A Multi-Scale Approach for Graph Link Prediction

Lei Cai,[1] **Shuiwang Ji**[2]
[1]Washington State University, [2]Texas A&M University
lei.cai@wsu.edu, sji@tamu.edu

Abstract

Deep models can be made scale-invariant when trained with multi-scale information. Images can be easily made multi-scale, given their grid-like structures. Extending this to generic graphs poses major challenges. For example, in link prediction tasks, inputs are represented as graphs consisting of nodes and edges. Currently, the state-of-the-art model for link prediction uses supervised heuristic learning, which learns graph structure features centered on two target nodes. It then learns graph neural networks to predict the existence of links based on graph structure features. Thus, the performance of link prediction models highly depends on graph structure features. In this work, we propose a novel node aggregation method that can transform the enclosing subgraph into different scales and preserve the relationship between two target nodes for link prediction. A theory for analyzing the information loss during the re-scaling procedure is also provided. Graphs in different scales can provide scale-invariant information, which enables graph neural networks to learn invariant features and improve link prediction performance. Our experimental results on 14 datasets from different areas demonstrate that our proposed method outperforms the state-of-the-art methods by employing multi-scale graphs without additional parameters.

Introduction

Multi-scale learning is an effective method to improve the performance of deep learning model for image-to-image tasks such as semantic segmentation (Chen et al. 2018b). Instances in different scales make the identifying and labeling difficult using convolutional neural networks since these instances should be labeled by considering a different range of pixels. For example, a face can be identified by only considering the human body nearby. However, Identifying a car depends on long range information. This problem can be addressed by a multi-scale convolution neural network, which takes images in different scales as input to increase the robustness of models.

Image data is easy to re-scale and applied to multi-scale learning framework since it has grid-like structures. Pixels in the image also contain locality and order information. However, in many real-world tasks like friend recommendation (Adamic and Adar 2003), movie recommendation (Koren, Bell, and Volinsky 2009), knowledge graph completion (Nickel et al. 2016) and metabolic network reconstruction (Oyetunde et al. 2016), the input data is represented as graphs for link prediction (Liben-Nowell and Kleinberg 2007). It is challenging to re-scale the graph data directly and apply it to link prediction tasks.

There have been many methods proposed to solve link prediction problems (Kipf and Welling 2016b; Perozzi, Al-Rfou, and Skiena 2014; Tang et al. 2015; Qiu et al. 2018). One class of these methods are heuristic methods (Lü and Zhou 2011), which employ a heuristic function to compute the similarity between two nodes and evaluate the existence of link. The heuristic function can compute the similarity score using a subgraph or the whole graph (Bar-Yossef and Mashiach 2008). However, these methods employ predefined heuristic function to compute the score and predict the existence of link. They cannot achieve satisfactory performance on different kinds of graphs (Zhang and Chen 2018). To enable heuristic methods to perform well on different graphs, (Zhang and Chen 2018) proposed the SEAL framework, which can learn graph structure features of an enclosing subgraph and transform the link prediction problem to a graph classification problem.

Although SEAL framework works well for link prediction tasks, it only employs a single scale enclosing subgraph as the input. SEAL employs graph convolutional neural layers (Kipf and Welling 2016a) to aggregate node features in the graph and extract representative features for each enclosing graph. Although the node information can be propagated deeper by stacking more graph convolution layers, each node is most influenced by its neighbors (Xu et al. 2018). Therefore, if nodes in a neighborhood contain similar and redundant information, the performance of graph convolution network will be affected. In this situation, long range neighbor information is required to achieve better performance for link prediction tasks.

To tackle this challenge, we propose a node aggregation method which transforms the input enclosing subgraph into different scales. By employing the proposed node aggregation method on the original graph, we can obtain a new enclosing subgraph. A series of graphs in different scales can be obtained by performing this procedure iteratively. These graphs in different scales can provide complementary information, which is useful to predict the existence of link. Our multi-scale link prediction method takes enclosing subgraphs in different scales as input and extracts graph structure features in different scales. The existence of link is predicted using hierarchical graph features. In addition, we provide theoretical analysis for our proposed method to guarantee that the aggregation method can preserve graph structure information.

Our contributions can be summarized as follows: 1) We propose a multi-scale link prediction framework, which employs a new node aggregation method to transform the graph into different scales to perform link prediction. 2) We provide theoretical analysis for our proposed method to guarantee that the aggregation method can alleviate redundant information and preserve useful information. 3) We evaluate our proposed multi-scale link prediction method on 14 datasets from different areas. Our proposed method achieves better performance compared with the baseline methods including a variant of heuristic methods, latent feature methods, and the state-of-the-art method (SEAL).

Related Work

In link prediction tasks, we are given an undirected graph $G = (V, E)$, where V is the set of vertices and $E \in V \times V$ is the set of observed links in the graph. The edges E can also be derived from the adjacency matrix A of the graph. Given any two vertices x and y in the graph, $A(x, y) = 1$ if there exists an edge between x and y and $A(x, y) = 0$ otherwise. The goal of the link prediction model is to determine whether a link is likely to exist between two nodes in the graph.

The link problem can be explored by analyzing the structure of graph centered on two nodes. The key idea is to explore which graph structure encourages two center nodes to connect. One category of link prediction method based on the graph structure is heuristic methods. Heuristic methods explore the graph structure centered on two target nodes by manually designed heuristic functions. The heuristic function can compute a similarity score between two nodes based on the graph structure. Heuristic methods include common neighbors, Adamic-Adar (Adamic and Adar 2003), preferential attachment (Barabási and Albert 1999), resource allocation (Zhou, Lü, and Zhang 2009), Katz (Katz 1953), PageRank (Brin and Page 2012), and SimRank (Jeh and Widom 2002). The drawback of heuristic methods is that the heuristic function is manually designed and cannot perform well on all datasets (Zhang and Chen 2018). For example, common neighbors can achieve good performance on social networks. However, it commonly fails on biology networks since nodes sharing more neighbors tend to disconnect with each other (Kovács et al. 2019).

