Pattern Recognition 158 (2025) 111020

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/pr

Collaborative graph neural networks for augmented graphs: A local-to-global perspective

Qihang Guo [a], Xibei Yang [a,b,*], Ming Li [c,d], Yuhua Qian [e]

[a] School of Economics and Management, Jiangsu University of Science and Technology, Zhenjiang, Jiangsu, 212100, China
[b] School of Computer, Jiangsu University of Science and Technology, Zhenjiang, Jiangsu, 212100, China
[c] Zhejiang Institute of Optoelectronics, Jinhua, Zhejiang, 321004, China
[d] Zhejiang Key Laboratory of Intelligent Education Technology and Application, Zhejiang Normal University, Jinhua, Zhejiang, 321004, China
[e] Institute of Big Data Science and Industry, Shanxi University, Taiyuan, Shanxi, 030006, China

ARTICLE INFO

Keywords:
Graph neural networks
Multi-perspective learning
Embedding fusion
Complementary learning

ABSTRACT

In the field of graph neural networks (GNNs) for representation learning, a noteworthy highlight is the potential of embedding fusion architectures for augmented graphs. However, prevalent GNN embedding fusion architectures mainly focus on handling graph combinations from a global perspective, often ignoring their collaboration with the information of local graph combinations. This inherent limitation constrains the ability of the constructed models to handle multiple input graphs, particularly when dealing with noisy input graphs collected from error-prone sources or those resulting from deficiencies in graph augmentation methods. In this paper, we propose an effective and robust embedding fusion architecture from a local-to-global perspective termed collaborative graph neural networks for augmented graphs (LoGo-GNN). Essentially, LoGo-GNN leverages a pairwise graph combination scheme to generate local perspective inputs. Together with the global graph combination, this serves as the basis to generate a local-to-global perspective. Specifically, LoGo-GNN employs a perturbation augmentation strategy to generate multiple augmentation graphs, thereby facilitating collaboration and embedding fusion from a local-to-global perspective through the use of graph combinations. In addition, LoGo-GNN incorporates a novel loss function for learning complementary information between different perspectives. We also conduct theoretical analysis to assess its expressive power under ideal conditions, demonstrating the effectiveness of LoGo-GNN. Our experiments, focusing on node classification and clustering tasks, highlight the superior performance of LoGo-GNN compared to state-of-the-art methods. Additionally, robustness analysis further confirms its effectiveness in addressing uncertainty challenges.

1. Introduction

Graph-structured data, characterized by complex interrelationships and nonlinear properties, has become an essential representation across numerous fields [1]. The proliferation of graph data in various fields such as social networks, bioinformatics, financial risk assessment, and recommendation systems has intensified the need for improved modeling and analysis. Recently, Graph Neural Networks (GNNs) have garnered widespread attention due to their remarkable performance in modeling graph data, which have been shown to rely heavily on information fusion in graph data [2–4]. The embedding fusion architecture, as a widely used information fusion mechanism for processing graph data, is an important variant of GNNs [5–7].

Currently, many representative embedding fusion approaches for augmented graphs have been developed to avoid being constrained by the effective expression ability of the raw input [8–11]. They usually involve two key steps. First, the design of graph augmentation strategies is crucial, aiming to extend the single-view data to multiple inputs. Second, leveraging a fusion architecture enables effective collaboration and learning from the inputs. This naturally raises two questions. (1) *How can effective graph augmentation methods be designed?* (2) *How can a flexible and effective embedding fusion architecture for multi-inputs be designed?*

To address the first question, several effective graph augmentation methods have been proposed recently, such as feature graph augmentation [9,10] and random walk graph augmentation [4,12], which mainly focus on the generation of topology relations. However, although the design of graph augmentation at the topology level is simple, it increases the difficulty of developing subsequent fusion architectures.

* Corresponding author at: School of Computer, Jiangsu University of Science and Technology, Zhenjiang, Jiangsu, 212100, China.
E-mail addresses: 211110701107@stu.just.edu.cn (Q. Guo), jsjxy_yxb@just.edu.cn (X. Yang), mingli@zjnu.edu.cn (M. Li), jinchengqyh@sxu.edu.cn (Y. Qian).

https://doi.org/10.1016/j.patcog.2024.111020
Received 7 April 2024; Received in revised form 3 September 2024; Accepted 11 September 2024
Available online 13 September 2024

This is attributed to the fact that embedding fusion mainly involves fusion of node embeddings, that is, the fusion at the node feature level, which requires additional fusion modules to handle multiple adjacency matrices [1,13,14]. Therefore, it is also necessary to develop graph augmentation methods at the node feature level to facilitate the design of fusion architectures.

To address the second question, most embedding fusion architectures for multi-inputs mainly focus on the fusion and learning of different graph embeddings, such as designing effective attention mechanisms [10] and constructing multi-channel learning [4,9,12]. They treat multiple inputs as a global perspective for processing, which means that the combination of all inputs is treated as a whole and the elements inside are processed in the same form. A common approach is to learn the graph representation for each input, and then fuse these embeddings through attention mechanisms [4,8,10,12]. However, they rarely focus on designing the embedding fusion architecture from the local perspectives of input combinations. Local combination evaluation is very common in classical model fusion strategies such as ensemble learning [15]. Learning only the combination of all inputs from a global perspective, without considering its collaboration with different partial input combinations (local perspectives), may limit the expressive power of the model. Here we provide a simple example that will help to better understand this issue.

As depicted in Fig. 1, we use superpixel graphs [16] of images to represent the input graphs (multiple views, such as View1, View2, and View3, derived from the raw graph and its graph augmentation) to elucidate the advantages of local-to-global perspective learning more intuitively. We refer to the combination of all views as the global perspective and consider the combination of partial views as the local perspective. It can be observed from Fig. 1 that global information and local information can form a complementary relationship, especially when subjected to noise attacks. The global information originates from the common learning process of all views, making it vulnerable to the influence of noise attacks. However, it is not difficult to identify crucial information by simultaneously considering information from different local perspectives. It is worth noting that crucial information may originate not only from global information but also from local information. Therefore, designing suitable input combination schemes and learning complementary information between different perspectives assists in better understanding the system complexity and adapting to various situations more flexibly and comprehensively.

In our study, we propose an effective and robust node embedding fusion architecture called collaborative graph neural networks for augmented graphs (LoGo-GNN) to address the above two issues. LoGo-GNN incorporates information collaboration and node embedding fusion from a local-to-global perspective to enhance the robustness and expressive power of the model. First, we develop a general perturbation augmentation strategy at the node feature level that serves the generation of different inputs. Additionally, we propose a novel pairwise graph combination scheme to generate local perspective inputs. Together with the combination of all input graphs, it serves as the core for generating local and global perspectives, which cohesively apply local and global perspective encoders for embedding fusion. Finally, the node embeddings generated from different perspectives are fused into a unified representation using an attention mechanism. The unified representation feeds into the final objective modules for downstream tasks. It is worth noting that the loss function of LoGo-GNN includes a novel complementary loss and an object loss to guide the efficient node embedding fusion and collaboration between different perspectives. Compared to the existing GNN architectures based on embedding fusion, our approach makes the following contributions.

(1) A novel embedding fusion architecture for augmented graphs is proposed, which extends single-view input through a perturbation augmentation strategy, and constructs a collaborative mechanism from a local-to-global perspective through a pairwise graph combination scheme. In addition, it also includes a novel global loss function to guide effective node embedding fusion and collaboration between different perspectives. We demonstrate the effectiveness of LoGo-GNN both theoretically and experimentally. We also conduct a robustness analysis to demonstrate the good robustness of LoGo-GNN in addressing uncertainty challenges.
(2) A novel perturbation augmentation strategy is developed, which associates node features with diverse topological information, enhancing the quality of node representation. It not only provides a novel scheme for extending a single input to multiple inputs, but also preserves the raw adjacent relationships. Detailed experimental analysis verifies the effectiveness of the perturbation augmentation strategy.
(3) A novel scheme of pairwise graph combination is proposed. It decomposes different input graphs into multiple graph combinations, enabling LoGo-GNN to explore the local perspective information between any pairwise graph combinations.
(4) A novel global loss function is developed. We introduce complementary learning between different perspectives into the embedding fusion architecture through a novel complementary loss function. The complementary loss function and the object loss serving the learning objective representation together form a global loss function to guide the efficient node embedding fusion and collaboration between different perspectives.

