IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 26 November 2023, accepted 20 December 2023, date of publication 25 December 2023,
date of current version 3 January 2024.

Digital Object Identifier 10.1109/ACCESS.2023.3346688

RESEARCH ARTICLE

A Graph Attention Network-Based Link Prediction Method Using Link Value Estimation

ZHIWEI ZHANG, XIAOYIN WU, GUANGLIANG ZHU, WENBO QIN, AND NANNAN LIANG
School of Informatics and Engineering, Suzhou University, Suzhou 234000, China

Corresponding author: Zhiwei Zhang (zzwloveai@gmail.com)

This work was supported in part by the Natural Science Foundation of Anhui Province under Grant 1908085QF283; in part by the Doctoral Scientific Research Funding under Grant 2019jb08, Grant 2021bsk016, and Grant 2023bsk024; in part by the University Synergy Innovation Program of Anhui Province under Grant GXXT-2022-047; in part by the Natural Science Research Projects in Universities under Grant 2022AH051372 and Grant 2023AH040314; and in part by the Suzhou University Scientific Research Funding under Grant 2021XJPT50, Grant 2022xhx004, and Grant 2022xhx099.

ABSTRACT Link prediction in complex networks is a critical process aimed at uncovering hidden or potential connections among nodes. This technique is widely utilized in areas such as knowledge graphs. Current Graph Neural Networks (GNNs) often focus exclusively on determining whether nodes are connected or assessing the strength of these links by leveraging node attributes. They typically use network structure and attributes to develop node representations through neighborhood aggregation. However, these methods often overlook the intrinsic importance of the links themselves. This paper thoroughly examines the significance of link value based on network structure and introduces an innovative approach for estimating this value, and proposes a method that incorporates link value into both the formulation and training of a link prediction graph attention network. This integration not only boosts the accuracy of link predictions but also provides a theoretical basis for understanding the prediction results. We conducted extensive experiments in link prediction employing widely recognized benchmark datasets. The findings reveal that our proposed framework for link prediction exhibits commendable performance and generalization capabilities, and overall performance improved by an average of 1.2%, thereby establishing it as an effective baseline model.

INDEX TERMS Complex network, graph neural network, link prediction, link value, structure analysis.

I. INTRODUCTION

Link prediction within complex networks aims to reveal hidden or potential connections among nodes, which includes predicting both unexplored links and those likely to emerge in the future. The primary goal of link prediction is to deepen the understanding of network topology, explore the principles of network evolution, functional characteristics, and potential practical significance. Furthermore, link prediction facilitates the anticipation of emerging connections in the network, refining decision-making support [1], [2], [3], [4]. In social networks, link prediction is used to suggest friends, enhance social media platform engagement, and predict events like disease transmission. In protein networks, it helps infer protein interactions, predict gene regulatory relationships, and advance our understanding of protein structure and function. In financial networks, link prediction aids in forecasting trading relationships and cash flow paths between financial entities, contributing to market trend analysis, changes, and risk assessment of various financial institutions. Theoretically, link prediction serves as a crucial tool for analyzing network structure and evolutionary patterns, enhancing understanding of these aspects, and identifying key nodes and clusters. It also encourages interdisciplinary research and collaboration. As a significant interdisciplinary research topic, link prediction attracts

The associate editor coordinating the review of this manuscript and approving it for publication was Dominik Strzalka.

substantial attention from researchers across various fields and is set to play an increasingly vital role in future interdisciplinary studies [5], [6], [7].

However, current approaches for link prediction using GNNs often overlook the diversity of link types or simply apply basic weighting during GNN training, neglecting the inherent value of links. According to network weak ties theory, different links in a network have distinct values due to their varying importance and influence. For instance, in social networks, weak ties are crucial for spreading information and resources, expanding social influence, and significantly impacting societal and economic development. The valuation of links is also crucial for network analysis tasks. For example, links with high edge-betweenness often serve as critical bridges between node groups, influencing boundaries and holding significant value in node set division. Considering these factors, this article focuses on assessing link value and corresponding GNN training that incorporates this value, thereby equipping a GNN for link prediction with a better ability to comprehend network topology characteristics.

The link value can serve as an important input feature for GNNs, aiding in the prediction model's more accurate learning and understanding of network structures. It is also important to note that in social networks, link value often represents the strength of interpersonal relationships. The principle of triangle closure (i.e., if there is a relationship between A and B, and between B and C, then it is very likely that there is also a relationship between A and C) can predict the formation of relationships more accurately when considering link value. Edges with high link values indicate stronger social connections, increasing their likelihood of participating in triangle closures. This, in turn, helps in understanding the clustering characteristics and small-world properties of the network. Therefore, incorporating link value into the development and training of GNNs fulfills a twofold purpose. It not only enhances the effectiveness of the GNN-based link prediction model discussed in this article but also provides a theoretical basis for understanding the predictability of outcomes. This study conducts an in-depth examination of link value derived from network structure. It introduces relevant evaluation metrics and seamlessly incorporates link value into the model. This integration leads to the creation of an innovative framework named the **L**ink **V**alue Based **G**raph **A**ttention **N**eural **N**etwork (LVGANN).

The principal innovations and contributions of this research are outlined below:

- Integrating node betweenness centrality and edge betweenness centrality in a network, a metric for evaluating link value is proposed, which better reflects and describes the different roles and values of different links (connections) in the real world. In social networks, for example, different links represent different levels of interaction and intensity.
- By integrating link value into the construction of link prediction graph attention networks, the representation learning of network nodes is enhanced, enabling not only the aggregation of neighboring nodes' features but also the incorporation of the value of the edges connected to these neighbors. Compared to traditional graph attention networks, the link prediction graph neural network proposed in this paper is capable of learning richer features from the network.
- In real-world open graph benchmark datasets, including mainstream ones like ogbl-ppa, ogbl-collab and ogbl-ddi, a series of detailed link prediction experiments were conducted. By comparing with different graph neural network models, it has been verified that the link prediction model LVGANN proposed in this paper serves as a baseline with good performance.

The structure of this article is organized as follows. Section II provides a detailed review of relevant literature in the field. Section III elaborates on a link value estimation metric and outlines the link prediction architecture that incorporates this value. Section IV describes the experimental setup used to validate the effectiveness of the proposed method, along with the presentation of results and their analysis. Finally, the paper concludes with a summary of key findings and explores potential avenues for future research in the final section.

II. RELATED WORKS

In recent years, link prediction has emerged as a significant area of focus in network science. Consistent with the central theme of this paper, the following sections will analyze and summarize link prediction in networks. Additionally, they will explore network analysis and learning, particularly through the use of GNNs.

A. LINK PREDICTION IN NETWORKS

Conventional heuristic-based methods for link prediction have often required the manual definition of similarity metrics between network nodes, presenting modeling challenges and limited generalizability. In contrast, network representation learning autonomously acquires node representations, supporting the computation of node similarities essential for link prediction. Its goal is to embed network nodes into a lower-dimensional space, transforming high-dimensional sparse feature vectors into compact embeddings in this space. A prominent method in this domain, particularly for link prediction, is network representation learning based on random walks. This approach captures contextual information of network nodes through random walks to learn embedding vectors, which are then used to compute node similarity and facilitate link prediction.

One of the most notable algorithms in this area is DeepWalk, introduced by Bryan et al. [8], which aims to map the structural properties of network nodes to a new vector space. Here, nodes close within the network are similarly proximate in the vector space, achieving this transformation through specific optimization objectives. The feature vectors representing node structural and attribute information are then concatenated for further network data mining tasks.