To tackle this challenge, (Zhang and Chen 2017) proposed a supervised heuristic method Weisfeiler-Lehman Neural Machine (WLNM) that can learn the heuristic function automatically based on the dataset. WLNM extracted a fixed size subgraph centered on two target nodes and fed it into a fully connected neural network to predict the existence of link between two target nodes. The state-of-the-art link prediction method SEAL (Zhang and Chen 2018) is also a supervised heuristic method. When predicting the link between two nodes x and y using the SEAL framework, an h-hop enclosing graph $G^h_{(x,y)}$ consists of nodes $\{i | \min(d(i, x), d(i, y)) \leqslant h\}$ where $d(i, x)$ is the shortest path/geodesic distance between i and x. SEAL learns the feature of enclosing subgraphs by graph convolution neural networks (Kipf and Welling 2016a; Zhang et al. 2018). The extracted graph structure feature is used to predict the existence of link between two target nodes by a classifier.

A Multi-Scale Approach for Graph Link Prediction

In this section, we introduce our multi-scale link prediction method. We start with the node aggregation method and multi-scale link prediction framework. Then the information loss analysis in the aggregation procedure is described.

Node Aggregation and Multi-Scale Graph

The overall pipeline of supervised heuristic methods for link prediction tasks can be summarized as enclosing subgraph extraction, subgraph structure feature extraction, and classification based link prediction. The key to supervised heuristic link prediction is how to extract the graph structure feature. Currently, the most effective method for graph feature extraction is graph convolution networks (Bruna et al. 2014; Duvenaud et al. 2015; Niepert, Ahmed, and Kutzkov 2016; Kipf and Welling 2016a). Graph convolution networks generate the embedding feature for each node by using a linear transformation and propagate the embedding feature to its neighbors. Therefore, the representation of each node is high-level features from itself as well as its neighborhood. Although the node feature can be propagated deeper by stacking more graph convolution layers, each node is most influenced by its neighbor nodes. If the representation of a node is similar to that of its neighborhood, the node cannot obtain information from long range neighbors efficiently. In this case, the enclosing subgraph may contain redundant information that can hurt the performance of link prediction models. To enable node features to propagate to its long range neighbors efficiently, we propose a node aggregation method that can remove redundant nodes and transform the graph into a new scale. Our proposed aggregation method takes an h-hop enclosing subgraph as the input and generate a new subgraph by removing redundant nodes. By aggregating the subgraph iteratively, we can obtain a series of subgraphs in different scales. Our proposed model employs multi-scale enclosing subgraphs as inputs and extracts hierarchical features to predict the existence of link. The key intuition of our proposed method is that enclosing graphs in different scales can provide complementary information for link prediction tasks.

To aggregate redundant nodes in a neighborhood, we need to evaluate the similarity between nodes. In this work, we proposed to assign a label to each node and the label can be used to discover redundant nodes in the graph. The label

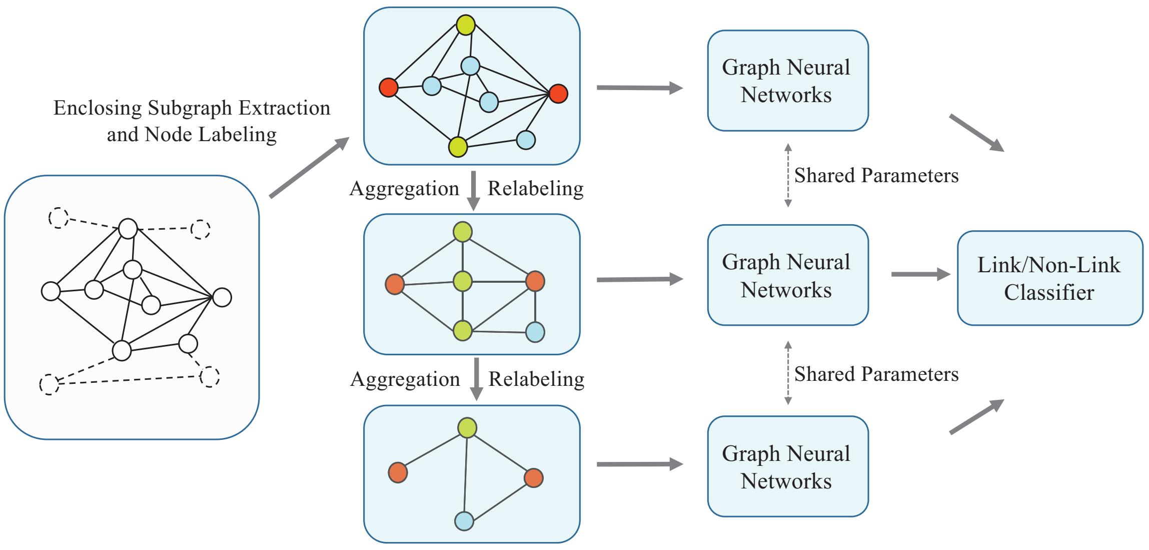

Figure 1: Diagram of our proposed method for link prediction tasks. The enclosing subgraph is extracted for link prediction. The subgraph is aggregated and relabeled in different scales. The subgraphs in different scales are fed into graph neural network for link prediction.

is also used as initial feature for each node for link prediction. The label is computed in terms of the contribution of determining the existence of link between two target nodes.