2. Related work

Graph neural networks based on embedding fusion architecture have spawned many representative works due to their excellent performance in downstream tasks [17,18]. One of the most noteworthy are the multi-channel learning models for augmented graphs. They share two common characteristics. (1) A single input is expanded into multiple inputs through graph augmentation techniques. (2) The fusion process handles all channels or inputs from a global perspective. Our LoGo-GNN is built on embedding fusion architectures for augmented graphs. Thus, we discuss work that is related to our method from these three perspectives.

Graph augmentation. Graph augmentation is a widely used technique for graph representation learning tasks, which has a significant impact on graph contrastive learning and fusion architecture, as shown in various studies [19–21]. The most effective forms of graph augmentation can be summarized into two categories. (1) Topology-level augmentation involves creating augmented graph instances through global graph sampling [19], edge manipulation [20], building feature graphs [9,10], and adjustments in adjacency matrix weights [22]. Most recent works augment the topologies from the perspective of homogeneity analysis [23], the learning between augmented graphs still belongs to homogeneous learning, and there are still many limitations in designing fusion architectures. (2) Node feature level augmentation mostly employs random feature masking and introducing Gaussian noise to feature vectors [20]. However, these techniques may introduce excessive local variations, disturbing the relational aspects among neighboring nodes. Fortunately, node feature level augmentation exhibits greater effectiveness in many scenarios when designed properly, helping to obtain more discriminative representations due to its precise impact on node features [8]. This leads to the promotion of graph augmentation from traditional learning-only forms to participate in fusion forms.

Multi-channel learning in graph neural networks. The multi-channel learning of graph neural networks can utilize the complementarity between multiple information transmission channels, thereby improving the performance of the model in graph-structure related tasks [1]. Most multi-channel learning models mainly focus on designing attention mechanisms and the fusion of input or channel information from a global perspective for example, fusion architectures for different

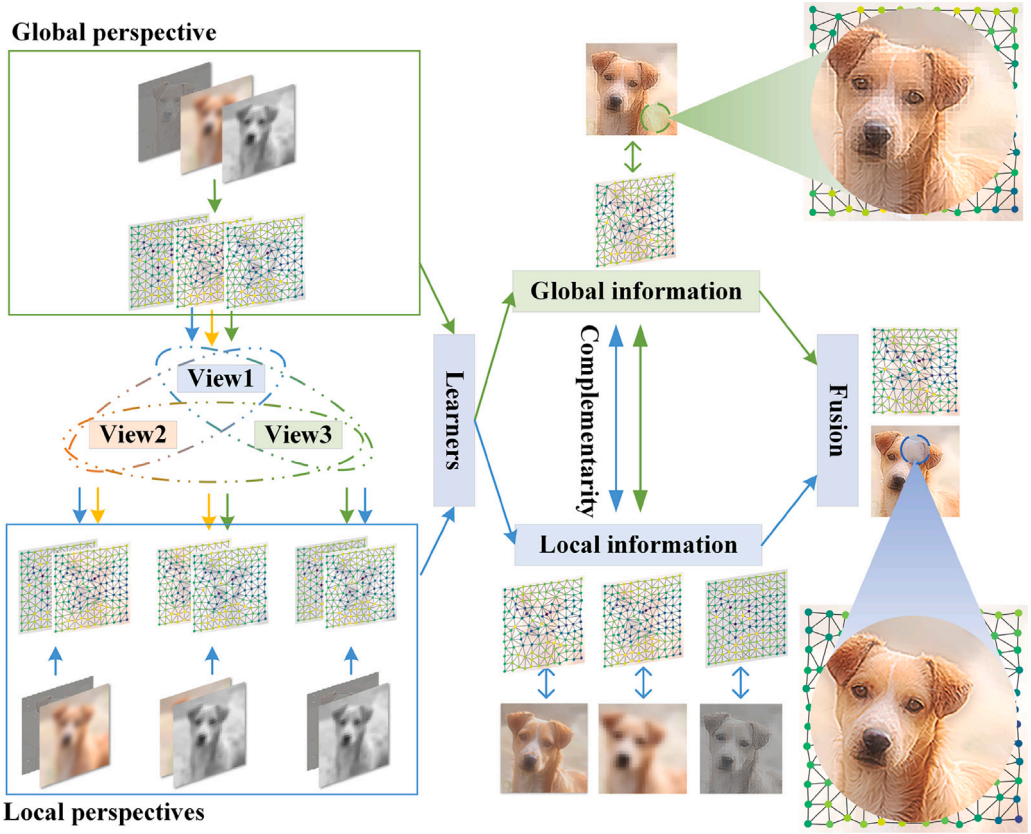


Fig. 1. Illustration of local-to-global perspective collaborative learning. It can be intuitively seen from the images that the flexible use of local and global information will result in clearer images. The graphs of SLIC superpixels are 8-nearest neighbor graphs in the Euclidean space and node colors denote the mean pixel intensities [16]. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

hop neighbor information [4,17,18], fusion architectures for different topology graph information [9,10], and fusion architectures for different augmented node feature information [8]. However, they rarely consider elastic learning mechanisms, such as contrastive relationship learning [20,24], multi-channel learning from different perspectives of input combination, and multi-scale learning which adapts to different environments [25], making it difficult to design robust and flexible fusion architectures. In this paper, our study draws inspiration from these ideas to explore a novel robust architecture of GNNs.

Local-to-global perspective graph learning. The local-to-global perspective graph learning paradigms can simultaneously capture global and local information from graphs, thereby enhancing model performance. Some works try design local and global learning through certain graph structure properties [2,26]. For example, LGD-GCN [26] leverages local and global information to disentangle node representations in the latent space, thereby improving model expressiveness. Similarly, LA-GCN [2] uses local graph augmentation to learn the distribution of neighbor node features conditioned on central node features, and employs these generated features to enhance the GNN's expressiveness. However, few studies consider the local-to-global relationships from different graph embedding fusion perspectives in GNNs. One of the most intuitive ways to explore such relationships from information sources (i.e., input combinations), which is a widely recognized approach in other fields, particularly ensemble learning [15]. Inspired by these works, we attempt to explore local-to-global learning from the perspective of input graph combinations in GNNs.

3. LoGo-GNN

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ denote a graph, where $\mathcal{V} = \{v_1, v_2, \ldots, v_N\}$, $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ represent the node set and the edge set respectively. We denote the feature matrix and the adjacency matrix as $\boldsymbol{X} \in \mathbb{R}^{N \times F}$ and $\boldsymbol{A} \in \{0,1\}^{N \times N}$, where $\boldsymbol{x}_i \in \mathbb{R}^F$ is the feature of v_i, and $\boldsymbol{A}_{i,j} = 1$ iff $(v_i, v_j) \in \mathcal{E}$. A small amount of class information of nodes in $\mathcal{G}$ is given during training in the semi-supervised setting. C denotes the number of classes, $\mathcal{V}_L$ denotes the set of labeled nodes. $\boldsymbol{Y}^L \in \mathbb{R}^{|\mathcal{V}_L| \times C}$ denotes the label indicator matrix. Our objective is to learn a GNN encoder $g(\cdot) \in \mathbb{R}^{N \times C}$ receiving the graph features and structure as input, which produces node embeddings in low dimensionality. We denote $\boldsymbol{H} = g(\cdot)$ as the learned embedding, where $\boldsymbol{h}_i$ is the embedding of node v_i. $\mathcal{T}$ is the perturbation augmentation function. $\mathcal{S}$ is the set of all topologies, $\boldsymbol{S}_i$ represents the ith topology. In Table 1, the notations used in the article are defined.

Table 1
Notation description.