Grover et al. [9] expanded on DeepWalk with node2vec, introducing Biased-Random Walks to balance the similarity of each random walk between deep and breadth-first search approaches. This model emphasizes community structure and node importance, showing notable performance in network node classification tasks [9].

LINE, introduced by Tang et al. [10], although not ostensibly employing a random walk strategy, is often classified alongside them in literature [11]. This is due to the shared use of probabilistic loss functions in both LINE and DeepWalk, aimed at minimizing the empirical probabilities of connected nodes and their vectorized similarity distances. Both methods also consider first and second-order similarities, optimizing two loss functions akin to random walk strategies. Tang et al. introduced various preprocessing and optimization methods, such as negative sampling, to enhance learning efficacy. In node similarity evaluation, Wang et al.'s SDNE [12] and Cao et al.'s DNGR [13] differ in their approach to defining similarity vectors. SDNE uses node similarity vectors directly as inputs, while DNGR defines similarity based on common paths from random walks. Despite its merits, DNGR faces limitations with network size and incorporating new nodes. The methods discussed have their strengths and limitations: they generally lack robustness, struggle with dynamic graphs, and face challenges in extending to new graphs. Their performance in link prediction evaluation can be inconsistent. Struc2Vec, proposed by Ribeiro et al. [14], focuses on structural homogeneity within networks. It constructs weighted multilayer graphs to generate node context and employs hierarchical measurements for node similarity assessment across different scales, laying a solid foundation for link prediction.

B. NETWORK ANALYSIS AND LEARNING BASED ON GNN

GNNs, as a prominent framework in graph deep learning, employ a neighborhood aggregation strategy. In this approach, the representation of the central node is iteratively learned and enhanced by amalgamating and transforming the representations of its adjacent nodes. These networks play a significant role in network analysis and learning [15], [16], [17]. Among various GNN models, the Graph Convolutional Network (GCN), introduced by Kipf and Welling [18], is particularly noteworthy. It captures graph features through convolutional operations in the spectral domain. This process involves computing edge weights using both the adjacency matrix and the degree matrix, followed by a weighted summation, as shown in equation (1).

$$H^{l+1} = \sigma\left(D^{-1/2}AD^{-1/2}H^{l}W^{l}\right) \tag{1}$$

where, H^{l} and H^{l+1} illustrates representation of the l-th and the $(l+1)$-th layer, respectively. A denotes the adjacency matrix of a network G, while D represents the degree matrix.

However, the Graph Convolutional Network (GCN) faces several limitations: 1) During the aggregation and updating of representations from neighboring nodes, the edge weights remain constant, lacking adaptability; 2) The GCN's use of convolutional aggregation across the entire graph and its reliance on updating gradients using the full graph create inefficiencies and scalability issues, especially in large-scale graphs; 3) GCN layers exhibit a low-pass filtering effect, predominantly capturing low-frequency information. As the number of GCN layers increases, node representations tend to converge, leading to an over-smoothing problem.

To address the issue of static edge weights in the GCN aggregation process, Veličković et al. [19] proposed the Graph Attention Network (GAT), which incorporates an attention mechanism. This model allows for the independent learning of edge weights as shown in equation (2). Once edge weights, denoted as α_{ij}, are determined, the model updates node representations using attention coefficients. This adaptability allows the model to fine-tune its parameters for specific tasks, thereby enhancing its flexibility and effectiveness.

$$\alpha_{ij} = \frac{\exp\left(LeakyReLU\left(\vec{a}^{T}\left[W\vec{h}_i||W\vec{h}_j\right]\right)\right)}{\sum_{k\in\mathcal{N}_i}\exp\left(LeakyReLU\left(\vec{a}^{T}\left[W\vec{h}_i||W\vec{h}_k\right]\right)\right)} \tag{2}$$

where $\vec{a}^{T}$ represents a weight vector, $\vec{h}_i$ indicates the representation vector of node i, W is a linear transformation matrix, $||$ is a concatenate operator, and $LeakyReLU$ denotes a nonlinear activation function.

At its core, the GAT is a model focused on weighted representation aggregation. Although it effectively addresses the issue of static link weights in GCN aggregation, it still depends on the entire graph for aggregation and gradient updates. This reliance can lead to inefficiencies in scenarios involving large-scale graphs or nodes with a high number of neighbors, thus impeding the model's ability to learn graph representations efficiently.

To address these challenges, Hamilton et al. [20] introduced GraphSAGE, an inductive learning approach. This model begins by collecting a subset of neighbors for each target node through random subgraph sampling. Similar to GCN's inward aggregation process, GraphSAGE combines these sampled node representations to form the target node's representation. This method has demonstrated its efficacy, particularly in tasks like graph context prediction.

Furthermore, GraphSAGE not only effectively addresses challenges such as whole graph gradient updates and limited learning efficiency in large-scale graphs, but it also bypasses the requirement of GCNs to access all nodes during training. However, with an increase in the number of GCN layers, issues like gradient vanishing and model over-smoothing become increasingly significant. To mitigate these issues, Li et al. [21] took inspiration from Convolutional Neural Networks (CNNs) and incorporated elements like residual connections, dense connections, and dilated convolutions. As a result, they meticulously developed a 56-layer DeepGCN [21]. This advancement led to a notable improvement of 3.7% in the mean Intersection over Union metric for tasks such as point cloud semantic segmentation, compared to existing methods.

To enhance the learning capabilities of Graph Neural Networks (GNNs), Zhou et al. introduced a hierarchical structure [22]. This structure utilizes micro-pooling to group nodes into predefined clusters, effectively transforming the initial graph into a more condensed form. However, this approach has a potential drawback: the loss of graph topological information, which could result in suboptimal outcomes. Additionally, the process may lead to different graphs producing identical coarse structures, making them indistinguishable after the transformation.

Besides, GNNs have largely overlooked the intrinsic significance of connections within networks. The theory of network weak ties highlights the varied roles different connections or links play. For instance, weak connections are crucial in maintaining network connectivity, ensuring stability in biological systems, and shaping community structure formation. In contrast, strong ties indicate a higher degree of interaction among entities. Therefore, it is clear that various links in a network have differing levels of importance, each contributing to distinct functionalities and roles. Although models like the GAT [19] introduce differentiated link weights via network structure during node representation learning, they fall short in fully assessing and leveraging the inherent value of these connections. This limitation results in a lack of interpretability in link prediction outcomes, especially in certain applications [23].

Hence, the aim of this paper is to tackle the aforementioned issues by introducing a link-centered Graph Neural Network specifically tailored for link prediction. This model is designed to seamlessly incorporate elements like link value, thereby improving both the interpretability and accuracy of the link prediction outcomes.

III. LINK PREDICTION FRAMEWORK INCORPORATING LINK VALUE

This section begins by outlining the foundational concepts related to link prediction and GNNs. It then explores the indicators used for estimating link values and how these are incorporated into the standard graph attention mechanism. Lastly, it provides a summary of the link prediction framework developed on the basis of the graph attention neural network, emphasizing the integration of link values.

A. PRELIMINARIES

Notations: To improve the clarity of descriptions and explanations related to link prediction and GNNs, we have compiled a detailed list of symbols and notations used in this paper. For their comprehensive representation, please refer to Table 1.

