | 0.719 |
| | 6 | 0.692 | 0.724 | 0.708 |
| | 8 | 0.705 | 0.683 | 0.694 |
| | 10 | 0.696 | 0.662 | 0.679 |
| ICN | 2 | 0.735 | 0.762 | 0.748 |
| | 4 | 0.752 | 0.775 | 0.763 |
| | 6 | 0.756 | 0.727 | 0.741 |
| | 8 | 0.734 | 0.747 | 0.740 |
| | 10 | 0.723 | 0.734 | 0.728 |
| SCN | 2 | 0.907 | 0.916 | 0.911 |
| | 4 | 0.913 | 0.936 | 0.924 |
| | 6 | 0.904 | 0.916 | 0.910 |
| | 8 | 0.902 | 0.924 | 0.913 |
| | 10 | 0.894 | 0.906 | 0.900 |
| CFN | 2 | 0.835 | 0.662 | 0.739 |
| | 4 | 0.903 | 0.664 | 0.765 |
| | 6 | 0.862 | 0.627 | 0.726 |
| | 8 | 0.874 | 0.615 | 0.722 |
| | 10 | 0.822 | 0.635 | 0.716 |

(2) Parameter setting of DNN model

In the DNN model, the number of hidden layers ($m$) and the number of neurons in each hidden layer ($n$) directly affect TDLP's learning ability. Thus, we further analyzed the TDLP performance under different values of $(m,\ n)$, where the set of values for $m$ and $n$ were {1, 2} and {4, 8, 16}, respectively. As shown in Table 6, the model showed a better prediction performance when the number of hidden layers was two, and there are differences in the number of hidden layer neurons for different experimental networks.

**Table 6.** Parameter $m$ and $n$ of DNN.

| **Network** | **$(m, n)$** | ***P*** | ***R*** | ***F*1** |
|---|---|---|---|---|
| DCN | $m = 1,\ n = 4$ | 0.684 | 0.625 | 0.653 |
| | $m = 1,\ n = 8$ | 0.674 | 0.685 | 0.679 |
| | $m = 1,\ n = 16$ | 0.695 | 0.716 | 0.705 |
| | $m = 2,\ n = (16,\ 8)$ | 0.694 | 0.72 | 0.707 |
| | $m = 2, n = (16, 4)$ | 0.706 | 0.732 | 0.719 |
| | $m = 2,\ n = (8,\ 4)$ | 0.621 | 0.736 | 0.674 |
| ICN | $m = 1,\ n = 4$ | 0.673 | 0.655 | 0.664 |
| | $m = 1,\ n = 8$ | 0.723 | 0.694 | 0.708 |
| | $m = 1,\ n = 16$ | 0.726 | 0.734 | 0.730 |
| | $m = 2, n = (16, 8)$ | 0.752 | 0.763 | 0.757 |
| | $m = 2,\ n = (16,\ 4)$ | 0.692 | 0.734 | 0.712 |
| | $m = 2,\ n = (8,\ 4)$ | 0.723 | 0.691 | 0.707 |
| SCN | $m = 1,\ n = 4$ | 0.882 | 0.925 | 0.903 |
| | $m = 1,\ n = 8$ | 0.875 | 0.934 | 0.904 |
| | $m = 1,\ n = 16$ | 0.914 | 0.917 | 0.915 |
| | $m = 2,\ n = (16,\ 8)$ | 0.912 | 0.924 | 0.918 |
| | $m = 2,\ n = (16,\ 4)$ | 0.914 | 0.932 | 0.923 |
| | $m = 2,\ n = (8,\ 4)$ | 0.872 | 0.934 | 0.902 |
| CFN | $m = 1,\ n = 4$ | 0.773 | 0.456 | 0.574 |
| | $m = 1,\ n = 8$ | 0.875 | 0.594 | 0.708 |
| | $m = 1,\ n = 16$ | 0.883 | 0.665 | 0.759 |
| | $m = 2,\ n = (16,\ 8)$ | 0.902 | 0.668 | 0.768 |
| | $m = 2,\ n = (16,\ 4)$ | 0.894 | 0.635 | 0.743 |
| | $m = 2,\ n = (8,\ 4)$ | 0.795 | 0.653 | 0.717 |

Based on the above analysis, Table 7 summarizes the parameter settings when the TDLP has the optimal prediction performance for different experimental networks.