Notation	Description
$\mathcal{G}$	A graph
$\mathcal{V}$	Node set
$\mathcal{E}$	Edge set
$\boldsymbol{X}$	Feature matrix
$\boldsymbol{x}_i$	ith node feature vector
$\boldsymbol{A}$	Adjacency matrix
$\mathcal{V}_L$	Set of labeled nodes
$\boldsymbol{Y}^L$	Label indicator matrix
$g(\cdot)$	GNN encoder
$\boldsymbol{H}$	Learned embedding
$\boldsymbol{h}_i$	ith node embedding
$\mathcal{T}$	Perturbation augmentation function
$\mathcal{S}$	Set of all topologies
$\boldsymbol{S}_i$	ith topology

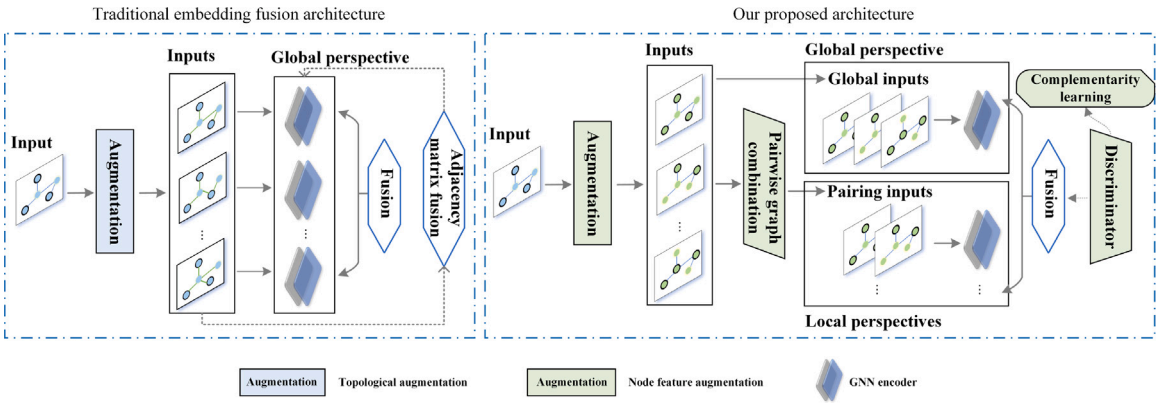


Fig. 2. A comparison between LoGo-GNN and traditional embedding fusion architectures.

3.1. Framework overview

The overall structure of LoGo-GNN is illustrated in Fig. 2. LoGo-GNN differs from traditional embedding fusion architectures in four main aspects (depicted by the green modules in Fig. 2). (1) LoGo-GNN incorporates a novel node feature augmentation (perturbation augmentation) strategy to enhance node representations, expanding a single input into multiple inputs. (2) LoGo-GNN includes a pairwise graph combination scheme to generate inputs from different local perspectives. (3) Each perspective shares a common encoder for multi-perspective representation learning. (4) LoGo-GNN also features a discriminator module, utilizing a newly defined complementary loss function to learn complementary information between different perspectives.

The following sections are structured to present the details and implementation of LoGo-GNN. In Section 3.2, we present the process of perturbation augmentation. In Section 3.3, we provide details on the implementation of the pairwise graph combination approach. In Section 3.4, we illustrate how we leverage GNN encoders for the embedding fusion and learning of multiple perspectives. In Section 3.5, we introduce our proposed global loss function and the process of model training. Finally, Section 3.6 provides details of the theoretical analysis about our model and traditional GNNs.

3.2. Perturbation augmentation

Among of the traditional graph augmentation methods used for fusion architecture, topology augmentation methods are predominant [4,10,18,27]. Introducing diverse topological information can greatly enhance the robustness of GNNs against topological attacks [10,27]. Although these approaches can be seen as a way to extend a single-view input, they mainly focus on the expansion of topological relationships and do not take into account the high-level information hidden between node features and topological relationships. Here we give a simple example to enhance understanding of this issue. In the benchmark dataset Cora for node classification, the node features are composed of word vectors derived from the titles and abstracts of each paper. The existence of edges between papers indicates there is a citation relationship. In situations where two papers exhibit similar word vectors but belong to different categories, identifying their differences becomes challenging. In such cases, uncovering latent high-level semantic information becomes crucial. It is worth noting that the differential information of neighboring node features can potentially provide valuable high-level information to distinguish the nodes of different categories. The high-level information in raw edge (citation) relationships can be considered as the difference between a certain paper's features and the features of the paper it cites. This concept can also be applied to edge relationships with different semantics, leading to the generation of more informative high-level information.

As previously mentioned, a perturbation augmentation strategy is designed, which introduces high-level information related to diverse topology information and node features. This strategy does not alter the topology relationships of the input graph but only improves the quality of node features. It performs graph augmentation from the perspective of node features, i.e., approaching an 'ideal' node feature matrix instead of approaching the 'ideal' topology. In the following, we provide a detailed explanation of the perturbation augmentation.

Given a perturbation augmentation function $\mathcal{T}$, we generate the perturbation augmentation graphs, denoted as:

$$\tilde{\mathcal{G}}_j = \mathcal{T}(\mathcal{G}, \boldsymbol{S}_j) = (\tilde{\boldsymbol{X}}_j, \boldsymbol{A}); \tag{1}$$

Given nodes $\tilde{v}_i \in \tilde{\mathcal{V}}$ $(1 \le i \le N)$, the corresponding feature vector is denoted by $\tilde{\boldsymbol{x}}_i$.

$$\tilde{\boldsymbol{x}}_i = \boldsymbol{x}_i + \sqrt{\sum_{\boldsymbol{x}_j \in \mathcal{N}(\boldsymbol{x}_i)} (\boldsymbol{x}_i - \boldsymbol{x}_j)^2} \Big/ \left|\mathcal{N}(\boldsymbol{x}_i)\right|; \tag{2}$$

where $\mathcal{N}(\boldsymbol{x}_i)$ corresponds to neighborhoods of node v_i based on the topology $\boldsymbol{S}$, $\left|\mathcal{N}(\boldsymbol{x}_i)\right|$ is the number of neighbors of node v_i. We further examine different topologies in the subsequent experimental section. $(\boldsymbol{x}_i - \boldsymbol{x}_j)^2$ represents the difference information (high-level information) between nodes, explaining the implicit high-level relations [28]. We analyze it in detail in the experiment section.

3.3. Pairwise graph combination

It is important to consider both global and local perspectives when dealing with multiple input graphs. To achieve the specific local perspectives, it is necessary to strategically combine these input graphs. Thus, the combination methods play a vital role in this process. GNNs operate by passing messages between multiple pairwise node combinations, thus aggregating information between nodes [29]. We are inspired by this mechanism and design a pairwise graph combination scheme which can be used as the inputs for the local perspective.

Pairwise graph combination. Given a graph set $\{\mathcal{G}_1, \mathcal{G}_2, \ldots, \mathcal{G}_k\}$, the combination of $\mathcal{G}_i$ and $\mathcal{G}_j$ is $\{\mathcal{G}_i, \mathcal{G}_j\}$. $\{\mathcal{G}_1, \mathcal{G}_2, \ldots, \mathcal{G}_k\}$ can be decomposed into $\binom{k}{2}$ pairwise graph combinations. In this paper, the graph combinations $\{\mathcal{G}, \tilde{\mathcal{G}}_1\}$, $\{\mathcal{G}, \tilde{\mathcal{G}}_2\}$, $\ldots$, serve as inputs to the each of the local perspective encoders. The utilization of pairwise graph combination reveals the local perspective information of any two graphs.

3.4. Model details

Our framework enables flexibility in choosing GNN architectures without imposing any constraints. To keep things simple, we adopt the standard single-layer graph convolution network (GCN) [29] as the basis for the embedding fusion, i.e., $g(\cdot) : \mathbb{R}^{N\times F} \times \mathbb{R}^{N\times N} \rightarrow$

$\mathbb{R}^{N\times F}$. For simplicity, the propagation process can be represented as follows: $\sigma(\tilde{\boldsymbol{A}}\boldsymbol{X}\boldsymbol{\Theta})$, where $\sigma(\cdot)$ denotes an activation function, such as $ReLU(\cdot) = \max(0,\cdot)$, $softmax(\boldsymbol{X}_{ij}) = \frac{exp(\boldsymbol{X}_{ij})}{\sum_{j=1}^{C} exp(\boldsymbol{X}_{ij})}$, $sigmoid(\boldsymbol{X}_{ij}) = \frac{1}{1+exp(-\boldsymbol{X}_{ij})}$, $tanh(\boldsymbol{X}_{ij}) = \frac{exp(\boldsymbol{X}_{ij})-exp(-\boldsymbol{X}_{ij})}{exp(\boldsymbol{X}_{ij})+exp(-\boldsymbol{X}_{ij})}$, $\tilde{\boldsymbol{A}} = \hat{\boldsymbol{D}}^{-\frac{1}{2}}\hat{\boldsymbol{A}}\hat{\boldsymbol{D}}^{-\frac{1}{2}}$ is the symmetrically normalized adjacency matrix, $\hat{\boldsymbol{A}} = \boldsymbol{A} + \boldsymbol{I}$, $\boldsymbol{I}$ is the identity matrix, and $\hat{\boldsymbol{D}} = \boldsymbol{D} + \boldsymbol{I}$ is the degree matrix. $\boldsymbol{\Theta} \in \mathbb{R}^{F\times F_h}$ corresponds to the weight matrix of the network, with F representing the dimension of input features and F_h representing the dimension of output features.