Link Prediction: holds significant importance within the realm of graph data mining. This task revolves around the prediction of potentially absent connections in a graph, as well as the projection of novel edges that might emerge in the future. In essence, link prediction strives to ascertain the presence or potential occurrence of links between two nodes. In the context of a given graph $G = (V, E)$, wherein V represents the node set and E symbolizes the link set, and with the all-encompassing link set $U = V \times V$, the objective of link prediction pertains to the anticipation of links between nodes v_i and v_j ($v_i, v_j \in V$). This anticipation is founded upon the known topological attributes and characteristics of G. To formalize, the process of link prediction through the application of a GNN entails the following sequential phases. Initially, E is divided into E^T and E^P, indicating the traing and validation set respectively, while the complementary set $U - E$ functions as the designated test set. It is evident that E^T united with E^P forms the entirety of E, and their intersection remains null. Subsequently, a model grounded in graph attention neural network principles proceeds to acquire node representations founded on E^T, subsequently undergoing validation against E^P to enhance the model's performance. Ultimately, by effecting Hadamard product operations upon the acquired node representations for v_i and v_j, the task of predicting the existence of a link connecting these nodes is effectively accomplished.

TABLE 1. Notations used in this paper.

Notations	Explanations
$G = (V, E)$	G represents a network, V and E indicate the node set and edge set, respectively.
A	$A = \{a_{ij} \mid i \in [0, n), j \in [0, n)\}$, A indicates the adjacent matrix of G, and if there is a link between node i and j, $a_{ij} = 1$, otherwise $a_{ij} = 0$.
V	$V = (v_1, v_2, \cdots, v_n)$, v_i depicts the i-th node in G.
E	$E = (e_1, e_2, \cdots, e_n)$, e_i depicts the i-th link in G.
$\|V\|$	The number of nodes *in* G.
$\|E\|$	The number of edges *in* G.
$N(i)$	The neighbors of node i.
e_{ij}	The e_{ij} between the node v_i and v_j
α_{ij}	The attention coefficient (weight) of e_{ij}.
w_{ij}	Attention coefficient that integrates link value of e_{ij}.
h_i	The representation of node i.
h_i^l	The l-th layer representation of node v_i.
$\vec{a}^T$	$\vec{a}^T$ represents a weight vector.
d	The dimension of node representation.
k	The average degree of nodes in G.
H^l	The l-th layer representation of G.
W	A linear transformation matrix.

Graph Attention Neural Networks are designed to efficiently and effectively learn features from graph data. In graph-structured datasets, nodes frequently establish connections with multiple others. However, not all of these connections are equally significant; some links are more critical than others. GATs employ attention weights to identify and prioritize these more important neighboring nodes. The inclusion of an attention mechanism offers a substantial benefit, especially in link prediction tasks. This advantage stems from the network's enhanced ability to understand and utilize the complex structural nuances inherent in graph data. Such autonomy in feature learning leads to a deeper comprehension of the relationships between nodes, resulting in improved performance in link prediction tasks.

B. LINK VALUE COMPUTATION

Different links within a network play varying roles, each with its own unique functions and values. Drawing from the concepts outlined by Ding and Wang [24], this article employs metrics such as node degree, node betweenness, and edge betweenness to determine the intrinsic value attributed to each link.

In the context of network $G = (V, E)$, the degree k_i of node $v_i \in V$ signifies the count of edges linked to v_i, mathematically expressed as $k_i = \sum_j a_{ij}$; The betweenness $B(v_i)$ of v_i is defined as the proportion of the total number of shortest paths in G that must traverse through v_i, as outlined in equation (3).

$$B(v_i) = \sum_{s \neq i \neq t} \frac{n_{st}^i}{g_{st}} \tag{3}$$

where the variable g_{st} denotes the cumulative count of shortest paths originating from s and terminating at t, whereas n_{st}^i signifies the quantity of shortest paths that traverse through the intermediary node v_i. Node betweenness serves as a metric that not only underscores the significance of a given node but also encapsulates its centrality within the network structure. Analogous to the conceptualization of $B(v_i)$, the determination of edge betweenness for edge $e_{st} = \{(v_s, v_t) | v_s, v_t \in V$ is formulated as delineated in equation (4). This metric offers insight into the centrality of links within the network, shedding light on its pivotal role in facilitating shortest paths between nodes v_s and v_t.

$$B(e_{st}) = \sum_{s \neq t} \frac{n_{st}^e}{g_{st}} \tag{4}$$

where the variable n_{st}^e represents the number of shortest paths starting from node s and ending at node t, passing through the link e_{st}. Edge betweenness quantifies the level of interconnectedness, influence, and importance of a particular link. A higher betweenness value indicates a more significant role in maintaining network cohesion. It is crucial to note that links with the same edge betweenness might vary in importance within networks of different topologies. Given this consideration, this study adopts a comprehensive approach, harnessing k_i, $B(v_i)$ and $B(e_{st})$ as calculation metrics. This approach capitalizes on the balanced influence of two nodes situated at either end of a link. This holistic perspective informs the computation of the link's value, e_{st}, as determined by the value assessment formula presented in equation (5).

$$LV(e_{st}) = B(e_{st}) + \frac{B(v_s) * k_t}{k_s + k_t} + \frac{B(v_t) * k_s}{k_s + k_t} \tag{5}$$

where $LV(e_{st})$ denotes the value attributed to the link e_{st}, while $B(e_{st})$ signifies the edge betweenness of link e_{st}. Furthermore, $B(v_s)$ and $B(v_t)$ pertain to the node betweenness metrics of nodes v_s and v_t respectively, which serve as the termini of the link in question. Subsequent to the derivation of the links value, this study integrates the obtained link value into the establishing process of the link prediction GNN. This fusion is achieved through a harmonious averaging and merging of equation (5) and (2), which confluence yields a novel attention coefficient denoted as w_{ij}, as depicted in equation (6).

$$w_{ij} = 2 * \alpha_{ij} * LV(e_{ij}) \Big/ (\alpha_{ij} + LV(e_{ij})) \tag{6}$$

In the subsequent phase, w_{ij} replaces the attention coefficient in the Graph Attention Network, marking the completion of the training and learning phase for the link prediction GNN. Additionally, existing challenges in training GNNs include issues such as over-smoothing and the vanishing gradient problem. To effectively counter these challenges, this research advocates for the integration of regularization techniques and residual connections into the architecture and training protocol of the link prediction graph attention neural network. A detailed discussion and implementation of these strategies will be thoroughly explored in the following section, with the aim of comprehensively addressing the aforementioned issues.

C. LINK VALUE-INTEGRATED LINK PREDICTION FRAMEWORK

Building upon the original GAT framework, this section introduces enhancements by incorporating link value considerations. Additionally, it merges a node representation normalization approach with a residual connection mechanism to create a comprehensive framework for link prediction, which is visually illustrated in Figure 1.