**Table 7.** Parameters of TDLP.

| Network | Node2vec | | | | | DNN | |
|---|---|---|---|---|---|---|---|
| | $p$ | $q$ | $l$ | $r$ | $d$ | $m$ | $n$ |
| DCN | 0.5 | 2 | 60 | 15 | 4 | 2 | 16, 4 |
| ICN | 0.5 | 2 | 50 | 10 | 4 | 2 | 16, 8 |
| SCN | 0.5 | 2 | 60 | 10 | 4 | 2 | 16, 4 |
| CFN | 0.5 | 2 | 15 | 10 | 4 | 2 | 16, 8 |

### *4.3. Baseline Methods*

To validate the effectiveness of our TDLP method, we compared it with five widely used baseline methods, including the traditional methods (i.e., CN and RA), the deep-learning methods only considering structure information (i.e., DeepWalk and Node2Vec), and the deep learning methods that can fuse attribute information (i.e., VGAE and GCN). These methods are introduced as follows.

CN [9]: As a way of measuring the structural similarity between nodes, this uses the number of common neighbors between two nodes to measure the possibility of a link being formed between them. The more common neighbors between two nodes, the higher the probability of link formation.

RA [8]: This method is also based on structural similarity. Differing from the CN method, it takes second-order neighbors of the node into consideration. In addition, the RA method adds a penalty coefficient to restrain the effect of height nodes on the probability of link formation.

DeepWalk [42]: As a graph-embedding method, this obtains the representation vector of each node through random walk, and then uses the vector similarity to measure the possibility of link formation between nodes.

Node2vec [43]: This method is similar to the DeepWalk method, but has different random walk strategies, which are controlled by hyper-parameter $p$ and $q$. The strategy of random walk under $(p = 1,\ q = 1)$ is the same as that of DeepWalk.

GCN [50]: As a representative method of network representation learning, the GCN method uses the idea of graph convolution to realize the fusion of network topology and node attribute information and converts it to low-dimensional embedding vectors. On this basis, the link prediction is achieved through vector similarity.

VGAE [48]: The VGAE method, which is another representative method of graph representation learning, combines auto-encoder with GCN to obtain the representation vector of network topology and node attribute information, and also realizes link prediction through vector similarity. The advantage of VGAE is that the over-smoothing problem of GCN can be effectively solved by the auto-encoder.

The parameter settings for the baseline methods are summarized in Table 8. For CN and RA, considering the sparsity of the experimental network, we set the threshold of link formation to 0. For DeepWalk and Node2vec, we used the same parameter settings as TDLP. For GCN and VGAE, there were two hidden layers (the number of neurons in the two hidden layers s 16 and 8, respectively) and the learning rate was 0.01.

**Table 8.** Parameters of baseline methods.

| Method | Parameter |
|---|---|
| CN | threshold of link formation: 0 |
| RA | threshold of link formation: 0 |
| DeepWalk | the same to TDLP (see Table 7) |
| Node2vec | the same to TDLP (see Table 7) |
| GCN | learning rate: 0.01, hidden layer: (16, 8) |
| VGAE | the same to GCN |

## 5. Experimental Results and Discussion

### 5.1. Experimental Results

The experimental results are divided into two parts. In the first part, the prediction performance of the baseline method and TDLP is compared under the scenario, considering only the network structure information. In the second part, we compared TDLP and two representative graph representation learning methods (i.e., GCN and VGAE), considering both structure and attribute information.

(1) Comparison of results based on the network structure information

Table 9 shows the performance metrics of each method without considering node attribute information. In the four experimental networks, we can observe that the predictive ability of the deep-learning-based methods is much higher than that of the traditional methods. Taking SCN as an example, for CN, the values of each performance metrics (i.e., $P$, $R$, and $F1$) are 0.100, 0.012 and 0.021, respectively. For node2vec, the values of these metrics are 0.726, 0.781 and 0.752, respectively. This indicates that, compared with common neighbors, the node structure information can better describe the formation characteristics of links in the network. Furthermore, by comparing the results of deep-learning methods, the performance of TDLP model is shown to be the best in experimental networks. This is especially true for the CFN, which has a higher network density. Even for the best, most comprehensive VGAE of the three methods, the $F1$ value does not exceed 0.5, while the $F1$ value of TDLP is 0.661. This result indicates that the supervised learning in the second stage of TDLP can better capture the rule of link formation than the vector similarity.