Local and global perspective encoders. Each perspective encoder consists of a GCN encoder and an input graph set (pairwise graph combination or the set of all graphs). Specifically, unlike GCN which takes a single graph as input, the local and global perspective encoders take multiple graphs as inputs and obtain embeddings through the mean pool.

$$g_{\boldsymbol{\Theta}_\theta} = \frac{1}{k}\sum_{\mathcal{G}_j\in\{\mathcal{G}_1,\mathcal{G}_2,\ldots,\mathcal{G}_k\}} ReLU(\tilde{\boldsymbol{A}}\boldsymbol{X}_j\boldsymbol{\Theta}_\theta) \tag{3}$$

Assuming the global graph combination is $\{\mathcal{G},\tilde{\mathcal{G}}_1,\tilde{\mathcal{G}}_2\}$, the inputs for the local perspective encoders include $\{\mathcal{G},\tilde{\mathcal{G}}_1\}$, $\{\mathcal{G},\tilde{\mathcal{G}}_2\}$, and $\{\tilde{\mathcal{G}}_1,\tilde{\mathcal{G}}_2\}$, resulting in the corresponding embeddings $\boldsymbol{H}_1 = g_{\boldsymbol{\Theta}_{\theta_1}}(\mathcal{G},\tilde{\mathcal{G}}_1)$, $\boldsymbol{H}_2 = g_{\boldsymbol{\Theta}_{\theta_2}}(\mathcal{G},\tilde{\mathcal{G}}_2)$ and $\boldsymbol{H}_3 = g_{\boldsymbol{\Theta}_{\theta_3}}(\tilde{\mathcal{G}}_1,\tilde{\mathcal{G}}_2)$. As for the global perspective encoder, its inputs consist of $\{\mathcal{G},\tilde{\mathcal{G}}_1,\tilde{\mathcal{G}}_2\}$, leading to the embedding $\boldsymbol{H}_4 = g_{\boldsymbol{\Theta}_{\theta_4}}(\mathcal{G},\tilde{\mathcal{G}}_1,\tilde{\mathcal{G}}_2)$. $\theta_1, \theta_2, \ldots, \theta_m$ represents different perspectives.

Attention mechanism. To efficiently fuse the embeddings generated by the local perspective encoders and the global perspective encoder, we incorporate an attention mechanism [9] to obtain a more comprehensive semantic representation.

With m embeddings, namely $\boldsymbol{H}_1, \boldsymbol{H}_2, \ldots,$ and $\boldsymbol{H}_m$, we use attention mechanisms to determine their respective importance.

$$(\boldsymbol{\alpha}_1,\boldsymbol{\alpha}_2,\ldots,\boldsymbol{\alpha}_m) = att(\boldsymbol{H}_1,\boldsymbol{H}_2,\ldots,\boldsymbol{H}_m) \tag{4}$$

where $\boldsymbol{\alpha}_1,\boldsymbol{\alpha}_2,\ldots,\boldsymbol{\alpha}_m \in \mathbb{R}^{N\times 1}$ indicate the attention values of N nodes with embeddings $\boldsymbol{H}_1,\boldsymbol{H}_2,\ldots,\boldsymbol{H}_m$, respectively. $\alpha_{ij} = softmax(w_{ij})$, $w_{ij} = \mathbf{q}_i^T \cdot tanh(\boldsymbol{\Theta}_{att}\cdot(\boldsymbol{h}_j)^T + \mathbf{b}_i)$, the unnormalized attention value w_{ij} for jth node in embedding matrices $\boldsymbol{H}_i$ through a shared attention vector $\mathbf{q}_i \in \mathbb{R}^{F'\times 1}$, $\boldsymbol{\Theta}_{att} \in \mathbb{R}^{F^h\times F'}$ is the weight matrix, and $\mathbf{b}_i \in \mathbb{R}^{F'\times 1}$ is the bias vector.

Finally, the new embedding $\boldsymbol{H}$ is aggregated by the attention mechanism with the attention values $\boldsymbol{\alpha}$.

$$\boldsymbol{H} = \boldsymbol{\alpha}_1\cdot\boldsymbol{H}_1 + \boldsymbol{\alpha}_2\cdot\boldsymbol{H}_2 + \cdots + \boldsymbol{\alpha}_m\cdot\boldsymbol{H}_m \tag{5}$$

3.4.1. Semi-supervised learning architecture

In semi-supervised learning scenarios, such as node classification tasks, we employ a single-layer GCN layer as the final encoder.

Final encoder. The fused embedding $\boldsymbol{H}$ serves as the input for the final encoder which is a single-layer GCN, resulting in the final output $\tilde{\boldsymbol{Y}}$.

$$\tilde{\boldsymbol{Y}} = softmax(\tilde{\boldsymbol{A}}\boldsymbol{H}\boldsymbol{\Theta}_{final}) \tag{6}$$

3.4.2. Unsupervised learning architecture

In unsupervised learning, the introduction of the decoder effectively captures self-supervised information from input graphs [30]. The LoGo-GNN architecture utilizes multiple perspective encoders to perform embedding fusion. This results in the fused embedding needing to be paired with multiple decoders to reconstruct each input graph. Each input graph corresponds to a decoder, and an additional decoder is added to reconstruct the topology relations of the raw input graph. During decoding, multiple two-layer decoder are utilized, and each decoder layer endeavors to invert its respective encoding process.

Final decoder. The final decoded outputs are reconstructed node attribute matrix $\tilde{\boldsymbol{X}}_i$ and the reconstructed graph structure $\tilde{\boldsymbol{A}}_{re}$. Specifically, the output of the ith input graph decoder is

$$\tilde{\boldsymbol{X}}_i^{re} = softmax(\tilde{\boldsymbol{A}}ReLU(\tilde{\boldsymbol{A}}\boldsymbol{H}\boldsymbol{\Theta}_{\theta_i'}^{(0)})\boldsymbol{\Theta}_{\theta_i'}^{(1)}) \tag{7}$$

where $\boldsymbol{\Theta}_{\theta_i'}^{(l)}$ represents the learning parameters of the ith reconstructed perspective decoder in the lth layer.

The output of the topology reconstruct decoder is

$$\tilde{\boldsymbol{H}} = \tilde{\boldsymbol{A}}ReLU(\tilde{\boldsymbol{A}}\boldsymbol{H}\boldsymbol{\Theta}_{\theta'}^{(0)})\boldsymbol{\Theta}_{\theta'}^{(1)} \tag{8}$$

$$\tilde{\boldsymbol{A}}^{re} = sigmoid(\tilde{\boldsymbol{H}}\tilde{\boldsymbol{H}}^T) \tag{9}$$

3.5. Training

The training objective function of LoGo-GNN comprises two components: complementary learning and object loss. The objective of the complementary learning ($\mathcal{L}_c$) is to guide the fusion of complementary information between different encoder perspectives, whereas the object loss ($\mathcal{L}_o$) instructs the model to perform downstream tasks effectively.

We define a global loss function denoted as $\mathcal{L}$ to facilitate the end-to-end training of LoGo-GNN.

$$\mathcal{L} = \eta\cdot\mathcal{L}_c + (1-\eta)\cdot\mathcal{L}_o \tag{10}$$

Discriminator and complementary loss. In LoGo-GNN architecture, the global encoder perspective struggles to learn the 'ideal' output. To address this issue, the local encoder perspective is introduced as a compensatory measure. In the given context, it is essential to learn complementary information from the aspect of different perspective encoders. We can use the idea of graph contrastive learning [23,24] to learn the relationships between different perspectives. However, the traditional approaches of contrastive learning focus on learning the consistency of node embeddings from various viewpoints, which is not in line with the objective of learning complementary information. To address this issue, we develop a novel complementary loss function (denoted as $\mathcal{L}_c$) for LoGo-GNN.