A key focus of this study is the emphasis on Node Representation Learning, a crucial aspect of understanding complex network structures. Within this framework, attention coefficients are skillfully integrated with link value. This integration is a significant advancement, demonstrated through an improved process for learning node representations within the graph attention network layer, as precisely outlined in Equation (7). The integration not only enhances the accuracy of the representations but also enriches the interpretability of the network's learning process. Subsequent equations provide a detailed methodology for calculating the node representation $h_i^{(l+1)}$ at layer $l+1$, effectively building upon the representation established in the previous layer l. This layered approach allows for a more nuanced understanding of node relationships and their respective importance within the network, thereby optimizing the overall efficacy of the graph attention network.

$$\begin{cases} z_i^{(l)} = W^{(l)} h_i^{(l)} \\ e_{ij}^{(l)} = LeakyReLU(\vec{a}^{(l)} \left(z_i^{(l)} || z_j^{(l)} \right) \\ \alpha_{ij}^{(l)} = \dfrac{\exp\left(e_{ij}^{(l)}\right)}{\sum_{k \in N(i)} \exp\left(e_{ik}^{(l)}\right)} \\ w_{ij}^{(l)} = 2 * \alpha_{ij}^{(l)} * LV(e_{ij}) \Big/ (\alpha_{ij} + LV(e_{ij})) \\ h_i^{(l+1)} = \sigma\left(\sum_{j \in N(i)} w_{ij}^{(l)} z_j^{(l)} \right) \end{cases} \tag{7}$$

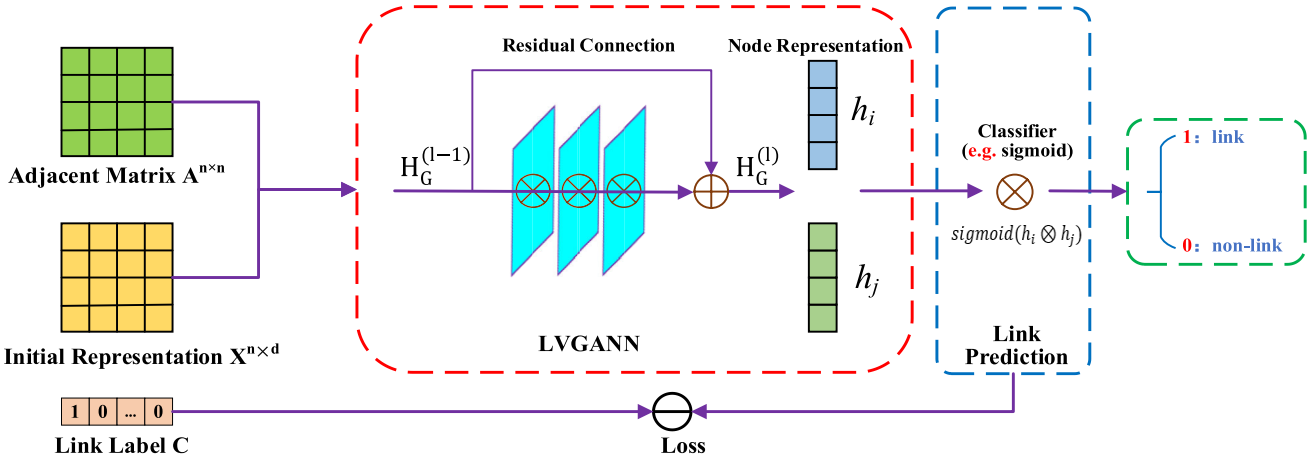

FIGURE 1. Expanded graph attention neural Network for link prediction incorporated link values.

where || indicates the representation concatenation, then $\vec{a}^{(l)}$ denotes a learnable weight vector, and $h_i^{(l+1)}$ represents the freshly acquired node v_i representation.

Node representation normalization is a critical aspect explored in this study. The method for normalizing node representations, detailed in Equation (8), is employed in this paper to address the challenge of over-smoothing. Over-smoothing is a phenomenon that occurs when node representations linked to connections are propagated across multiple layers within the link prediction neural network LVGANN. This issue can lead to homogenization of node features, diminishing the model's ability to capture distinct characteristics and relationships within the network. By implementing normalization techniques, the study aims to maintain the distinctiveness of node representations while leveraging the depth of the network for enhanced learning.

$$NodeNorm\left(h_i^{(l+1)}\right) = \frac{h_i^{(l+1)} - u_i^{(l+1)}}{\sigma_i^{(l+1)}} \tag{8}$$

where $h_i^{(l+1)}$ signifies the representation of v_i at $(l+1)$-th layer, while $u_i^{(l+1)}$ and $\sigma_i^{(l+1)}$ indicate the mean and standard deviation, respectively, of the elements within $h_i^{(l+1)}$.

Residual Connection is a mechanism primarily introduced to address the issue of vanishing gradients within models. In the context of the LVGANN model discussed in this paper, the integration of residual connections aims to enhance the synergistic flow of information across both shallow and deep layers. This integration not only ensures a harmonious alignment of information between these layers but also fosters a more balanced network structure. Such a structure is instrumental in facilitating the training of a more profound LVGANN, thereby enhancing the network's predictive capabilities. In this study, the methodology incorporates residual blocks through skip-layer connections. Consequently, the node representation $h_i^{(l+1)}$ benefits from both node representation normalization and residual connections, as formulated in Equation (9). This dual advantage allows for a robust training process and contributes to the overall efficacy and reliability of the model.

$$h_i^{(l+1)} = ReLU\left(NodeNorm\left(h_i^{(l)}\right)\right) + f(h_i^{(l)}, W^{(l)}) \tag{9}$$

Up to this point, the construction of LVGANN for link prediction has been successfully accomplished within this study. For the purpose of ensuring the reproducibility of the research results in this paper and facilitating readers to further develop on the basis of this study, the pseudocode related to the LVGANN model is summarized as follows.

Following the construction phase of the LVGANN, the next step is Link Prediction. Once the LVGANN is established, a convolution process is applied to both the adjacency matrix of the network and its node representations. For each pair of nodes (v_i, v_j) in the graph G, a binary classifier—specifically, the sigmoid function—is applied to the Hadamard product $\otimes$ of h_i and h_j for nodes v_i and v_j. This procedure is detailed in Equation (10) and is used to predict whether there is a link between v_i and v_j. In this context, a prediction of 1 indicates the existence of a link connecting v_i and v_j, while a prediction of 0 signifies the absence of such a link.

$$\mathcal{P}\left(v_i, v_j\right) = sigmoid\left(h_i \otimes h_j\right) \tag{10}$$

IV. EXPERIMENTS AND DISCUSSION

To conduct a thorough evaluation of the LVGANN model and to facilitate a comparative analysis with other baseline methods, we adhere to the experimental framework outlined in our previous research [1], [25]. Our methodology involves a detailed examination across multiple benchmark datasets to comprehensively assess the model's performance. This section begins by introducing the benchmark datasets utilized, outlining the baseline methods for comparison, and defining the performance evaluation metrics. Following this, we present the experimental results and conduct a comparative analysis. Through this rigorous examination, our objective is to demonstrate the stability and reliability of the LVGANN model.