**Table 9.** Comparison of prediction results without attribute information.

| Metrics | Method | SCN | ICN | DCN | CFN |
|---|---|---|---|---|---|
| $P$ | CN | 0.100 | 0.112 | 0.106 | 0.135 |
| | RA | 0.153 | 0.115 | 0.108 | 0.152 |
| | DeepWalk | 0.613 | 0.457 | 0.435 | 0.368 |
| | Node2vec | 0.726 | 0.532 | 0.468 | 0.391 |
| | GCN | 0.621 | 0.591 | 0.621 | 0.503 |
| | VGAE | 0.657 | 0.585 | 0.641 | 0.487 |
| | TDLP | 0.862 | 0.710 | 0.698 | 0.712 |
| $R$ | CN | 0.012 | 0.016 | 0.014 | 0.026 |
| | RA | 0.018 | 0.022 | 0.014 | 0.028 |
| | DeepWalk | 0.664 | 0.472 | 0.421 | 0.325 |
| | Node2vec | 0.781 | 0.568 | 0.485 | 0.371 |
| | GCN | 0.607 | 0.552 | 0.603 | 0.436 |
| | VGAE | 0.632 | 0.569 | 0.615 | 0.453 |
| | TDLP | 0.914 | 0.725 | 0.677 | 0.616 |
| $F1$ | CN | 0.021 | 0.028 | 0.025 | 0.044 |
| | RA | 0.032 | 0.037 | 0.025 | 0.047 |
| | DeepWalk | 0.637 | 0.464 | 0.428 | 0.345 |
| | Node2vec | 0.752 | 0.549 | 0.476 | 0.381 |
| | GCN | 0.614 | 0.571 | 0.612 | 0.467 |
| | VGAE | 0.644 | 0.577 | 0.628 | 0.469 |
| | TDLP | 0.887 | 0.717 | 0.687 | 0.661 |

(2) Comparison of results based on the network structure and node attribute information

Table 10 shows the performance metrics of the methods that can integrate the network structure and node attribute information of the experimental networks. One significant change is that the performance of all methods is significantly improved after adding the node attribute information. For example, without considering the node attribute information, the $F1$ values of VGAE in experimental networks are 0.644, 0.577, 0.628, and 0.469, respectively. After introducing the node attribute information, the $F1$ values of VGAE rise to 0.756, 0.715, 0.689, and 0.508, respectively. In addition, when considering both structure and attribute information, the TDLP method also outperforms the GCN and VGAE methods. These results indicate that the two-stage link prediction in TDLP can further enhance the deep learning model's ability to capture the distribution of links in the network. It also shows that the node attribute information cannot be ignored when link prediction is conducted, at least in the present experimental networks.

**Table 10.** Comparison of prediction results considering attribute information.

| Metrics | Method | SCN | ICN | DCN | CFN |
|---|---|---|---|---|---|
| $P$ | GCN | 0.621 | 0.591 | 0.621 | 0.503 |
| | GCN * | 0.723 | 0.625 | 0.627 | 0.515 |
| | VGAE | 0.657 | 0.585 | 0.641 | 0.487 |
| | VGAE * | 0.762 | 0.726 | 0.686 | 0.526 |
| | TDLP | 0.862 | 0.710 | 0.698 | 0.712 |
| | TDLP * | 0.902 | 0.751 | 0.705 | 0.897 |
| $R$ | GCN | 0.607 | 0.552 | 0.603 | 0.436 |
| | GCN * | 0.701 | 0.613 | 0.651 | 0.471 |
| | VGAE | 0.632 | 0.569 | 0.615 | 0.453 |
| | VGAE * | 0.751 | 0.704 | 0.693 | 0.492 |
| | TDLP | 0.914 | 0.725 | 0.677 | 0.616 |
| | TDLP * | 0.931 | 0.776 | 0.732 | 0.665 |
| $F1$ | GCN | 0.614 | 0.571 | 0.612 | 0.467 |
| | GCN * | 0.712 | 0.619 | 0.639 | 0.492 |
| | VGAE | 0.644 | 0.577 | 0.628 | 0.469 |
| | VGAE * | 0.756 | 0.715 | 0.689 | 0.508 |
| | TDLP | 0.887 | 0.717 | 0.687 | 0.661 |
| | TDLP * | 0.916 | 0.763 | 0.718 | 0.764 |

Note: * means that the represent vector of the corresponding method contains node attribute information.