Projection Head. We follow a learning architecture that is similar to the traditional graph contrastive learning method [21,31]. Specifically, the learned node embeddings $\boldsymbol{H}_j$ are fed into a shared projection head, which comprises a multilayer perceptron (MLP) with two hidden layers and the ReLU activation function. The resulting node embeddings, $\{\boldsymbol{Z}_1,\boldsymbol{Z}_2,\ldots,\boldsymbol{Z}_m\}$, are used to construct the discriminator.

Discriminator. A complementary objective, i.e., a discriminator, is designed to maximize the complementary information between different encoder perspectives. The working of this function is illustrated in Eq. (11).

$$\mathcal{L}_c = exp(-\frac{1}{|\mathcal{V}|}\sum_{l\in\mathbf{V}}\sum_{i=1}^{m-1}\sum_{j\neq i}^{m}\left\|\boldsymbol{Z}_{l,i}-\boldsymbol{Z}_{l,j}\right\|_2) \tag{11}$$

where $|\mathcal{V}|$ is the number of nodes. $\boldsymbol{Z}_{l,i}$ $(1 < i \leq m)$ are the representation feature vectors of the lth node for the ith perspective encoder.

Object loss for semi-supervised learning. We adopt the cross-entropy error as the loss function of the final labeled node embeddings for semi-supervised node classification tasks, denoted as $\mathcal{L}_o$:

$$\mathcal{L}_o = -\sum_{v_k\in\mathcal{V}_L}\sum_{j}^{C}\boldsymbol{Y}_{kj}^L\log(\tilde{\boldsymbol{Y}}_{kj}) \tag{12}$$

Object loss for unsupervised learning. We establish a reconstruction loss to assess the reconstructed graph from decoding and incorporate the contrastive loss function $\mathcal{J}$ of GCA [20] to enhance the self-supervised learning of the model. Due to space constraints, the intricate details of $\mathcal{J}$ will not be expounded upon. The ultimate object function $\mathcal{L}_o$ is defined as follows:

$$\begin{aligned}\mathcal{L}_o = &\sum_{j\in\{\mathcal{G}_1,\mathcal{G}_2,\ldots,\mathcal{G}_k\}}\left\|\tilde{\boldsymbol{X}}_j-\tilde{\boldsymbol{X}}_j^{re}\right\|_2\\ &+\lambda_1\left\|\tilde{\boldsymbol{A}}-\tilde{\boldsymbol{A}}^{re}\right\|_2+\lambda_2\sum_{i}^{m}\sum_{j\in\mathcal{G}set_i}\sum_{k\in\mathcal{G}set_i,j\neq k}\mathcal{J}_{j,k}^i\end{aligned} \tag{13}$$

where λ_1 and λ_2 are hyperparameters. $\mathcal{G}set_i$ is the set of input graphs for the ith perspective encoder.

Remark on Computational Complexity. The feature transformation computational complexity of LoGo-GNN training is $\mathcal{O}\big((k+\binom{k}{2})|\mathcal{E}|(FD+LD^2)\big)$, where D is the number of hidden channels, k is the number of input graphs, and L is the number of layers. In contrast, the complexity of GCN [29], GAT [5], as some base GNN models, are given by a feature transformation computational complexity $\mathcal{O}\big(|\mathcal{E}|(FD+LD^2)\big)$. The computational complexity of LoGo-GNN heavily relies on the number of input graphs. When k is 1, LoGo-GNN shares an identical computational complexity with base models.

3.6. Theoretical analysis

We investigate why our proposed method outperforms the existing GNN model by characterizing the expressive power of these variants, with our theoretical analysis mainly based on Ref. [10]. Assuming the existence of an infinite graph $\bar{\mathcal{G}}$ with a node set $\bar{\mathcal{V}}$ and a probability space with measure $\tilde{P}$, a formal function analysis views the given graph $\mathcal{G}$ as a subgraph of $\bar{\mathcal{G}}$. The nodes in $\mathcal{G}$ are considered independent and identically distributed samples of $\bar{\mathcal{V}}$ based on $\tilde{P}$. Let $\mathcal{V}\subset\bar{\mathcal{V}}$ be a node set and $f:\mathcal{V}\to\mathbb{R}$ be a bounded and continuous function defined on it. Each node in $\mathcal{V}$ has m features. Let g be a bounded, differentiable function on $\mathbb{R}$ satisfying $|g(x)|\le 1$ and $\int_K g'(t)dt<\infty$ where K is a compact set. We denote $\boldsymbol{A}_{\mu,v}:=\boldsymbol{A}(\mu,v)$ as the kernel associated with the (μ,v) element of a given matrix $\boldsymbol{A}$. $\hat{x}_\mu$ and $\hat{x}_v$ stand for the 'idea' feature vectors related to $\boldsymbol{A}_{\mu,v}$ of node μ and node v, respectively. x_μ and x_v stand for the input feature vectors of node μ and node v, respectively. The unknown function on $\bar{\mathcal{G}}$ can be represented through an integral representation framework.

$$f\left(\hat{x}_v\right)=\int_{R^m}\alpha(w)g\left(w^{\mathrm{T}}\int_V \boldsymbol{A}_{\mu,v}\hat{x}_\mu d\tilde{P}(\mu)\right)dw.$$

Likewise, this integral can be approximated with the Monte-Carlo method to get an approximation defined as

$$\hat{f}_L^*\left(\hat{x}_v\right):=\sum_{j=1}^{L}\beta_j g\left(w_j^{\mathrm{T}}\int_{\mathcal{V}}\boldsymbol{A}_{\mu,v}\hat{x}_\mu d\tilde{P}(\mu)\right).$$

where $\beta_j=\frac{T}{L}\operatorname{sign}\left[\alpha\left(w_j\right)\right]$ and the $\{w_1,w_2,\ldots,w_L\}$ are independently chosen subject to the probability distribution P^*, which is unknown in advance. As such, all the parameters $\{\beta_1,\beta_2,\ldots,\beta_L\}$ and $\{w_1,w_2,\ldots,w_L\}$ need to be optimized algorithmically (with the given training samples), as the philosophy of BP training. Denote the distance between f and $\hat{f}_L^*$ by

$$d_{\mathcal{V}}\left(f,\hat{f}_L^*\right):=\sqrt{\frac{1}{|\mathcal{V}|}}E\left[\int_{\mathcal{V}}\left(f\left(\hat{x}_v\right)-\hat{f}_L^*\left(\hat{x}_v\right)\right)^2 d\tilde{P}(v)\right].$$

where $E[\cdot]$ denotes the expectation value with respect to the probability distribution $\tilde{P}$, and $|\mathcal{V}|$ denotes the volume of the node set $\mathcal{V}$.

Theorem 1. *For a graph signal $\mathcal{G}$ with an ideal latent complete node feature matrix set $\mathcal{X}_M$ and given the bounded loss function $\mathcal{L}$, the expected object loss of GNN with the node feature matrix set $\mathcal{X}_M$ and its subset $\mathcal{X}_m$ are expressed as $\mathcal{L}(\mathcal{X}_M;\boldsymbol{A}_{\mathcal{G}})$ and $\mathcal{L}(\mathcal{X}_m;\boldsymbol{A}_{\mathcal{G}})$, respectively. Then, the difference between the $\mathcal{L}(\mathcal{X}_M;\boldsymbol{A}_{\mathcal{G}})$ and $\mathcal{L}(\mathcal{X}_m;\boldsymbol{A}_{\mathcal{G}})$ is bounded by:*

$$\left|\mathcal{L}\left(\mathcal{X}_M;\boldsymbol{A}_{\mathcal{G}}\right)-\mathcal{L}\left(\mathcal{X}_m;\boldsymbol{A}_{\mathcal{G}}\right)\right|\le\sqrt{I\left(\left(\mathcal{X}_M\backslash\mathcal{X}_m\right)\mid\mathcal{X}_m;\boldsymbol{A}_{\mathcal{G}}\right)}.$$

and the difference decreases with the increase of m.