The node labeling function is designed based on two criteria: 1) Identifying the role of two target nodes. 2) Providing relative position information for each node. In this work, we employ the following function to label each node i in the enclosing subgraph centered on two target nodes x and y:

$$f_l(i) = 1 + \min(d_x, d_y) + d_x + d_y, \tag{1}$$

where $d_x = d(i,x), d_y = d(i,y)$ are minimal distance between node i and two target nodes x and y, respectively. The label of target node is set to 1. If $d_x = \infty$ or $d_y = \infty$, then $f_l(i) = 0$. Since the goal of our proposed method is to aggregate redundant nodes, we mainly focus on nodes which may contain similar information. We propose to aggregate nodes with the same label in neighborhoods:

$$\{(i,j)|f_l(i) = f_l(j) \text{ and } A(i,j) = 1\}. \tag{2}$$

After aggregating all nodes that satisfy the condition in (2) in the enclosing subgraph, we can obtain a new subgraph and relabel it using $f_l(\cdot)$. By conducting this procedure iteratively, we can obtain subgraphs in different scales, which provide complementary information for link prediction. The label of each node is used as the initial feature. Enclosing subgraphs in different scales are fed into a graph neural network to predict whether the link exists between two target nodes. The whole pipeline of our proposed model can be summarized in Figure 1. Note that our graph feature extraction network contains multiple copies of a single network and these networks share the same parameters. The graph structure extraction network can take the graph with any size as the input since parameters in the network are linear transformation matrices, which are not related to the graph size.

Our proposed method can transform an enclosing subgraph into different scales by aggregating redundant nodes. There exists an inevitable loss in terms of graph structure information during the aggregation procedure. Therefore, in the aggregation procedure we must guarantee that: 1) Only nodes containing similar information can be aggregated. 2) The aggregation procedure should not affect the relationship between the two target nodes significantly. In this work, we provide a theoretical analysis of our proposed aggregation method in terms of the two criteria.

Node-Wise Loss Analysis

We propose to aggregate nodes with the same label in a neighborhood since they contain similar information and satisfy some important properties. Here, we describe these properties in detail.

Theorem 1. *Given any non-target nodes i and j, we have that $d(i,x) + d(i,y) = d(j,x) + d(j,y)$ if $A(i,j) = 1$ and $f_l(i) = f_l(j)$.*

Theorem 2. *After aggregating the node pair (i,j) which satisfies the condition in (2) into a new node k, then $0 \leqslant f_l(i) - f_l(k) \leqslant 1$.*

The proof of Theorem 1 and Theorem 2 are described in the Appendix.

Discussion: It can be observed from Theorem 1 that the summation of distance to targets is equal for node i, j. The two nodes have the same minimal distance to target nodes and the same summation distance. Therefore, two nodes contain the same information in terms of distance relationship to target nodes. We can observe from Theorem 2 that after aggregating the label of nodes changes in a small range. Therefore, Theorem 1 guarantees that aggregated nodes contain similar information in terms of distance metric and Theorem 2 provides the boundary of the label difference for aggregated nodes. Although the changing of graph structure is more complex in practice, the experimental results show that the aggregated graph can still provide useful information for link prediction tasks. These properties can help preserve the graph structure information when aggregating nodes.

Target-Wise Loss Analysis

The graph aggregation method can aggregate redundant nodes in the graph. As discussed above, there exists an upper boundary for the changing of node label. In addition, the relationship between two target nodes should not change. If there exists a link between two target nodes in the original graph, the two nodes should also be connected in the aggregated graph. In this work, we evaluate the node relationship between two nodes using two heuristic functions: Katz index and Random Walk.

Katz index Analysis: Since the node aggregation method is proposed based on the minimal distance between nodes, we evaluate the effect of aggregating nodes using a distance based metric named Katz index. The definition of Katz index between two target nodes x and y is described as follows:

$$\text{Katz}_{x,y} = \sum_{l=1}^{\infty} \beta^l |\text{walks}^{\langle l \rangle}(x,y)|, \tag{3}$$

where β^l $(0 < \beta < 1)$ is weights for the walk with length l, $\text{walks}^{\langle l \rangle}(x,y)$ is a set walks with length l between x and y, and $|\cdot|$ counts the number of elements in the set. Katz index computes the number of walks between x and y and assigns higher weights to shorter walks.

Here, we refer the Katz index of two target nodes in original enclosing subgraph as $\text{Katz}_{x,y}$ and that in the aggragated graph as $\text{Katz}'_{x,y}$. The relationship changed between two target nodes in terms of Katz index can be expressed as $\text{Katz}_{x,y} - \text{Katz}'_{x,y}$. Our proposed aggregation method only affects walks through node i and j. Based on the analysis in the proof of Theorem 1, we have that the difference between original walk through node i and the corresponding walk after aggregation is between 0 and 1. Therefore, $\text{Katz}_{x,y} - \text{Katz}'_{x,y}$ is bounded by:

$$0 < \text{Katz}_{x,y} - \text{Katz}'_{x,y} < \beta^1 \frac{\lambda_{ij}}{|\Gamma(i) \cap \Gamma(j)|}, \tag{4}$$

where $\Gamma(\cdot)$ represents the neighbor set of a node, $|\cdot|$ counts the number of elements in the set, λ_{ij} is the number of walk through node i and j.

Random Walk Analysis: Since the influence of node x to node y in graph neural networks is equivalent to the random walk starting at x (Xu et al. 2018), we also evaluate the effect of aggregating nodes using the random walk method. The random walk computes the probability of a walker start at x and end with y. At each time step, the walker can move to a random neighbor with probability $1-\alpha$ or return to x with probability α. The random walk score from node x to y can be described as:

$$[\pi_x]_y = \alpha \sum_{w:x \rightsquigarrow y} P(w)(1-\alpha)^{|w|}, \tag{5}$$

where $w : x \rightsquigarrow y$ represents a walk from node x to y, $|w|$ is the length of the walk, and $P(w)$ is the probability of a walker traveling following w. The $[\pi_x]_y$ takes all walks from node x to y into consideration. The probability $P(w)$ of the walk $w = \langle x, v_1, \dots, v_{l-1}, y \rangle$ can is computed by:

$$P(w) = \frac{1}{|\Gamma(x)|} \prod_{i=1}^{l-1} \frac{1}{|\Gamma(v_i)|}. \tag{6}$$