### 5.2. Discussion

According to the concept of model construction, the existing work can be divided into two categories. The core idea of the first work category is to transform the structural information into the measurement of linking probability between nodes using various indices or representation vectors, such as CN in traditional methods and DeepWalk in deep learning-based methods. The second category of work emphasizes the role that node attributes play in link prediction, which focuses on the effective integration of network structure and node attribute information, while the link prediction method is relatively simple (e.g., the similarity of representation vectors in GCN and VGAE).

The model in this paper is essentially different from the above two categories of methods in terms of modeling concept. We believe that the node attributes and the capturing of link formation rules in the network are equally important in link prediction. Correspondingly, the prediction task in our model is disaggregated into two stages. The representation vectors of the network structure and node attributes are obtained in the first stage, which lays the foundation for the second stage. In the second stage, the model captures the rules of link formation through supervised learning, and then completes the link prediction.

Table 11 shows the change of TDLP comprehensive performance (i.e., the metric $F1$) on the experimental networks compared with the baseline methods. Compared with the first category of baseline methods, the prediction performance of TDLP on the experimental networks shows different degrees of improvement. On the SCN network, for example, the performance increase of TDLP over the four baseline methods (i.e., CN, RA, DeepWalk, and Node2vec) is 0.866, 0.855, 0.250, and 0.135, respectively, when only the structure information is considered. In addition, when TDLP introduces attributes, the prediction performance will be further improved. This further illustrates that the important role of node attributes in link prediction should not be ignored. Meanwhile, TDLP implements node attribute fusion based on the embedded representation of network structure, which is a complement to the prediction method that only considers the structural representation of networks. On the other hand, the performance of TDLP is also better than that of the second category of baseline methods (i.e., GCN and VGAE) which can fuse node attributes. For example, compared with GCN, the performance improvement of TDLP on four experimental networks is 0.204, 0.144, 0.079, and 0.272, respectively. This reflects that in the link prediction, besides node attributes, the link formation rule in the network is another important factor that affects the prediction result, and a more detailed method design is needed to capture it. In short, the TDLP method not only supplements and enriches the existing work, but also provides a new research perspective for link prediction based on deep learning.

**Table 11.** The change of TDLP comprehensive performance compared with the baseline methods.

| | **First Category** | | | | **Second Category** | |
|---|---|---|---|---|---|---|
| | **Traditional Methods** | | **Deep Learning Methods** | | **Deep Learning Methods** | |
| | **CN** | **RA** | **DeepWalk** | **Node2vec** | **GCN** | **VGAE** |
| SCN | +0.866<br>(+0.895) | +0.855<br>(+0.884) | +0.250<br>(+0.279) | +0.135<br>(+0.164) | +0.204 | +0.160 |
| ICN | +0.689<br>(+0.735) | +0.680<br>(+0.726) | +0.253<br>(+0.299) | +0.168<br>(+0.214) | +0.144 | +0.048 |
| DCN | +0.662<br>(+0.693) | +0.662<br>(+0.693) | +0.259<br>(+0.290) | +0.211<br>(+0.242) | +0.079 | +0.029 |
| CFN | +0.617<br>(+0.720) | +0.614<br>(+0.717) | +0.316<br>(+0.419) | +0.280<br>(+0.383) | +0.272 | +0.256 |

Note: The values in brackets of the first category are the performance improvement of TDLP after introducing node attributes.

## 6. Conclusions

In this paper, we proposed a deep learning model for link prediction (named TDLP), which divides the link prediction task into two stages. Specifically, the representation vector of network structure information and node attribute information is obtained in the first stage, while link prediction is realized through the supervised learning that takes place in the second stage. Extensive experiments on four real networks showed that the method outperforms the baseline methods, including the state-of-the-art methods. The main findings are summarized as follows.