Proof. Suppose $\mathcal{X}_M=(\mathcal{X}_m,\mathcal{X}_{M\backslash m})$, where $\mathcal{X}_{M\backslash m}$ represents the additional features in $\mathcal{X}_M$ not in $\mathcal{X}_m$. Using the chain rule:

$$I(\mathcal{X}_M;\mathcal{A}_{\mathcal{G}})=I(\mathcal{X}_m,\mathcal{X}_{M\backslash m};\mathcal{A}_{\mathcal{G}})=I(\mathcal{X}_m;\mathcal{A}_{\mathcal{G}})+I(\mathcal{X}_{M\backslash m};\mathcal{A}_{\mathcal{G}}\mid\mathcal{X}_m)$$

The difference can be written as:

$$|\mathcal{L}(\mathcal{X}_M;\mathcal{A}_{\mathcal{G}})-\mathcal{L}(\mathcal{X}_m;\mathcal{A}_{\mathcal{G}})|\le I(\mathcal{X}_{M\backslash m};\mathcal{A}_{\mathcal{G}}\mid\mathcal{X}_m)$$

Applying the square root bound:

$$I(\mathcal{X}_{M\backslash m};\mathcal{A}_{\mathcal{G}}\mid\mathcal{X}_m)\le\sqrt{I((\mathcal{X}_M\setminus\mathcal{X}_m)\mid\mathcal{X}_m;\mathcal{A}_{\mathcal{G}})}$$

Thus, we get:

$$|\mathcal{L}(\mathcal{X}_M;\mathcal{A}_{\mathcal{G}})-\mathcal{L}(\mathcal{X}_m;\mathcal{A}_{\mathcal{G}})|\le\sqrt{I((\mathcal{X}_M\setminus\mathcal{X}_m)\mid\mathcal{X}_m;\mathcal{A}_{\mathcal{G}})}$$

Taking the GCN models as an example, the expressive power of GCN models with sufficiently large L trained through gradient descent learning algorithms is limited in probability based on Monte Carlo methods and the existing literature [10,32], i.e., $d_V\left(f,\hat{f}_L^*\right)\le\frac{C}{\sqrt{L}}$, where $C=\int_{R^m}|\alpha(w)|dw<+\infty$. $d_V\left(f,\hat{f}_L^*\right)$ represents the distance between the ideal model f and GCN models with sufficiently large L trained through gradient descent learning algorithms $\hat{f}_L^*$.

In practical implementations, the existing GCN model uses the augmented node feature matrix x as an approximation for the 'ideal' augmented node feature matrix $\hat{x}$. According to Ref. [10], by evaluating the integral $\int_{\mathcal{V}}\boldsymbol{A}_{\mu,v}\hat{x}_\mu d\tilde{P}(\mu)$ in a Monte Carlo manner through uniform sampling in the node space, the widely used GCN model can be expressed concisely.

$$\hat{f}_L^{GCN}\left(x_v\right):=\sum_{j=1}^{L}\beta_j g\left(w_j^{\mathrm{T}}\frac{1}{N}\sum_{i=1}^{N}\boldsymbol{A}_{\mu_i,v}x_{\mu_i}\right).$$

In the same manner, the global perspective encoder can be formulated as

$$\hat{f}_L^{Global}\left(x_v;x_v^2;x_v^3\right):=\sum_{j=1}^{L}\beta_j g(\alpha_1 w_{j,1}^{\mathrm{T}}\frac{1}{N}\sum_{i=1}^{N}\boldsymbol{A}_{\mu_i,v}x_{\mu_i}+\alpha_2 w_{j,1}^{\mathrm{T}}\frac{1}{N}\sum_{i=1}^{N}\boldsymbol{A}_{\mu_i,v}x_{\mu_i}^2+\alpha_3 w_{j,1}^{\mathrm{T}}\frac{1}{N}\sum_{i=1}^{N}\boldsymbol{A}_{\mu_i,v}x_{\mu_i}^3).$$

where average pooling is used to obtain the final output for multiple outputs from each perspective in this paper. Therefore, $\alpha_1=\alpha_2=\alpha_3=\frac{1}{3}$.

Our model includes a global perspective encoder and multiple local perspective encoders, which can be formulated as

$$\begin{aligned}\hat{f}_L^{LOGO-GNN}\left(x_v;x_v^2;x_v^3\right):=&\sum_{j=1}^{L}\beta_j g(\alpha_1' w_{j,1}^{\mathrm{T}}\frac{1}{N}\sum_{i=1}^{N}\boldsymbol{A}_{\mu_i,v}x_{\mu_i}\\&+\alpha_2 w_{j,1}^{\mathrm{T}}\frac{1}{N}\sum_{i=1}^{N}\boldsymbol{A}_{\mu_i,v}x_{\mu_i}^2\\&+\alpha_3 w_{j,1}^{\mathrm{T}}\frac{1}{N}\sum_{i=1}^{N}\boldsymbol{A}_{\mu_i,v}x_{\mu_i}^3)\\&+\alpha_2' w_{j,2}^{\mathrm{T}}(\alpha_1^2\frac{1}{N}\sum_{i=1}^{N}\boldsymbol{A}_{\mu_i,v}x_{\mu_i}+\alpha_2^2\frac{1}{N}\sum_{i=1}^{N}\boldsymbol{A}_{\mu_i,v}x_{\mu_i}^2)\\&+\cdots\\&+\alpha_4' w_{j,4}^{\mathrm{T}}(\alpha_1^4\frac{1}{N}\sum_{i=1}^{N}\boldsymbol{A}_{\mu_i,v}x_{\mu_i}^2+\alpha_4^4\frac{1}{N}\sum_{i=1}^{N}\boldsymbol{A}_{\mu_i,v}x_{\mu_i}^3))\end{aligned}$$

where $\{x,x^2,x^3\}$ stands for the raw node feature matrix and the augmented node feature matrices. The node feature matrix set for the inputs of the global perspective encoder is $\{x,x^2,x^3\}$. $\{x,x^2\}$, $\{x,x^3\}$, $\{x^2,x^3\}$ are the node feature matrix set for the inputs of the local perspective encoder. $\alpha_1=\alpha_2=\alpha_3=\frac{1}{3}$, $\alpha_1^i=\alpha_2^i=\frac{1}{2}$, and α_1', α_2', α_3', α_4' are the attention coefficients to be learned.

Based on Theorem 1, it is logical to assume that GCN has perspective insufficiency measured by l_1 which partially implies the differences between x and $\hat{x}$, the local perspective encoder by l_2 that partially implies the differences between $\{x,x^2\}$ or $\{x,x^3\}$ or $\{x^2,x^3\}$ and $\hat{x}$, the global perspective encoder by l_3 which partially implies the differences between $\{x,x^2,x^3\}$ and $\hat{x}$, with $l_1,l_2\ge l_3$.

Theorem 2. *To simplify the analysis, we only focus on the node feature matrix set associated with the raw input. In the context of real implementations in discrete cases, the expressive powers for GCN, global perspective*

encoder and our model, respectively, can be bounded by

$$d_{\mathcal{V}}\left(f,\hat{f}_L^{GCN}\right) \le \frac{C}{\sqrt{L}} + \|\beta\|\,\|w\|\,\left|g'(0)\right| \mathcal{O}\left(l_1\right),$$

$$d_{\mathcal{V}}\left(f,\hat{f}_L^{Global}\right) \le \frac{C}{\sqrt{L}} + \|\beta\|\,\|w\|\,\left|g'(0)\right| \mathcal{O}\left(\frac{1}{3}l_1+\frac{1}{3}l_2+\frac{1}{3}l_3\right),$$

$$d_{\mathcal{V}}\left(f,\hat{f}_L^{LOGO-GNN}\right) \le \frac{C}{\sqrt{L}} + \|\beta\|\,\|w\|\,\left|g'(0)\right| \mathcal{O} \left((\frac{1}{3}\alpha_1'+\frac{1}{2}\alpha_2'+\frac{1}{2}\alpha_3'+\frac{1}{2}\alpha_4')(l_1+l_2)+\frac{1}{3}\alpha_1' l_3\right).$$

it indicates that LoGo-*GNN can perform more favorably than GCN with certain appropriate attention coefficients under ideal conditions, that is, approaching the theoretical approximation error* $d_{\mathcal{V}}\left(f,\hat{f}_L^{*}\right)$ *with higher probability. In addition, due to the additional attention coefficient and local perspective encoders of* LoGo-*GNN, its robustness and flexibility are stronger than the global perspective encoder.*