The PyTorch Pseudocode for LVGANN

Input: a network G, and its adjacent matrix *adj*.
Output: The node representation matrix for link classification.
h is the node representation
e is the edge representation
W is a trainable weight matrix
a is a trainable parameter of attention mechanism.
LV is the value matrix of edges in G.
LV = Apply Equations (3) to (5) to calculate the link value of each edge in G.

```
class LVGANNLayer( ): # define the layer of LVGANN
  def __init__(self, in_feats, out_feats):
  super(LVGANNLayer, self).__init__()
  W=Parameter(torch.empty(size=(in_feats, out_feats)))
  a=Parameter(torch.empty(size=(2∗out_feats, 1)))
  init.xavier_uniform_(W.data, gain=1.414)
  init.xavier_uniform_(a.data, gain=1.414)
  def forward(self, X, adj):
  h = torch.mm(X, self.W)
  N = h.size()[0]
  a=torch.cat([h.repeat(1,N).view(N∗ N,-1), h.repeat(N, 1)],
dim=1).view(N,-1, 2∗out_feats)
  e=leakyrelu(torch.matmul(a, self.a).squeeze(2))
  zero_vec = -9e15∗torch.ones_like(e)
  attention = torch.where(adj > 0, e, zero_vec)
  attention = 2∗a∗attention/(a+attention)
  attention = softmax(attention, dim=1)
  attention = dropout(attention, training=self.training)
  return torch.matmul(attention, h)
class LVGANN():
  def __init__(self, nfeat, nhid, nclass, dropout):
  super(LVGANN, self).__init__()
  self.dropout = dropout
  self.LVGANN1 = LVGANNLayer(nfeat, nhid)
  self.LVGANN2 = LVGANNLayer(nhid, nclass)
  def forward (self, x, adj):
  x = dropout(x,dropout, training=self.training)
  x = elu(self.self.LVGANN1 (x, adj))
  x=dropout(x, dropout, training=self.training)
  x = self.self.LVGANN2(x, adj)
  return log_softmax(x, dim=1)
```

A. SETTINGS

The experimental setup employed in this study is meticulously designed to ensure the accuracy and reliability of the results. The experiments were conducted on a high-performance Workstation Server, specifically a Dell T640. This server is equipped with the CentOS-7-x86_64-1611 operating system, which offers a stable and efficient environment for computational tasks. The GPU used in this setup is the Tesla V100s, known for its robust computational capabilities, particularly in handling large-scale data processing and neural network training.

Furthermore, all experiments of this article were performed using CUDA version 10.2, which provides optimized support for deep learning frameworks and GPU acceleration. The programming environment was standardized using Python 3.7, ensuring compatibility and ease of use for implementing complex algorithms. For the graph neural network implementation, PyTorch version 1.11 was chosen for its dynamic computational graph and extensive library support. Additionally, torch_geometric 2.1 was used, which is a specialized graph deep learning library, constructed and builded on the framework of PyTorch, for deep learning on graphs and provides essential tools for implementing and testing the LVGANN model.

B. DATASETS

The graph datasets employed in this study are sourced from the Open Graph Benchmark (OGB),[1] s detailed in Table 2. These datasets include ogbl-ppa [26], ogbl-collab [27], and ogbl-ddi [28]. To maintain methodological consistency, we adhere to the default partitioning settings prescribed by OGB for splitting these diverse datasets, each characterized by its unique scale and complexity. This approach ensures uniformity in the experimental procedure and enables an accurate assessment of the performance of our methods across different types of graph data.

TABLE 2. Benchmark dataset statistic information.

Dataset	#Nodes	#Edges	Split Type
ogbl-ppa	576,289	30,326,273	Throughput
ogbl-collab	235,868	1,285,465	Time
ogbl-ddi	4,267	1,334,889	Protein target

C. BASELINES

The performance evaluation of the Link Value Based Graph Attention Neural Network (LVGANN) includes a comparative analysis against three established classical graph neural networks: Graph Convolutional Network (GCN) [17], Graph Attention Network (GAT) [18], GraphSAGE [19], and our previous work, EdgeConvNorm [23]. It is crucial to highlight that the use of these GNNs in this study is focused specifically on node representation learning. To support the link prediction task, an additional step involves the application of the same sigmoid classifier to the Hadamard product of the representations of node pairs. This methodology ensures a comprehensive assessment of the LVGANN's capabilities in comparison with these well-known models, providing insights into its effectiveness in the context of link prediction.

- **GCN**: This model was developed to extend the concept of convolutional operations from regular grids, such as those used in image processing, to the domain of irregular graphs. Within the GCN framework, a graph is inputted, and a feature transformation is applied to each node, taking into account the representations of neighboring nodes. Crucially, this shared transformation is key to preserving the inherent local structure of the network. The fundamental principle involves creating a new representation for each node by aggregating features from its neighbors. The GCN is versatile, finding applications in tasks ranging from node clustering to link prediction, demonstrating its adaptability and

[1] https://ogb.stanford.edu/

effectiveness in various graph-based analytical contexts.

- **GraphSAGE**, a variant of the GCN, was developed to tackle the challenge of scalability. It generates node representations through feature sampling and aggregation. Unlike GCN, which processes the entirety of graph nodes during each forward pass, GraphSAGE excels at training on large-scale graphs and generating embeddings for previously unseen nodes. This approach significantly enhances its adaptability and scalability, making it a powerful tool for dealing with extensive and complex graph structures.
- **GAT** first brings attention mechanisms into the realm of GNNs, enabling the differential assignment of significance to distinct nodes. The embedded attention mechanism in GAT is designed to independently evaluate the importance of neighboring nodes, thereby recognizing the varied impacts of different neighbors. This dynamic capability allows nodes to allocate more attention to influential neighbors while giving less attention to those of lesser importance. Consequently, GAT enhances the model's ability to prioritize and learn from the most relevant connections in the graph, leading to more nuanced and effective node representation.
- **EdgeConvNorm** is designed to enhance the process of link representation learning. It accomplishes this by introducing a specialized edge convolution operation, which is adept at capturing the essence of links. Furthermore, the model integrates a normalization strategy to refine the learned link representations. This integration is particularly effective in mitigating the issue of over-smoothing, a common challenge in edge convolution-based link prediction models. A notable aspect of EdgeConvNorm is its implementation in the construction of the link prediction model, where it utilizes stacked edge convolutional layers. This layered approach enables the model to effectively extract and process complex link features, thereby improving the accuracy and robustness of link prediction.

D. PERFORMANCE INDICATOR

In evaluating the effectiveness of link prediction methods, a critical metric is *Hits@n*. *Hits@n* gauges the accuracy of predictions by tallying the number of correctly predicted links within the top n predicted links. Specifically, for each test scenario, all predicted links are ranked according to their predicted probabilities, arranged in descending order. A detailed analysis is conducted to determine whether each correct link appears within the top n ranks of these predictions. If a correct link is found within this range, it is counted as a successful prediction *hit*. The total number of these*hits* is then aggregated across all test cases and normalized by the total number of correctly predicted links. Formally, in this context, assuming there are k correct links, for each link e_{ij}, if it ranks among the top n predicted links, $p_{e_{ij}}$ is assigned a value of 1; otherwise, $p_{e_{ij}}$ is set to 0. Therefore, the calculation of *Hits@n* is executed as per Equation (11), providing a quantifiable measure of the model's predictive accuracy.

$$Hits@n = \left(\frac{1}{k}\right) * \sum p_{e_{ij}} \tag{11}$$

The *Hits@n* metric serves as an indicator of the precision exhibited by link prediction models when forecasting the top n links. An increased *Hits@n* value signifies improved model performance. To evaluate model effectiveness, this study utilizes specific values of n as parameters, namely 10, 50, and 100. It is important to note that each model is subjected to 10 iterations, generating a series of results. The final outcomes are determined by calculating the average *Hits@n*, along with the corresponding standard error, across these multiple iterations. Notably, this metric poses a greater challenge compared to ROC-AUC, as it demands consistent prioritization of predictive accuracy across a wide range of predictions. The implication is that a model must not only be accurate but also consistently rank the most relevant links higher, a task that is inherently more stringent and revealing of a model's predictive capability.