First, based on the embedded representation of node characteristics, the TDLP method transforms the link prediction into the supervised classification task, which can more effectively capture the link distribution in the network. Its performance (accuracy, recall, and $F1$) is significantly better than that of traditional methods (e.g., CN and RA) and deep-learning-based methods (e.g., DeepWalk and Node2vec).

Second, through many experiments, we found that, compared with the results obtained when only considering the network structure information, the performance of the TDLP and two baseline methods (i.e., GCN and VGAE) was significantly improved after introducing the node attribute information. The performance metric values of the TDLP were the highest. This not only indicates that the use of attribute information can help improve the accuracy of link prediction, but also further illustrates that the TDLP method has an increased ability to capture link formation rules.

Generally, from the perspective of attribute and structure fusion and link distribution capture, we proposed a deep-learning framework for link prediction, which can be used when only considering the structure information and when considering both the structure and attribute information. Accordingly, this framework is a supplement, enriching existing research work. In addition, our work lays a methodological foundation for practical applications based on link prediction, such as system recommendations and technology forecasting. For example, accurate friend recommendation can enhance the stability of online dating community users, which is crucial to the development of the community.

In future work, we will focus on reducing the computational complexity of the TDLP method to make it more suitable for scenarios with a large number of attributes. We also aim to study link prediction considering the structural and attribute information on dynamic networks.

**Author Contributions:** Methodology, P.L. and L.G.; Validation, H.W.; Writing—original draft, M.R. All authors have read and agreed to the published version of the manuscript.

**Funding:** This work was supported by the National Natural Science Foundation of China under Grant 71871108.

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** The network data can be downloaded via https://gitee.com/liu-peng2022/network_data.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Kim, J.; Hastak, M. Social network analysis: Characteristics of online social networks after a disaster. *Int. J. Inf. Manag.* **2018**, *38*, 86–96. [CrossRef]
2. Radivojac, P.; Clark, W.T.; Oron, T.R.; Schnoes, A.M.; Wittkop, T.; Sokolov, A.; Graim, K.; Funk, C.; Verspoor, K.; Ben-Hur, A.; et al. A large-scale evaluation of computational protein function prediction. *Nat. Methods* **2013**, *10*, 221–227. [CrossRef] [PubMed]
3. Lü, L.; Zhou, T. Link prediction in complex networks: A survey. *Phys. A Stat. Mech. Its Appl.* **2011**, *390*, 1150–1170. [CrossRef]
4. Watts, D.J.; Strogatz, S.H. Collective dynamics of 'small-world' networks. *Nature* **1998**, *393*, 440–442. [CrossRef] [PubMed]
5. Santos, F.P.; Lelkes, Y.; Levin, S.A. Link recommendation algorithms and dynamics of polarization in online social networks. *Proc. Natl. Acad. Sci. USA* **2021**, *118*, 50. [CrossRef]
6. Choi, J.; Lee, J.; Yoon, J.; Jang, S.; Kim, J.; Choi, S. A two-stage deep learning-based system for patent citation recommendation. *Scientometrics* **2022**, *127*, 6615–6636. [CrossRef]
7. Nickel, M.; Murphy, K.; Tresp, V.; Gabrilovich, E. A Review of Relational Machine Learning for Knowledge Graphs. *Proc. IEEE* **2015**, *104*, 11–33. [CrossRef]
8. Zhou, T.; Lü, L.; Zhang, Y.-C. Predicting missing links via local information. *Eur. Phys. J. B* **2009**, *71*, 623–630. [CrossRef]
9. Newman, M.E.J. Clustering and preferential attachment in growing networks. *Phys. Rev. E* **2001**, *64*, 025102. [CrossRef] [PubMed]
10. Chen, J.; Zhang, J.; Xu, X.; Fu, C.; Zhang, D.; Zhang, Q.; Xuan, Q. E-LSTM-D: A Deep Learning Framework for Dynamic Network Link Prediction. *IEEE Trans. Syst. Man, Cybern. Syst.* **2019**, *51*, 3699–3712. [CrossRef]
11. Cai, L.; Ji, S. A multi-scale approach for graph link prediction. In Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020; Volume 34, pp. 3308–3315.
12. Zhang, M.; Chen, Y. Weisfeiler-lehman neural machine for link prediction. In Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, New York, NY, USA, 13–17 August 2017; pp. 575–583.
13. Jones, B.F.; Wuchty, S.; Uzzi, B. Multi-University Research Teams: Shifting Impact, Geography, and Stratification in Science. *Science* **2008**, *322*, 1259–1262. [CrossRef] [PubMed]
14. Wang, Z.-Z.; Zhu, J.J.H. Homophily versus preferential attachment: Evolutionary mechanisms of scientific collaboration networks. *Int. J. Mod. Phys. C* **2014**, *25*, 1440014. [CrossRef]
15. Aral, S.; Muchnik, L.; Sundararajan, A. Distinguishing influence-based contagion from homophily-driven diffusion in dynamic networks. *Proc. Natl. Acad. Sci. USA* **2009**, *106*, 21544–21549. [CrossRef]
16. Berahmand, K.; Nasiri, E.; Rostami, M.; Forouzandeh, S. A modified DeepWalk method for link prediction in attributed social network. *Computing* **2021**, *103*, 2227–2249. [CrossRef]
17. Kumar, A.; Singh, S.S.; Singh, K.; Biswas, B. Link prediction techniques, applications, and performance: A survey. *Phys. A Stat. Mech. Its Appl.* **2020**, *553*, 124289. [CrossRef]
18. Zhou, M.; Kong, Y.; Zhang, S.; Liu, D.; Jin, H. The Deep Fusion of Topological Structure and Attribute Information for Link Prediction. *IEEE Access* **2020**, *8*, 34398–34406. [CrossRef]
19. Gao, H.; Wang, Y.; Lyu, S.; Shen, H.; Cheng, X. GCN-ALP: Addressing Matching Collisions in Anchor Link Prediction. In Proceedings of the 2020 IEEE International Conference on Knowledge Graph (ICKG), Nanjing, China, 9–11 August 2020; pp. 412–419. [CrossRef]