To estimate the influence of aggregating node i and j, we compute the difference between $[\pi_x]_y$ in the original enclosing subgraph and $[\pi_x]'_y$ in the subgraph after aggregation. Since the aggregation procedure only affects the walks via node i and j, after aggregating the node i and j into k the difference can be represented as:

$$[\pi_x]'_y - [\pi_x]_y = \alpha\Big(\sum_{w:x \rightsquigarrow k \rightsquigarrow y} P(w)(1-\alpha)^{|w|} - \sum_{w:x \rightsquigarrow i \rightsquigarrow y} P(w)(1-\alpha)^{|w|} - \sum_{w:x \rightsquigarrow j \rightsquigarrow y} P(w)(1-\alpha)^{|w|}\Big).$$

Among walks via node i, $\alpha \sum_{w:x \rightsquigarrow i \rightsquigarrow y} P(w)(1-\alpha)^{|w|}$ can be expressed as $\frac{1}{\Gamma(i)}\theta_i$, where $\theta_i = \alpha P(w)(1-\alpha)^{|w|}\Gamma(i)$. In terms of walks via node j, $\alpha \sum_{w:x \rightsquigarrow j \rightsquigarrow y} P(w)(1-\alpha)^{|w|}$ can be expressed as $\frac{1}{\Gamma(j)}\theta_j$, where $\theta_j = \alpha P(w)(1-\alpha)^{|w|}\Gamma(j)$. Then $[\pi_x]'_y - [\pi_x]_y$ is bounded by:

$$[\pi_x]'_y - [\pi_x]_y < \frac{\theta_i + \theta_j}{|\Gamma(i) \cup \Gamma(j)|} - \frac{\theta_i}{\Gamma(i)} - \frac{\theta_j}{\Gamma(j)}. \tag{7}$$

Discussion: It can be seen from equation 4 and 7 that when two nodes share a large number of neighbor nodes, the information loss in the aggregation procedure is limited. In this work, we employed enclosing subgraphs with three different scales for link prediction tasks. Our experimental results show that if we use subgraphs in different scales individually, the performance of model using input in different scales is competitive. This demonstrates that the information loss in aggregation procedure is limited and the relationship between two target nodes does not change. Our experimental results using all subgraphs in different scales also demonstrate that enclosing subgraphs with different scales can provide complementary information, which can improve the performance of link prediction models.

Experiments

In this section, we evaluate our proposed method for link prediction tasks on 14 different datasets. We perform the qualitative evaluation in terms of area under the curve (AUC). The code and datasets are attached in supplementary.

Datasets

In the experiments, we evaluate our proposed method on 14 datasets including BUP, C.ele, USAir, SMG, EML, NSC, YST, Power, KHN, ADV, GRQ, LDG, HPD, ZWL (Watts and Strogatz 1998; Newman 2001). The details of these

Table 1: AUC comparison with baseline methods (80% training links).

Model	BUP	C.ele	USAir	SMG	EML	NSC	YST
CN	84.46(±3.12)	81.96(±2.08)	92.96(±0.80)	81.30(±0.94)	82.01(±0.68)	97.20(±0.46)	68.50(±0.68)
Jaccard	83.21(±3.13)	76.43(±2.14)	89.43(±0.93)	77.65(±1.14)	81.79(±0.66)	96.98(±0.48)	68.36(±0.68)
PA	65.18(±3.21)	75.92(±2.07)	88.79(±1.38)	83.00(±1.23)	77.89(±0.87)	97.99(±0.24)	77.37(±0.78)
AA	85.22(±3.09)	83.63(±1.91)	94.08(±0.76)	82.08(±0.92)	82.19(±0.66)	97.27(±0.46)	68.53(±0.68)
RA	85.39(±2.95)	83.96(±1.75)	94.61(±0.69)	82.05(±0.91)	82.16(±0.65)	97.28(±0.45)	68.52(±0.69)
Katz	87.10(±2.73)	84.84(±2.05)	92.01(±0.88)	86.09(±1.06)	88.45(±0.68)	98.00(±0.31)	80.56(±0.78)
PR	90.13(±2.45)	**89.14**(±1.35)	93.74(±1.01)	89.13(±0.90)	89.46(±0.63)	98.05(±0.29)	81.40(±0.75)
SR	85.47(±2.75)	75.65(±2.24)	79.21(±1.50)	78.39(±1.14)	86.90(±0.71)	97.19(±0.48)	73.93(±0.95)
ENS	85.79(±3.30)	76.09(±2.07)	88.90(±1.37)	83.14(±1.17)	78.09(±0.86)	98.00(±0.24)	77.53(±0.78)
N2V	80.25(±5.55)	80.08(±1.52)	85.40(±0.96)	78.30(±1.22)	83.06(±1.42)	96.23(±0.95)	77.07(±0.36)
SEAL	93.32(±0.84)	87.44(±1.21)	95.21(±0.77)	91.53(±0.46)	92.01(±0.38)	99.55(±0.01)	90.72(±0.25)
mLink	**93.54**(±0.63)	89.08(±0.86)	**96.43**(±0.33)	**92.05**(±0.32)	**92.03**(±0.31)	**99.65**(±0.01)	**91.40**(±0.13)