E. EXPERIMENTS AND DISCUSSIONS

The experimental configurations for the baseline methods have been meticulously refined based on the published source code provided by OGB and [23]. In this study, the LVGANN model functions as a comparative baseline, and its parameters are aligned with those of the GAT. These parameters have not been further optimized, as the primary focus is on comparing relative performances among the models.

Throughout our experimentation, we consistently present the most optimal results from each run. The final performance assessment involves computing the mean and standard deviation of the highest results obtained from 10 iterations. Detailed results, which demonstrate the superior comparative performance of all models across various datasets, are presented in Tables 3 to 5.

TABLE 3. The experimental outcomes obtained from various baselines and LVGANN applied to the ogbl-collab.

Metric / Models	Hits@10		Hits@50		Hits@100	
	Val.	Test	Val.	Test	Val.	Test
GCN	0.3273 ±0.0138	0.2657 ±0.0128	0.5036 ±0.0130	0.4264 ±0.0118	0.5830 ±0.0054	0.5024 ±0.0056
GraphSAGE	0.3225 ±0.0111	0.2498 ±0.0115	0.5193 ±0.0065	0.4371 ±0.0061	0.5998 ±0.0037	0.5200 ±0.0035
GAT	0.3729 ±0.0123	0.3054 ±0.0112	0.5671 ±0.0114	0.4735 ±0.0183	0.6372 ±0.0026	0.5510 ±0.0141
EdgeConvNorm	0.3361 ±0.1307	0.3125 ±0.8014	0.5536 ±0.3012	0.4921 ±0.0513	0.5217 ±0.4033	0.4895 ±0.6104
LVGANN	**0.3813 ±0.0046**	**0.3341 ±0.0019**	**0.5841 ±0.0027**	**0.5106 ±0.0134**	**0.6402 ±0.0130**	**0.5723 ±0.0048**

In analyzing the ogbl-collab dataset, we have expanded our approach to encompass temporal considerations, adopting the same data splitting method as presented in OGB. The primary goal is to predict future author collaboration relationships based on historical data, with an emphasis on assigning higher ranks to genuine collaborations. A review of Table 3 clearly shows that the LVGANN developed in this study surpasses other baseline models in performance, and an average accuracy of 0.5038 was achieved. Notably, compared to the GAT, LVGANN and EdgeConvNorm exhibit significant improvements, with a maximum enhancement of 7.8%. Given the dynamic and complex nature of the ogbl-collab, a multi-graph dataset, LVGANN is able to derive deeper insights into node attributes and relational dynamics. This capacity, bolstered by the integration of link value, empowers LVGANN to more effectively predict the existence of a link between two nodes.

For the ogbl-ddi dataset, we utilized a protein-target split method. The primary objective of this approach is to predict drug-drug interactions using information from previously established interactions. Upon analyzing the results in Table 4, it is clear that while the LVGANN shows improved performance compared to the GAT, it does not outperform the GraphSAGE model, but LVGANN still achieved an average accuracy of 0.5760. This difference in performance can be attributed to the specific characteristics of the ogbl-ddi dataset, especially the protein interaction characteristics.

TABLE 4. The experimental results obtained from diverse baselines and LVGANN applied to the ogbl-ddi.

Metric / Models	Hits@10		Hits@50		Hits@100	
	Val.	Test	Val.	Test	Val.	Test
GCN	0.4483 ±0.3710	0.238 6±0.1065	0.6890 ±0.1990	0.6800 ±0.3710	0.7847 ±0.1530	0.7929 ±0.2640
GraphSAGE	**0.6124 ±0.1421**	**0.106 9±0.1362**	**0.7034 ±0.3510**	**0.8572 ±0.0199**	**0.7201 ±0.0021**	**0.9157 ±0.0069**
GAT	0.4716 ±0.0417	0.099 4±0.0407	0.6573 ±0.0071	0.5407 ±0.0913	0.7110 ±0.0231	0.7642 ±0.0471
EdgeConvNorm	0.2307 ±0.4101	0.301 3±0.0027	0.4157 3±0.6037	0.5283 ±0.4051	0.4947 ±0.6129	0.5247 ±0.2104
LVGANN	0.5347 ± 0.0184	0.100 4±0.0601	0.6917 ±0.1094	0.6017 ±0.0134	**0.7238 ±0.1063**	0.8035 ±0.1705

The ogbl-ddi dataset is characterized by its dense network structure, consisting of a total of 4,267 nodes and a remarkable 1,334,889 edges. This inherent density presents significant challenges for predictive models. Nonetheless, the innovative node sampling strategy employed by GraphSAGE proves to be highly effective in this scenario. By strategically selecting nodes during the training process, GraphSAGE adeptly mitigates issues related to the dataset's large-scale graph, the computational demands of full graph gradient updates, and the challenges associated with training efficiency. This approach not only enhances the model's ability to manage the dataset's complexity but also optimizes overall performance, making GraphSAGE particularly suitable for handling densely interconnected networks like the ogbl-ddi dataset.

As a result, the GraphSAGE model stands out as the most suitable choice for link prediction tasks within densely interconnected graph data, such as that found in the ogbl-ddi dataset. The strategic implementation of node sampling techniques endows GraphSAGE with a significant advantage, allowing it to effectively navigate the challenges presented by dense graphs.

Regarding the ogbl-ppa, which represents the most extensive benchmark in this study, we adopt a biological throughput approach for partitioning edges into training, validation, and test sets. This partitioning strategy is in accordance with the standard methodology established in the OGB framework. The primary objective of this approach is to predict specific types of protein relationships, with a focus on physical protein-protein interactions. To facilitate this prediction, the study leverages alternative protein connections. These auxiliary connections provide a simplified yet effective means of assessment and have shown a strong correlation with the targeted interactions, thus enhancing the reliability and accuracy of the predictions in the context of protein relationships.

TABLE 5. The experimental results obtained from diverse baselines and LVGANN applied to ogbl-ppa.

Metric / Models	Hits@10		Hits@50		Hits@100	
	Val.	Test	Val.	Test	Val.	Test
GCN	0.1307 ±0.0081	0.1293 ±0.1034	0.1637 ±0.0171	0.1501 ±0.0308	0.1736 ±0.0170	0.1637 ±0.0043
GraphSAGE	0.1401 ±0.0318	0.1386 ±0.0318	0.1709 ±0.0532	**0.1682 ±0.2041**	0.1803 ±0.0011	**0.1780 ±0.0510**
GAT	**0.1675 ±0.0602**	0.1435 ±0.0032	0.1796 ±0.0062	0.1464 ±0.0107	0.1582 ±0.0143	0.1601 ±0.0712
EdgeConvNorm	0.1183 ±0.0019	0.1037 ±0.0417	0.1206 ±0.3058	0.1282 ±0.0.034	0.1402 ±0.0131	0.1513 ±0.5108
LVGANN	0.1594 ±0.0140	**0.1639 ±0.0216**	**0.1801 ±0.1131**	0.1407 ±0.0025	**0.1812 ±0.0163**	0.1697 ±0.0109

Regrettably, the empirical results presented in Table 5 reveal a significant insight. While the LVGANN shows a slight performance over other three benchmark models, although LVGANN achieved an average accuracy of 0.1658, the overall experimental outcomes for both LVGANN and these models do not meet the anticipated expectations,. This shortfall can be attributed to the vastness and complexity of the data within the ogbl-ppa dataset. Moreover, the intricate dynamics between proteins (nodes) and the nuanced roles of their interconnections remain largely unexplored, highlighting a critical knowledge gap. This gap underscores the

need for innovative approaches in representation learning and aggregation strategies, specifically tailored to capture and elucidate the complex interactions among proteins. Such strategies are essential for forging a path towards more accurate predictive modeling in protein interaction networks.