20. Zhu, J.; Rossi, R.A.; Rao, A.; Mai, T.; Lipka, N.; Ahmed, N.K.; Koutra, D. Graph Neural Networks with Heterophily. In Proceedings of the Thirty-Fifth AAAI Conference on Artificial Intelligence, Virtual, 2–9 February 2021; Volume 35, pp. 11168–11176.
21. Boardman, J.D.; Domingue, B.W.; Fletcher, J.M. How social and genetic factors predict friendship networks. *Proc. Natl. Acad. Sci. USA* **2012**, *109*, 17377–17381. [CrossRef]
22. Eisenberg, E.; Levanon, E.Y. Preferential Attachment in the Protein Network Evolution. *Phys. Rev. Lett.* **2003**, *91*, 138701. [CrossRef]
23. Barabási, A.; Jeong, H.; Néda, Z.; Ravasz, E.; Schubert, A.; Vicsek, T. Evolution of the social network of scientific collaborations. *Phys. A Stat. Mech. Its Appl.* **2002**, *311*, 590–614. [CrossRef]
24. Barranco, O.; Lozares, C.; Muntanyola-Saura, D. Heterophily in social groups formation: A social network analysis. *Qual. Quant.* **2018**, *53*, 599–619. [CrossRef]
25. Kossinets, G.; Watts, D.J. Empirical Analysis of an Evolving Social Network. *Science* **2006**, *311*, 88–90. [CrossRef]
26. Leicht, E.A.; Holme, P.; Newman, M.E.J. Vertex similarity in networks. *Phys. Rev. E* **2006**, *73*, 026120. [CrossRef] [PubMed]
27. Wu, Z.; Lin, Y.; Wang, J.; Gregory, S. Link prediction with node clustering coefficient. *Phys. A Stat. Mech. Its Appl.* **2016**, *452*, 1–8. [CrossRef]
28. Lü, L.; Jin, C.-H.; Zhou, T. Similarity index based on local paths for link prediction of complex networks. *Phys. Rev. E* **2009**, *80*, 046122. [CrossRef]
29. Lada, A.; Adar, E. Friends and neighbors on the web. *Soc. Netw.* **2003**, *25*, 211–230.
30. Cannistraci, C.V.; Alanis-Lobato, G.; Ravasi, T. From link-prediction in brain connectomes and protein interactomes to the local-community-paradigm in complex networks. *Sci. Rep.* **2013**, *3*, 1613. [CrossRef] [PubMed]
31. Jeh, G.; Widom, J. Simrank: A measure of structural-context similarity. In Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, New York, NY, USA, 23–26 July 2002; pp. 538–543.
32. Tong, H.; Faloutsos, C.; Pan, J.-Y. Fast random walk with restart and its applications. In Proceedings of the Sixth International Conference on Data Mining, Washington, DC, USA, 18–22 December 2006; pp. 613–622.
33. Wang, C.; Satuluri, V.; Parthasarathy, S. Local probabilistic models for link prediction. In Proceedings of the Seventh IEEE International Conference on Data Mining, Omaha, NE, USA, 28–31 October 2007; pp. 322–331.
34. Clauset, A.; Moore, C.; Newman, M.E.J. Hierarchical structure and the prediction of missing links in networks. *Nature* **2008**, *453*, 98–101. [CrossRef]
35. Yu, K.; Chu, W.; Yu, S.; Tresp, V.; Xu, Z. Stochastic relational models for discriminative link prediction. In Proceedings of the 19th International Conference on Neural Information Processing Systems, Cambridge, MA, USA, 4–7 December 2006; pp. 1553–1560.