Proof. Due to $l_2 \ge l_3$, it is logical to suppose that the insufficient of different local perspective encoders are uniformly represented as l_2. It roughly admits that

$$\|\frac{1}{N}\sum_{i=1}^{N}\boldsymbol{A}_{\mu_i,v}x_{\mu_i} - \int_V \boldsymbol{A}_{\mu,v}\hat{x}_\mu d\tilde{P}(\mu)\| \propto \mathcal{O}\left(l_1\right)$$

$$\|\alpha_1\frac{1}{N}\sum_{i=1}^{N}\boldsymbol{A}_{\mu_i,v}x_{\mu_i} + \alpha_2\frac{1}{N}\sum_{i=1}^{N}\boldsymbol{A}_{\mu_i,v}x_{\mu_i}^2 + \alpha_3\frac{1}{N}$$

$$\sum_{i=1}^{N}\boldsymbol{A}_{\mu_i,v}x_{\mu_i}^3 - \int_V \boldsymbol{A}_{\mu,v}\hat{x}_\mu d\tilde{P}(\mu)\| \propto \mathcal{O}\left(\frac{1}{3}l_1+\frac{1}{3}l_2+\frac{1}{3}l_3\right)$$

$$\begin{aligned}
&\| \alpha_1'(\alpha_1\frac{1}{N}\sum_{i=1}^{N}\boldsymbol{A}_{\mu_i,v}x_{\mu_i} + \alpha_2\frac{1}{N}\sum_{i=1}^{N}\boldsymbol{A}_{\mu_i,v}x_{\mu_i}^2 + \alpha_3\frac{1}{N}\sum_{i=1}^{N}\boldsymbol{A}_{\mu_i,v}x_{\mu_i}^3)\\
&+ \alpha_2'(\alpha_1^2\frac{1}{N}\sum_{i=1}^{N}\boldsymbol{A}_{\mu_i,v}x_{\mu_i} + \alpha_2^2\frac{1}{N}\sum_{i=1}^{N}\boldsymbol{A}_{\mu_i,v}x_{\mu_i}^2)\\
&+ \cdots\\
&+ \alpha_4'(\alpha_1^4\frac{1}{N}\sum_{i=1}^{N}\boldsymbol{A}_{\mu_i,v}x_{\mu_i}^2 + \alpha_4^4\frac{1}{N}\sum_{i=1}^{N}\boldsymbol{A}_{\mu_i,v}x_{\mu_i}^3) - \int_V \boldsymbol{A}_{\mu,v}\hat{x}_\mu d\tilde{P}(\mu)\,\|\\
&\propto \mathcal{O}\left((\frac{1}{3}\alpha_1'+\frac{1}{2}\alpha_2'+\frac{1}{2}\alpha_3'+\frac{1}{2}\alpha_4')(l_1+l_2)+\frac{1}{3}\alpha_1' l_3\right)
\end{aligned}$$

The values of α_i' are related to the attention coefficient. Based on triangular inequality, we have $d_{\mathcal{V}}\left(f,\hat{f}_L^{GCN}\right) \le d_{\mathcal{V}}\left(f,\hat{f}_L^{*}\right) + d_{\mathcal{V}}\left(\hat{f}_L^{*},\hat{f}_L^{GCN}\right)$. After a careful deduction based on Taylor series for the activation $g(\cdot)$, the latter can be bounded by $\|\beta\|\,\|w\|\,\left|g'(0)\right| \mathcal{O}\left(l_1\right)$, completing the proof of $d_{\mathcal{V}}\left(f,\hat{f}_L^{GCN}\right)$, where β represents the discrete form of the compressed function, $g'(0)$ represents the derivative of the activation function $g(\cdot)$ when the value is 0. Similar tricks can be used to prove $d_{\mathcal{V}}\left(f,\hat{f}_L^{Global}\right)$ and $d_{\mathcal{V}}\left(f,\hat{f}_L^{LOGO-GNN}\right)$.

We analyze in detail the advantages of our model compared to the aforementioned mainstream GCN models. We summarize from Theorem 1 and Theorem 2 that the expressive power of GNN under ideal conditions is primarily determined by the set of augmented feature matrices. Additionally, the expressive power of embedding fusion architectures for augmented graphs is superior to GCN under ideal conditions.

However, it is often challenging to achieve an ideal condition where complete information is available. In such cases, complementary collaboration can provide valuable information when local and global perspectives are combined. This suggests that LoGo-GNN has an advantage over the global perspective encoder. We provide a detailed analysis of this advantage in the experiment section.

4. Experiments

In this section, we describe the experiments that are conducted to evaluate LoGo-GNN and answer the following research questions.

Q1: How does our method perform in node and graph classification tasks?

Q2: What role does each module play in the proposed method?

Q3: How is the robustness of LoGo-GNN reflected?

Q4: How does our method perform in node clustering tasks?

Q5: How does perturbation augmentation work in LoGo-GNN?

Q6: How do different hyperparameters affect LoGo-GNN?

Datasets. The experiments are conducted over six real-world datasets which are summarized in Table 2. Cora, Citeseer, Pubmed, and DBLP are the research paper citation networks [29,33]. ACM is extracted from the ACM dataset [34], and Chameleon is extracted from the Wikipedia network [35]. The super-pixels datasets test graph classification using the popular MNIST and CIFAR10 image classification datasets [16].

LOGO-GNN. In the experiment section, we use **OURS** to represent the LoGo-GNN based on semi-supervised learning, and **OURS-UN** to represent the LoGo-GNN based on unsupervised learning. It is worth noting that the learned embeddings based on unsupervised learning are used for training linear classifiers to obtain the classification results on node classification tasks.

Baselines. We compare LoGo-GNN with state-of-the-art methods. (1) Base encoder: GCN [29]. (2) Attention-based encoder: GAT [5], MAGCN [10], DGCN [12] and PA-GCN [8]. (3) Multi-scale/view information fusion-based encoder: MixHop [17], N-GCN [4], MOGCN [18], MAGCN [10], DGCN [12] and PA-GCN [8]. (4) Contrastive learning-based encoder: NCLA [22], PA-GCN [8], GraphCL [31], IGCL [36], and GCA [20]. (5) Unsupervised learning model: K-means, Deepwalk [37], GAE [30], and VGAE [30].

Parameter Setting. All dataset-specific hyperparameters are summarized in Table 3. The raw input graph is denoted as Raw. It is worth noting that we use multiple graph relations to perform perturbation augmentation as the input graphs (denoted as PR, PC, PK), which are described as follows:

(1) PR: The first graph relations (topology) are provided by the raw graph relations of these datasets.
(2) PC: The second graph relations (topology) are provided by cosine similarity. Cosine similarity is utilized to measure text similarity. Specifically, there is an edge between two nodes if the text similarity is greater than the similarity threshold (hyperparameter *cos*-threshold).
(3) PK: The third graph relations (topology) are provided by kNN graph construction (hyperparameter k) [38].

Implementation Details. We train our model using the full batch in each training epoch. We implement our algorithm in Pytorch and optimize it with the Adam [39] algorithm. In traditional graph data, we select a different number of labeled nodes per class for the training set and choose 1000 nodes as the test set. In super-pixels datasets, we choose 10000 images as the test set. We report the mean classification accuracy (ACC) of our method after 10 runs over the dataset splits that are specified above. The hyperparameter η is selected in the search range {0.05; 0.1; 0.15; ... ; 0.95}, hyperparameter λ_1 and λ_2 are selected in the search range {0; 0.1; 0.2; ... ; 1}. Additionally, the hyperparameter *cos*-threshold is selected in the search range {0.1; 0.15; 0.15; ... ; 0.5}, and k is selected in the search range {5; 10; 15; ... ; 30}. The number of iterations for Eq. (1) is set to {1, 2, 3}. The

Table 2
Data details.