Model	Power	KHN	ADV	LDG	HPD	GRQ	ZWL
CN	57.32(±0.39)	77.83(±0.46)	88.39(±0.21)	86.40(±0.19)	71.49(±0.41)	89.56(±0.44)	91.93(±0.16)
Jaccard	57.32(±0.39)	76.19(±0.51)	86.87(±0.17)	85.33(±0.20)	71.35(±0.41)	89.57(±0.44)	91.66(±0.18)
PA	45.09(±1.03)	81.18(±0.83)	89.44(±0.23)	84.27(±0.16)	81.98(±0.31)	73.95(±0.38)	82.11(±0.25)
AA	57.32(±0.39)	78.38(±0.44)	88.71(±0.22)	86.69(±0.18)	71.55(±0.41)	89.59(±0.43)	92.08(±0.17)
RA	57.32(±0.39)	78.38(±0.44)	88.71(±0.21)	86.70(±0.18)	71.54(±0.41)	89.59(±0.43)	92.08(±0.16)
Katz	59.59(±1.51)	84.60(±0.79)	92.13(±0.21)	92.96(±0.19)	85.47(±0.35)	89.81(±0.59)	96.42(±0.12)
PR	59.88(±1.51)	88.43(±0.80)	92.78(±0.18)	94.46(±0.19)	87.19(±0.34)	89.98(±0.57)	97.20(±0.12)
SR	70.18(±0.75)	79.55(±0.90)	86.18(±0.22)	90.95(±0.14)	81.73(±0.37)	89.81(±0.58)	95.97(±0.16)
ENS	77.14(±1.36)	81.44(±0.83)	89.47(±0.23)	84.43(±0.17)	82.14(±0.31)	74.65(±0.39)	82.22(±0.25)
N2V	70.37(±1.15)	82.21(±1.19)	77.70(±0.83)	91.88(±0.56)	79.61(±1.14)	91.33(±0.53)	94.38(±0.51)
SEAL	81.37(±0.93)	92.69(±0.14)	95.07(±0.13)	96.44(±0.13)	92.26(±0.09)	97.10(±0.12)	97.46(±0.02)
mLink	**83.14**(±0.61)	**92.89**(±0.11)	**95.21**(±0.10)	**96.62**(±0.11)	**92.64**(±0.08)	**97.56**(±0.10)	**97.67**(±0.01)

datasets are summarized in Supplementary. In order to evaluate the performance of our proposed method, different percents of link are used to train these models. We randomly select 50% and 80% existing links from each graph as positive training samples. The remaining 50% and 20% links are used as positive test samples to evaluate models. Following the standard manner of learning-based link prediction, the same number of nonexistent links are sampled from the dataset as negative training samples and negative test samples, respectively.

Baseline Methods

We compare our proposed multi-scale link prediction (mLink) with eight different heuristic methods including: common neighbors (CN), Jaccard, preferential attachment (PA), Adamic-Adar (AA), resource allocation (RA), Katz, PageRank (PR), SimRank (SR). The ensemble model that employs eight heuristic scores is also compared in our experiments. We also compare our proposed method with an important latent feature method node2vec (N2V) (Grover and Leskovec 2016). The state-of-the-art method SEAL is also compared in this work. We follow the configuration of SEAL (Zhang and Chen 2018) in our experiments.

In order to guarantee that the comparison between our proposed method and the SEAL framework is fair, we employ the same graph neural network (DGCNN) (Zhang and Chen 2018) to predict the existence of link for both SEAL and our proposed mLink. The details of configuration about baseline methods and our proposed method are shown in the supplementary material.

Experiments Setup

The damping factor in the Katz method is set to 0.001. The damping factor in the PageRank (PR) is set to 0.85. The constant factor in the SimRank (SR) is set to 0.8. For node2vec, we use 128-dimensional embeddings from the software with default parameters.

The graph convolution layer employs a learnable linear transformation to generate the embedding for each node in the graph and the feature of each node is propagated to its neighborhoods. The node feature in each layer is concatenated as the final feature of each node. Since each enclosing subgraph contains a different number of nodes and edges, the sort pooling layer is employed to generate a fixed size feature vector for each graph.

In our experiments, we perform our aggregation method and relabel the enclosing subgraph for two times. Then the enclosing subgraphs in three scales are fed into a graph structure feature extraction network. The feature extraction networks for graphs in different scales share the same parameters. We employ four graph convolution layers to extract node features. The number of channel for four graph convolution layers is set to 32, 32, 32, 1. The ratio of sort pooling layer is set to 0.6. The classification network consists of two 1-D convolution layers and a fully connected layer. The number of channel for two 1-D convolution layers is set to 16, 32. We train the DGCNN network for 50 epochs. The hop number h is set to 2 for those graphs.

Table 2: AUC comparison with baseline methods (50% training links).