36. Guimerà, R.; Sales-Pardo, M. Missing and spurious interactions and the reconstruction of complex networks. *Proc. Natl. Acad. Sci. USA* **2009**, *106*, 22073–22078. [CrossRef] [PubMed]
37. Acar, E.; Dunlavy, D.M.; Kolda, T.G. Link Prediction on Evolving Data Using Matrix and Tensor Factorizations. In Proceedings of the 2009 IEEE International Conference on Data Mining Workshops, Miami, FL, USA, 6 December 2009; pp. 262–269. [CrossRef]
38. Ma, X.; Sun, P.; Qin, G. Nonnegative matrix factorization algorithms for link prediction in temporal networks using graph communicability. *Pattern Recognit.* **2017**, *71*, 361–374. [CrossRef]
39. Chen, B.; Li, F.; Chen, S.; Hu, R.; Chen, L. Link prediction based on non-negative matrix factorization. *PLoS ONE* **2017**, *12*, e0182968. [CrossRef]
40. Liu, Y.; Zhao, C.; Wang, X.; Huang, Q.; Zhang, X.; Yi, D. The degree-related clustering coefficient and its application to link prediction. *Phys. A Stat. Mech. Its Appl.* **2016**, *454*, 24–33. [CrossRef]
41. Fronczak, A.; Hoyst, J.A.; Jedynak, M.; Sienkiewicz, J. Higher order clustering coefficients in barabsialbert networks. *Phys. A: Stat. Mech. Its Appl.* **2002**, *316*, 688–694. [CrossRef]
42. Perozzi, B.; Al-Rfou, R.; Skiena, S. DeepWalk: Online learning of social representations. In Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, New York, NY, USA, 24–27 August 2014; pp. 701–710.
43. Grover, A.; Leskovec, J. node2vec: Scalable Feature Learning for Networks. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 13–17 August 2016; pp. 855–864.
44. Tang, J.; Qu, M.; Wang, M.; Zhang, M.; Yan, J.; Mei, Q. LINE: Large-scale information network embedding. *arXiv* **2015**, arXiv:1503.03578.
45. Wang, D.; Cui, P.; Zhu, W. Structural deep network embedding. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, New York, NY, USA, 13–17 August 2016; pp. 1225–1234.
46. Yun, S.; Kim, S.; Lee, J.; Kang, J.; Kim, H.J. Neo-gnns: Neighborhood overlap-aware graph neural networks for link prediction. *Adv. Neural Inf. Process. Syst.* **2021**, *34*, 13683–13694.
47. Zhang, M.; Chen, Y. Link prediction based on graph neural networks. In Proceedings of the 32nd International Conference on Neural Information Processing Systems, Red Hook, NY, USA, 3–8 December 2018; pp. 5171–5181.
48. Kipf, T.N.; Welling, M. Variational graph auto-encoders. *arXiv* **2016**, arXiv:1611.07308.
49. Liu, P.; Xia, H. Structure and evolution of co-authorship network in an interdisciplinary research field. *Scientometrics* **2015**, *103*, 101–134. [CrossRef]
50. Kipf, T.N.; Welling, M. Semi-supervised classification with graph convolutional networks. *arXiv* **2016**, arXiv:1609.02907.