It is also important to note that LVGANN exhibits increased time complexity compared to its predecessor, the GAT, and other models used in this study. This higher complexity stems from the incorporation of betweenness centrality calculations for link value assessment in LVGANN, which involves a more complex computational process and consequently higher time requirements.

In conclusion, the challenges posed by the extensive dataset of the ogbl-ppa and the intricate nature of protein interactions underline the urgent need for novel strategies. These new approaches must transcend current state-of-the-art methods to achieve a more precise depiction of protein interactions, thereby enhancing the effectiveness of predictive models in this demanding field.

F. TIME COMPLEXITY ANALYSIS

The time complexity of the LVGANN primarily hinges on several key factors: the betweenness centrality of nodes, the betweenness centrality of links, the calculation of link value, the computation of weights and their normalization, and the linear transformation of node representations. In the following contents, we will succinctly analyze and calculate the time complexity of LVGANN, providing a clear understanding of its computational demands in various scenarios.

(1) Node Betweenness Centrality. The computation of node betweenness centrality typically involves determining the shortest paths between all pairs of nodes in a network. To calculate a node's betweenness centrality, it's essential to account for all the shortest paths that traverse through that node. Consequently, the time complexity of computing node betweenness centrality is dependent on the chosen shortest path algorithm and the structure of the network. A commonly used method employs a breadth-first search (BFS) based algorithm for each node. In an unweighted graph G, the time complexity for computing centrality for each node using a BFS-based method is $O(|V| + |E|)$, where $|V|$ is the number of nodes and $|E|$ is the number of edges. Therefore, the overall time complexity for the entire network G is $O(|V|\cdot(|V|+|E|))$, accounting for the BFS computation across all nodes.

(2) Link Betweenness Centrality. The computation of link betweenness centrality involves identifying the shortest paths between all pairs of nodes within a graph. Unlike node betweenness centrality, this measure focuses on the significance of edges, rather than nodes, in these paths. For unweighted graphs, the prevalent method to calculate link betweenness centrality utilizes the breadth-first search (BFS) algorithm. The time complexity for employing BFS to determine the shortest path for each node is $O(|V|+|E|)$, where $|V|$ represents the number of nodes and $|E|$ the number of edges. To ascertain the link betweenness centrality for an entire graph, a BFS search is required for each node. Consequently, this leads to an overall time complexity of $O(|V| \cdot (|V| + |E|))$, reflecting the cumulative computational effort needed to perform BFS for every node in the graph.

(3) Link Value Estimation. Referring to Equation (5), it becomes clear that once the node betweenness centrality and link betweenness centrality have been determined, the time complexity of calculating the link value is contingent upon computing the degrees of the nodes. Typically, for a graph represented by an adjacency matrix, the time complexity of calculating the degrees for all nodes is $O(|V|^2)$, where $|V|$ denotes the number of vertices in the graph. This calculation is crucial as it directly influences the efficiency and feasibility of the link value computation process in large-scale graph structures.

(4) Weight Calculation and Normalization. Based on formulas (7) and (8), it is clear that the weight calculation and normalization in LVGANN involve the application of weight scores and the *softmax* function. The computation of the weight scores involves linear transformation of the node features, and the application of these weight scores to the *softmax* function results in the computation of link weights. The time complexities of these two steps are respectively $O(|V| \cdot d)$ and $O(|V| \cdot k)$, where k indicates the average node degree of G. Moreover, the weight normalization process involves normalizing these weights, with this step having a time complexity of $O(|V| \cdot d)$. Therefore, the overall time complexity of weight calculation and normalization is $O(|V| \cdot (d + k))$.

(5) Node Representation Linear Transformation. For each node, the representation linear transformation involves the weighted summation of neighboring nodes' features using weights, followed by a linear transformation to obtain a new representation of the node. The time complexity of this step is $O(|V| \cdot d)$. It should be noted that the aforementioned weight calculation, normalization, and linear transformation of node representations pertain to one layer of LVGANN. If LVGANN has F layers, then the time complexity needs to be multiplied by F.

In summary, the time complexity of the LVGANN proposed in this paper is $O(2\,|V| \cdot (|V| + |E|) + |V|^2 + F \cdot |V| \cdot (d + k) + F \cdot |V| \cdot d)$. This level of complexity is relatively high, which may lead to suboptimal performance in large-scale networks. However, it is worth noting that we have addressed the issue of LVGANN's high time complexity in a separate research work, exploring potential solutions to optimize its efficiency in handling extensive network datasets.

V. CONCLUSION

This paper introduces significant advancements in the realm of link prediction in network analysis. A novel metric integrating node and edge betweenness centrality is proposed to evaluate link value, offering a more nuanced understanding of the varying roles and values of links in networks, such as the differing levels of interaction in social networks. Additionally, by incorporating this link value into the construction of link prediction graph attention networks, the paper enhances

node representation learning. This approach not only aggregates features of neighboring nodes but also factors in the value of the edges connecting these neighbors, allowing for richer feature learning compared to traditional graph attention networks. Furthermore, the paper conducts extensive link prediction experiments on real-world open graph benchmark datasets, including ogbl-ppa, ogbl-collab, and ogbl-ddi. These experiments showed an average performance improvement of 1.2% overall, and validate the proposed LVGANN as a robust baseline model with commendable performance compared to various graph neural network models.

Through experimentation, it is evident that the LVGANN proposed in this paper performs well on various benchmark datasets, particularly on the real-world datasets ogbl-collab and ogbl-ppa, thereby validating the practicality of LVGANN. Besides, it can be observed from the experimental consequences in Tables 3 to 5 that LVGANN has good performance stability and robustness, without significant fluctuations. Nonetheless, it is imperative to acknowledge that this enhanced performance comes at the cost of an elevated time complexity $O(2\,|V|\cdot(|V|+|E|)+|V|^2+F\cdot|V|\cdot(d+k)+F\cdot|V|\cdot d)$ when compared to other baselines. As a direct consequence of this observation, our forthcoming efforts will be dedicated to targeted refinements and optimizations. These endeavors aim to mitigate the elevated time complexity exhibited by the LVGANN model. By addressing this particular limitation, we seek to strike a balance between predictive prowess and computational efficiency, thus further enhancing the practical utility and applicability of LVGANN in real-world scenarios.