Model	BUP	C.ele	USAir	SMG	EML	NSC	YST
CN	73.62(±2.06)	72.29(±0.82)	87.93(±0.43)	70.26(±0.54)	70.80(±0.44)	91.87(±0.68)	61.37(±0.29)
Jaccard	72.93(±2.06)	69.75(±0.86)	84.82(±0.52)	69.06(±0.66)	70.74(±0.45)	91.73(±0.72)	61.33(±0.30)
PA	64.21(±2.37)	73.81(±0.97)	87.59(±0.50)	80.87(±0.47)	76.53(±0.27)	96.31(±0.58)	75.92(±0.40)
AA	74.08(±2.00)	73.37(±0.80)	88.61(±0.40)	70.68(±0.49)	70.87(±0.43)	91.91(±0.68)	61.38(±0.29)
RA	74.12(±1.97)	73.42(±0.82)	88.73(±0.39)	70.67(±0.49)	70.87(±0.43)	91.92(±0.68)	61.38(±0.29)
Katz	81.61(±3.40)	79.99(±0.59)	88.91(±0.39)	80.65(±0.58)	84.16(±0.64)	95.99(±0.62)	77.28(±0.37)
PR	84.07(±3.39)	**84.95**(±0.58)	90.57(±0.39)	84.59(±0.45)	85.43(±0.63)	96.06(±0.60)	77.90(±3.69)
SR	80.98(±3.03)	76.05(±0.80)	81.09(±0.59)	75.28(±0.74)	83.05(±0.64)	95.59(±0.68)	73.71(±0.41)
ENS	76.52(±7.51)	74.11(±0.96)	87.81(±0.50)	81.06(±0.47)	76.84(±0.28)	96.32(±0.58)	76.08(±0.39)
N2V	80.94(±2.65)	75.53(±1.23)	84.63(±1.58)	73.50(±1.22)	80.15(±1.26)	94.20(±1.25)	73.62(±0.74)
SEAL	85.10(±0.82)	81.23(±1.52)	93.23(±1.46)	86.56(±0.53)	85.83(±0.46)	99.07(±0.02)	85.56(±0.28)
mLink	**87.50**(±0.53)	83.49(±0.93)	**94.65**(±0.36)	**88.49**(±0.38)	**87.06**(±0.35)	**99.17**(±0.01)	**87.80**(±0.17)
Model	Power	KHN	ADV	LDG	HPD	GRQ	ZWL
CN	53.58(±0.22)	66.40(±0.25)	79.70(±0.20)	74.23(±0.19)	62.76(±0.11)	78.52(±0.19)	81.80(±0.01)
Jaccard	53.38(±0.22)	66.02(±0.25)	79.02(±0.20)	73.92(±0.20)	62.71(±0.11)	78.52(±0.18)	81.66(±0.01)
PA	46.79(±0.69)	78.04(±0.46)	88.61(±0.15)	82.24(±0.16)	80.42(±0.10)	73.00(±0.28)	81.17(±0.14)
AA	53.38(±0.22)	66.60(±0.25)	79.90(±0.19)	74.37(±0.18)	62.77(±0.11)	78.53(±0.18)	81.89(±0.08)
RA	53.38(±0.22)	66.60(±0.25)	79.89(±0.19)	74.37(±0.18)	62.77(±0.11)	78.53(±0.18)	81.89(±0.08)
Katz	57.34(±0.51)	78.99(±0.20)	90.04(±0.17)	88.61(±0.19)	81.60(±0.12)	82.50(±0.21)	93.72(±0.06)
PR	57.34(±0.52)	82.34(±0.21)	90.97(±0.15)	90.50(±0.19)	83.15(±0.17)	82.64(±0.22)	**95.11**(±0.09)
SR	56.16(±0.45)	75.87(±0.19)	84.87(±0.14)	87.95(±0.14)	78.88(±0.22)	82.68(±0.24)	94.00(±0.10)
ENS	62.70(±0.95)	78.16(±0.46)	88.66(±0.16)	82.50(±0.17)	80.58(±0.10)	73.60(±0.27)	81.37(±0.13)
N2V	55.40(±0.84)	78.53(±0.72)	74.67(±0.98)	88.82(±0.44)	75.84(±1.03)	84.24(±0.35)	92.06(±0.61)
SEAL	65.80(±1.10)	87.43(±0.17)	92.75(±0.14)	92.98(±0.16)	88.05(±0.10)	90.07(±0.15)	94.94(±0.02)
mLink	**67.02**(±0.63)	**88.10**(±0.15)	**93.06**(±0.10)	**93.13**(±0.12)	**88.42**(±0.09)	**92.44**(±0.12)	95.05(±0.01)

Table 3: AUC comparison with enclosing subgraphs in different scales (50% training links).

Model	BUP	C.ele	USAir	SMG	EML	NSC	YST
Scale-1	85.10(±0.82)	81.23(±1.52)	93.23(±1.46)	86.56(±0.53)	85.83(±0.46)	99.07(±0.02)	85.56(±0.28)
Scale-2	84.94(±0.79)	80.34(±1.33)	92.05(±1.26)	87.06(±0.47)	83.74(±0.42)	99.04(±0.01)	85.97(±0.26)
Scale-3	84.47(±0.81)	77.71(±1.02)	90.15(±0.97)	86.04(±0.49)	83.94(±0.43)	99.03(±0.02)	85.93(±0.25)
All Scales	**87.50**(±0.53)	83.49(±0.93)	**94.65**(±0.36)	**88.49**(±0.38)	**87.06**(±0.35)	**99.17**(±0.01)	**87.80**(±0.17)
Model	Power	KHN	ADV	LDG	HPD	GRQ	ZWL
Scale-1	65.80(±1.10)	87.43(±0.17)	92.75(±0.14)	92.98(±0.16)	88.05(±0.10)	90.07(±0.15)	94.94(±0.02)
Scale-2	65.36(±1.02)	88.03(±0.16)	92.23(±0.13)	92.87(±0.15)	88.27(±0.13)	90.78(±0.09)	94.64(±0.02)
Scale-3	65.37(±1.04)	87.94(±0.16)	90.98(±0.14)	92.36(±0.15)	88.15(±0.14)	90.72(±0.09)	93.71(±0.02)
All Scales	**67.02**(±0.63)	**88.10**(±0.15)	**93.06**(±0.10)	**93.13**(±0.12)	**88.42**(±0.09)	**92.44**(±0.12)	**95.05**(±0.01)

Results and Analysis

We evaluate our proposed method (mLink) on 14 different datasets from different areas. We perform each method for ten times. The average AUC results and standard deviations are shown in Table 1 and Table 2. It can be seen from results that SEAL and mLink generally achieve better performance than heuristic methods and latent feature method since these methods can learn graph structure features based on the given dataset to perform link prediction. In C.ele and ZWL datasets, PageRank method performs better than our proposed method. However, it only works well in some networks from special areas. By employing multi-scale graphs, our proposed method outperforms the SEAL framework by large margins on most datasets in terms of AUC and standard deviations, which also demonstrates multi-scale enclosing subgraph can improve the performance for link prediction tasks without introducing additional parameters. Our proposed method can outperform the state-of-the-art method SEAL both using 50% and 80% training links. We can observe from results that our proposed method can perform well even with only 50% training links. When using 80% training links, both SEAL and our proposed method can achieve satisfactory performance. If we only use 50% training links, our proposed method can outperform SEAL significantly. Our proposed multi-scale link prediction method can also be considered as a data-augmentation method for link prediction.