REFERENCES

[1] Z. Zhang, X. Wu, H. Xu, L. Cui, H. Zhang, and W. Qin, "Link value estimation based graph attention network for link prediction in complex networks," in *Proc. 9th Int. Symp. Syst. Secur., Saf., Rel. (ISSSR)*, Hangzhou, China, Jun. 2023, pp. 189–196.
[2] K. Feng, C. Li, X. Zhang, and J. Zhou, "Towards open temporal graph neural networks," in *Proc. ICLR*, 2023, p. 23.
[3] B. Zhang, S. Luo, L. Wang, and D. He, "Rethinking the expressive power of GNNs via graph biconnectivity," in *Proc. ICLR*, 2023, p. 59.
[4] L. Cai, J. Li, J. Wang, and S. Ji, "Line graph neural networks for link prediction," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 9, pp. 5103–5113, Sep. 2022.
[5] B. P. Chamberlain, S. Shirobokov, E. Rossi, F. Frasca, T. Markovich, N. Y. Hammerla, M. M. Bronstein, and M. Hansmire, "Graph neural networks for link prediction with subgraph sketching," in *Proc. ICLR*, 2023, p. 27.
[6] Z. Zhang, P. Cui, and W. Zhu, "Deep learning on graphs: A survey," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 1, pp. 249–270, Jan. 2022.
[7] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu, "A comprehensive survey on graph neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 1, pp. 4–24, Jan. 2021.
[8] B. Perozzi, R. Al-Rfou, and S. Skiena, "DeepWalk: Online learning of social representations," in *Proc. 20th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2014, pp. 701–710.
[9] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in *Proc. KDD*, 2016, pp. 855–864.
[10] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, "LINE: Large-scale information network embedding," in *Proc. 24th Int. Conf. World Wide Web*, May 2015, pp. 1067–1077.
[11] W. L. Hamilton, R. Ying, and J. Leskovec, "Representation learning on graphs: Methods and applications," *IEEE Data Eng. Bull.*, vol. 40, no. 3, pp. 52–74, Sep. 2017.
[12] D. Wang, P. Cui, and W. Zhu, "Structural deep network embedding," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2016, pp. 1225–1234.
[13] S. Cao, W. Lu, and Q. Xu, "Deep neural networks for learning graph representations," in *Proc. AAAI Conf. Artif. Intell.*, 2016, pp. 1145–1152.
[14] L. F. R. Ribeiro, P. H. P. Saverese, and D. R. Figueiredo, "Struc2vec: Learning node representations from structural identity," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2017, pp. 385–394.
[15] M. Zhang, X. Wang, M. Zhu, C. Shi, Z. Zhang, and J. Zhou, "Robust heterogeneous graph neural networks against adversarial attacks," in *Proc. AAAI*, 2022, pp. 4363–4370.
[16] A. Feng, C. You, S. Wang, and L. Tassiulas, "KerGNNs: Interpretable graph neural networks with graph kernels," in *Proc. AAAI*, 2022, pp. 6614–6622.
[17] A. Goodge, B. Hooi, S. K. Ng, and W. S. Ng, "LUNAR: Unifying local outlier detection methods via graph neural networks," in *Proc. AAAI*, 2022, pp. 6737–6745.
[18] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. ICLR*, 2017, p. 14.
[19] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," in *Proc. ICLR*, 2018, p. 12.
[20] W. L. Hamilton, R. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Proc. NIPS*, 2017, p. 19.
[21] G. Li, M. Müller, A. Thabet, and B. Ghanem, "DeepGCNs: Can GCNs go as deep as CNNs?" in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 9267–9276.
[22] K. Zhou, Q. Song, X. Huang, D. Zha, N. Zou, and X. Hu, "Multi-channel graph convolutional networks," in *Proc. IJCAI*, 2020, pp. 1352–1358.
[23] Z. Zhang, X. Wang, C. Guan, Z. Zhang, H. Li, and W. Zhu, "AutoGT: Automated graph transformer architecture search," in *Proc. ICLR*, 2023, p. 14.
[24] J. Ding and J. Wang, "Route importance assessment method based on edge weight and agglomeration coefficient," *Comput. Appl. Softw.*, vol. 38, no. 9, pp. 39–44, 2021.
[25] Z. Zhang, L. Cui, and J. Wu, "Exploring an edge convolution and normalization based approach for link prediction in complex networks," *J. Netw. Comput. Appl.*, vol. 189, Sep. 2021, Art. no. 103113.
[26] D. Szklarczyk, A. L. Gable, D. Lyon, A. Junge, S. Wyder, J. Huerta-Cepas, M. Simonovic, N. T. Doncheva, J. H. Morris, P. Bork, L. J. Jensen, and C. V. Mering, "STRING v11: Protein–protein association networks with increased coverage, supporting functional discovery in genome-wide experimental datasets," *Nucleic Acids Res.*, vol. 47, no. D1, pp. D607–D613, Jan. 2019.
[27] K. Wang, Z. Shen, C. Huang, C. Wu, Y. Dong, and A. Kanakia, "Microsoft academic graph: When experts are not enough," *Quant. Sci. Stud.*, vol. 1, no. 1, pp. 396–413, 2020.
[28] D. S. Wishart et al., "DrugBank 5.0: A major update to the DrugBank database for 2018," *Nucleic Acids Res.*, vol. 46, no. D1, pp. D1074–D1082, 2018.

ZHIWEI ZHANG was born in Fuyang, China, in 1988. He received the B.S. degree in computer science and technology from Suzhou University, Suzhou, China, in 2010, the M.S. degree in computer system architecture from the Kunming University of Science and Technology, Kunming, in 2013, and the Ph.D. degree in computer science and technology from the South China University of Technology, Guangzhou, in 2016. He has been a Lecturer with the Data Science and Big-Data Application Laboratory, School of Informatics and Engineering, Suzhou University, since 2017. He has authored more than 20 articles. His research interests include social computing, social network analysis, complex networks, and big-data applications.

XIAOYIN WU was born in Quanjiao, China, in 1979. She received the B.S. degree in computer and applications from Anhui Agricultural University, in 2001, and the M.S. degree in computer application technology from the Hefei University of Technology, Anhui, in 2008. She has been an Associate Professor with the Department of Computer Science and Technology, School of Informatics and Engineering, Suzhou University, Suzhou, China, since 2001. She has authored more than ten articles. Her research interests include computer hardware technology, the Internet of Things technology, and big-data applications.

GUANGLIANG ZHU was born in Heze, China, in 1974. He received the B.S. degree in electronic instruments and measurement technology from the Changchun University of Science and Technology, Jilin, in 1997, the M.S. degree in military equipment from the PLA University of Science and Technology, Jiangsu, in 2004, and the Ph.D. degree in military operations research from Second Artillery Engineering University, Shaanxi, in 2013. He has been an Associate Professor in computer science and technology, since 2019. He is the author of more than 20 articles. His research interests include network security, the application of artificial intelligence in education, and big data applications.

WENBO QIN was born in Xuchang, China, in 1994. He received the B.S. degree in measurement and control technology and instrumentation from Henan Polytechnic University, in 2015, the M.S. degree in mechanical engineering from Henan Polytechnic University, in 2018, and the Ph.D. degree in detection technology and automation devices from Northeastern University, in 2022. Since 2022, he has been with the School of Information Engineering, Suzhou University. From September 2022 to September 2023, he was an Associate Researcher with the Hefei Comprehensive National Science Center, Institute of Artificial Intelligence. He has published more than ten SCI academic papers. His main research interests include micro-nano detection, intelligent detection, and artificial intelligence.

NANNAN LIANG was born in Suzhou, China, in 1988. She received the B.S. degree in information management and information system from the Anhui University of Technology, Maanshan, in 2009, and the M.S. degree in information management and information system from the Hefei University of Technology, Hefei, in 2012. She has been a Lecturer with the Data Science and Big-Data Application Laboratory, School of Informatics and Engineering, Suzhou University, Suzhou, China, since 2012. She has authored more than ten articles. Her research interests include big data image processing, material mechanics image analysis, and big-data applications.