Information Loss Analysis

To demonstrate that the aggregated subgraph does not lose important structure information, we train the graph neural network with three single scale graphs. We denote the original enclosing subgraph as Scale-1. The aggregated graphs are represented as Scale-2 and Scale-3, respectively. The

Table 4: Averaged AUC comparison with Multi-scale graph embedding methods (50% training links).

Model	BUP	C.ele	USAir	SMG	EML	NSC	YST	KHN	ADV	LDG	HPD	GRQ	ZWL
Role2Vec	85.85	75.74	82.37	73.67	76.73	97.62	76.5	73.89	67.24	86.95	74.72	92.76	85.75
HARP	83.90	77.06	79.11	72.87	82.22	97.82	76.15	81.79	71.80	89.09	74.18	85.84	87.17
mLink	**87.50**	**83.49**	**94.65**	**88.49**	**87.06**	**99.17**	**87.80**	**92.89**	**95.21**	**96.62**	**92.64**	**97.56**	**97.67**

AUC results using three single graphs are shown in Table 3. It can be seen from results that models using graphs in different scales achieve comparative performance. It also demonstrates that the structure information is preserved during the aggregation procedure. The model that employs all three graphs achieve the best performance since graphs in different scales can provide complementary information.

Comparasion with Multi-scale Graph Embedding Methods

We also compare our method with multi-scale graph embedding methods including Role2Vec (Ahmed et al. 2018) and HARP (Chen et al. 2018a). Experimental results are shown in Table 4. We can see that our method can outperform the multi-scale graph embedding methods. The two methods are proposed to learn the embedding in a hierarchical manner. Both HARP and R2V belong to node embedding methods. When links in the graph are limited, the performance will be affected, especially for the plain graph without node attributes.

Conclusions

In this work, we explore the supervised heuristic method for link prediction tasks. To enable the graph neural network extract features from graphs in different scales and improve the performance of supervised heuristic models, we propose a novel node aggregation method to transform the graph into different scales while preserving important information. The theory analysis is also provided for discussing information loss in the re-scaling procedure. We compare our proposed method with baseline methods on 14 datasets from different areas. Our experimental results show that our proposed method can achieve better performance by employing inputs in different scales.

Appendix

A. Proof of Theorem 1

Proof. If $\min(d(i,x),d(i,y)) = d(i,x)$ and $\min(d(j,x),d(j,y)) = d(j,x)$, then $f_l(i) = 1 + d(i,x) + d(i,x) + d(i,y)$ and $f_l(j) = 1 + d(j,x) + d(j,x) + d(j,y)$. Based on the above assumption, We can obtain that:

$$f_l(i) - f_l(j) = 2(d(i,x) - d(j,x)) + (d(i,y) - d(j,y)) = 0$$

Since $A(i,j) = 1$, then $d(i,x) = d(j,x)$ or $|d(i,x) - d(j,x)| = 1$. We also have that $d(i,y) = d(j,y)$ or $|d(i,y) - d(j,y)| = 1$. Because node i,j share the same label $f_l(i) = f_l(j)$, to guarantee $f_l(i) - f_l(j) = 0$ in this situation, we must have that $d(i,x) = d(j,x)$ and $d(i,y) = d(j,y)$. Then the theorem is proved.

The same conclusion can also be easily obtained if $\min(d(i,x),d(i,y)) = d(i,y)$ and $\min(d(j,x),d(j,y)) = d(j,y)$. If $\min(d(i,x),d(i,y)) = d(i,x)$ and $\min(d(j,x),d(j,y)) = d(j,y)$, then $f_l(i) = 1 + d(i,x) + d(i,x) + d(i,y)$ and $f_l(j) = 1 + d(j,y) + d(j,x) + d(j,y)$. Based on the above assumption, We can obtain that:

$$f_l(i) - f_l(j) = (2d(i,x) - d(j,x)) + (d(i,y) - 2d(j,y)) = 0$$

To guarantee $f_l(i) - f_l(j) = 0$ in this situation, we have that $d(i,x) = d(j,y)$ and $d(i,y) = d(j,x)$. Then the theorem is proved.

The same conclusion can also be easily obtained when $\min(d(i,x),d(i,y)) = d(i,y)$ and $\min(d(j,x),d(j,y)) = d(j,x)$

This completes the proof of the theorem. □

B. Proof for Theorem 2

Proof. When aggregating the node pair (i,j) into a new node k, it is equivalent to add all neighbors of node j to neighbors of node i and remove node j from the graph or add all neighbors of node i to neighbors of node j and remove node i from the graph. Therefore, $f_l(k)$ can be derived from $f_l(i)$.

Based on the proof of Theorem 1, we have that $d(i,x) = d(j,x)$ and $d(i,y) = d(j,y)$ if $\min(d(i,x),d(i,y)) = d(i,x)$ and $\min(d(j,x),d(j,y)) = d(j,x)$ or $\min(d(i,x),d(i,y)) = d(i,y)$ and $\min(d(j,x),d(j,y)) = d(j,y)$. In this situation, aggregating node j to node i does not affect the minimal distance for node i to target nodes x and y, since the minimal distance from j to target nodes are the same. We can draw conclusion that:

$$f_l(k) = f_l(i) = 1 + \min(d(i,x),d(i,y)) + d(i,x) + d(i,y).$$

Therefore, we have $f_l(i) - f_l(k) = 0$ that satisfies $0 \leqslant f_l(i) - f_l(k) \leqslant 1$.

Based on the proof of Theorem 1, we have that $d(i,x) = d(j,y)$ and $d(i,y) = d(j,x)$ if $\min(d(i,x),d(i,y)) = d(i,x)$, $\min(d(j,x),d(j,y)) = d(j,y)$, $d(i,x) \neq d(i,y)$ and $d(j,x) \neq d(j,y)$. In this situation, we have that $d(i,y) = d(j,y) + 1$, since if there exists shorter a path from i to j that $d(i,y) < d(j,y)$, it violates $\min(d(i,x),d(i,y)) = d(i,x)$ and $d(i,x) \neq d(i,y)$.

Therefore, we have $f_l(i) - f_l(k) = 1$ that satisfies $0 \leqslant f_l(i) - f_l(k) \leqslant 1$.

This completes the proof of the theorem. □

Acknowledgments

This work was supported in part by National Science Foundation grant DBI-1661289.

References

Adamic, L. A., and Adar, E. 2003. Friends and neighbors on the web. *Social networks* 25(3):211–230.

Ahmed, N. K.; Rossi, R.; Lee, J. B.; Willke, T. L.; Zhou, R.; Kong, X.; and Eldardiry, H. 2018. Learning role-based graph embeddings. *arXiv preprint arXiv:1802.02896*.

Bar-Yossef, Z., and Mashiach, L.-T. 2008. Local approximation of pagerank and reverse pagerank. In *Proceedings of the 17th ACM conference on Information and knowledge management*, 279–288. ACM.

Barabási, A.-L., and Albert, R. 1999. Emergence of scaling in random networks. *science* 286(5439):509–512.

Brin, S., and Page, L. 2012. Reprint of: The anatomy of a large-scale hypertextual web search engine. *Computer networks* 56(18):3825–3833.

Bruna, J.; Zaremba, W.; Szlam, A.; and Lecun, Y. 2014. Spectral networks and locally connected networks on graphs. In *International Conference on Learning Representations (ICLR2014), CBLS, April 2014*.

Chen, H.; Perozzi, B.; Hu, Y.; and Skiena, S. 2018a. Harp: Hierarchical representation learning for networks. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Chen, L.-C.; Papandreou, G.; Kokkinos, I.; Murphy, K.; and Yuille, A. L. 2018b. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence* 40(4):834–848.

Duvenaud, D. K.; Maclaurin, D.; Iparraguirre, J.; Bombarell, R.; Hirzel, T.; Aspuru-Guzik, A.; and Adams, R. P. 2015. Convolutional networks on graphs for learning molecular fingerprints. In *Advances in neural information processing systems*, 2224–2232.

Grover, A., and Leskovec, J. 2016. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, 855–864. ACM.

Jeh, G., and Widom, J. 2002. Simrank: a measure of structural-context similarity. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, 538–543. ACM.

Katz, L. 1953. A new status index derived from sociometric analysis. *Psychometrika* 18(1):39–43.

Kipf, T. N., and Welling, M. 2016a. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Kipf, T. N., and Welling, M. 2016b. Variational graph auto-encoders. *arXiv preprint arXiv:1611.07308*.

Koren, Y.; Bell, R.; and Volinsky, C. 2009. Matrix factorization techniques for recommender systems. *Computer* (8):30–37.

Kovács, I. A.; Luck, K.; Spirohn, K.; Wang, Y.; Pollis, C.; Schlabach, S.; Bian, W.; Kim, D.-K.; Kishore, N.; Hao, T.; et al. 2019. Network-based prediction of protein interactions. *Nature communications* 10(1):1240.

Liben-Nowell, D., and Kleinberg, J. 2007. The link-prediction problem for social networks. *Journal of the American society for information science and technology* 58(7):1019–1031.

Lü, L., and Zhou, T. 2011. Link prediction in complex networks: A survey. *Physica A: statistical mechanics and its applications* 390(6):1150–1170.

Newman, M. E. 2001. The structure of scientific collaboration networks. *Proceedings of the national academy of sciences* 98(2):404–409.

Nickel, M.; Murphy, K.; Tresp, V.; and Gabrilovich, E. 2016. A review of relational machine learning for knowledge graphs. *Proceedings of the IEEE* 104(1):11–33.

Niepert, M.; Ahmed, M.; and Kutzkov, K. 2016. Learning convolutional neural networks for graphs. In *International conference on machine learning*, 2014–2023.

Oyetunde, T.; Zhang, M.; Chen, Y.; Tang, Y.; and Lo, C. 2016. Boostgapfill: improving the fidelity of metabolic network reconstructions through integrated constraint and pattern-based methods. *Bioinformatics* 33(4):608–611.

Perozzi, B.; Al-Rfou, R.; and Skiena, S. 2014. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, 701–710. ACM.

Qiu, J.; Dong, Y.; Ma, H.; Li, J.; Wang, K.; and Tang, J. 2018. Network embedding as matrix factorization: Unifying deepwalk, line, pte, and node2vec. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, 459–467. ACM.

Tang, J.; Qu, M.; Wang, M.; Zhang, M.; Yan, J.; and Mei, Q. 2015. Line: Large-scale information network embedding. In *Proceedings of the 24th international conference on world wide web*, 1067–1077. International World Wide Web Conferences Steering Committee.

Watts, D. J., and Strogatz, S. H. 1998. Collective dynamics of ‘small-world’ networks. *nature* 393(6684):440.

Xu, K.; Li, C.; Tian, Y.; Sonobe, T.; Kawarabayashi, K.-i.; and Jegelka, S. 2018. Representation learning on graphs with jumping knowledge networks. In *International Conference on Machine Learning*, 5449–5458.

Zhang, M., and Chen, Y. 2017. Weisfeiler-lehman neural machine for link prediction. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 575–583. ACM.

Zhang, M., and Chen, Y. 2018. Link prediction based on graph neural networks. In *Advances in Neural Information Processing Systems*, 5165–5175.

Zhang, M.; Cui, Z.; Neumann, M.; and Chen, Y. 2018. An end-to-end deep learning architecture for graph classification. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Zhou, T.; Lü, L.; and Zhang, Y.-C. 2009. Predicting missing links via local information. *The European Physical Journal B* 71(4):623–630.