Datasets	Nodes	Edges	Features	Classes	Training	Test
Cora	2708	5429	1433	7	140	1000
Citeseer	3327	4732	3703	6	120	1000
ACM	3025	13 128	1870	3	60	1000
Chameleon	2277	36 101	2325	4	80	1000
DBLP	17 716	105 734	1639	4	80	1000
Pubmed	19 717	44 338	500	3	60	1000
MNIST	–	–	–	10	55 000	10 000
CIFAR10	–	–	–	10	45 000	10 000

Table 3
Hyperparameter settings.

Dataset	Learning rates	Weight decay	Training epochs	Dropout
Cora	0.005	1.00E−05	100	0.5
Citeseer	0.005	1.00E−05	100	0.5
ACM	0.005	1.00E−05	100	0.5
Chameleon	0.005	1.00E−05	100	0.5
DBLP	0.001	1.00E−05	500	0.5
Pubmed	0.001	1.00E−05	1000	0.5
MNIST	0.001	1.00E−05	5000	0.5
CIFAR10	0.001	1.00E−05	5000	0.5

Table 4
Notation description.

Notation	Description
Raw	The raw input graph
RawC	The convolution augmentation graph via Eq. (14).
PR	The augmentation graph based on raw graph relations.
PC	The augmentation graph based on cosine similarity graph relations.
PK	The augmentation graph based on kNN graph relations.
Global	Learned representation of global perspective ({RAW,PC,PK}).
Local1	Learned representation of local perspective ({RAW,PC}).
Local2	Learned representation of local perspective ({RAW,PK}).
Local3	Learned representation of local perspective ({PC,PK}).
Local-Global	Fused representation from the local-to-global perspective.

experiment results of each experiment are obtained with the optimal hyperparameters and the number of iterations.

Evaluation Metrics. Following existing works in evaluating node or graph classification [16,29,40], we adopt classification accuracy (ACC) to evaluate the performance of baselines and LoGo-GNN for node classification tasks. ACC is computed on all testing examples based on each dataset. Additionally, we adopt normalized mutual information (NMI) and adjusted random index (ARI) to evaluate the performance of baselines and LoGo-GNN for node clustering tasks.

4.1. Performance on node and graph classification (Q1)

4.1.1. Performance on node classification

This subsection reports the mean and standard error of classification accuracy (ACC) after 10 runs. It is worth pointing out that the results of DeepWalk [37] and NCLA [22] are obtained from corresponding papers. The semi-supervised node classification results are reported in Table 5. We make the following observations:

(1) Compared to baselines, LoGo-GNN has superior performance over most datasets. It is worth noting that LoGo-GNN based on semi-supervised learning architecture (**OURS**) is superior to LoGo-GNN based on unsupervised learning architecture (**OURS-UN**). This can be attributed to the fact that LoGo-GNN based on a semi-supervised learning architecture is an end-to-end fusion framework, where label information can be used to guide the embedding fusion process in the process of training.
(2) LoGo-GNN based on semi-supervised learning architecture (**OURS**) still has advantages over other models (MAGCN, MOGCN, PA-GCN, etc.) that use multi-topologies/views information fusion. In addition, LoGo-GNN based on unsupervised learning architecture (**OURS-UN**) is superior to most graph contrastive learning models (GCA, IGCL, etc.). This superiority can be attributed to the fact that LoGo-GNN integrates the information from global to local encoder perspectives (from global input graphs to multiple pair graph combinations) and conducts suitable topology learning on fused node embeddings.
(3) Following the failure cases shown in Table 5, taking the Pubmed dataset as an example, the performance of LoGo-GNN is weaker than that of NCLA. This can be attributed to the fact that NCLA includes an augmentation learning mechanism, which can efficiently learn self-supervised information between input graph and augmentation instances when raw graph relations are reliable. Similarly, taking the Citeseer dataset as an example, the performance of LoGo-GNN is also weaker than that of LA-GCN. LA-GCN uses learnable local feature augmentation based on the raw graph relationships and focuses more on enhancing information from feature perspectives. However, overall, the performance of LoGo-GNN is still superior to these methods. The difference is that LoGo-GNN focuses on the design of fusion architecture and uses multiple graph relation information instead of just raw graph relation information to generate augmentation instances due to the possibility of introducing unreliable edge relationships. The advantages of LoGo-GNN are more evident in noisy environments. We analyze the robustness of LoGo-GNN in the subsequent experiment section.

4.1.2. Performance on graph classification

Following Ref. [16], we use the image classification datasets for graph classification and take GCN [29] and GAT [5] as baselines. The mean and standard error of classification accuracy (ACC) are reported after 10 runs. The graph classification results are reported in Table 6. It is clear that LoGo-GNN has superior performance compared to baselines. This reflects that LoGo-GNN is equally competitive in graph classification tasks.

4.2. Ablation study (Q2)

4.2.1. LoGo-GNN architecture study

The superiority of the proposed LoGo-GNN has been verified by the aforementioned comparison experiments. We also perform the following ablation studies to verify the validity of each component in LoGo-GNN in this subsection:

- Multi perspectives-GCN+MLP: GCN with local-to-global encoder perspectives via a standard MLP for node classification tasks;
- Multi perspectives-GAT+GAT: Graph attention networks (GAT) with local-to-global encoder perspectives via a single-layer GAT for node classification tasks;
- Global-GCN+GCN: GCN with global encoder perspective via a single-layer GCN for node classification tasks;
- **OURS-UN**-w/o: LoGo-GNN based on unsupervised learning architecture without constraint $\mathscr{L}_c$;
- **OURS**-w/o: LoGo-GNN based on semi-supervised learning architecture without constraint $\mathscr{L}_c$;
- **OURS-UN**: the proposed LoGo-GNN based on unsupervised learning architecture;
- **OURS**: the proposed LoGo-GNN based on semi-supervised learning architecture;

The ablation results for the six datasets are listed in Table 7. We then make the following observations:

Table 5
ACC of node classification tasks (%), where $\boldsymbol{A}$ and $\boldsymbol{X}$ are the adjacency matrix and feature matrix, and $\boldsymbol{Y}$ is the label information. (Bold: best)

Method	Training data	Datasets					
		Cora	Citeseer	ACM	Chameleon	DBLP	Pubmed
DeepWalk [37]	$\boldsymbol{A}$	67.2	43.2	–	–	–	65.3
NCLA [22]	$\boldsymbol{X}$, $\boldsymbol{A}$	82.2	71.7	–	–	–	**82.0**
GCA [20]	$\boldsymbol{X}$, $\boldsymbol{A}$	82.7 ± 0.6	71.8 ± 0.7	89.4 ± 0.6	35.6 ± 0.2	70.6 ± 0.5	76.8 ± 0.5
IGCL [36]	$\boldsymbol{X}$, $\boldsymbol{A}$	83.5 ± 0.5	72.1 ± 0.6	89.2 ± 0.7	47.6 ± 0.4	73.6 ± 0.3	80.8 ± 0.6
GraphCL [31]	$\boldsymbol{X}$, $\boldsymbol{A}$	83.1 ± 0.6	72.1 ± 0.6	90.2 ± 0.6	48.6 ± 0.5	72.9 ± 0.5	80.6 ± 0.6
OURS-UN	$\boldsymbol{X}$, $\boldsymbol{A}$	83.5 ± 0.4	73.0 ± 0.3	91.6 ± 0.7	51.5 ± 0.6	73.8 ± 0.3	80.9 ± 0.8
GCN [29]	$\boldsymbol{X}$, $\boldsymbol{A}$, $\boldsymbol{Y}$	81.5 ± 0.2	70.4 ± 0.4	87.8 ± 0.2	47.6 ± 0.4	70.2 ± 0.5	79.0 ± 0.6
GAT [5]	$\boldsymbol{X}$, $\boldsymbol{A}$, $\boldsymbol{Y}$	83.2 ± 0.7	72.6 ± 0.7	87.4 ± 0.3	47.9 ± 0.4	71.0 ± 0.3	79.0 ± 0.6
PA-GCN [8]	$\boldsymbol{X}$, $\boldsymbol{A}$, $\boldsymbol{Y}$	83.6 ± 0.2	70.4 ± 0.3	90.9 ± 0.3	49.0 ± 0.3	72.0 ± 0.4	79.3 ± 0.3
MOGCN [18]	$\boldsymbol{X}$, $\boldsymbol{A}$, $\boldsymbol{Y}$	82.4 ± 1.2	72.4 ± 0.8	90.1 ± 1.4	46.9 ± 0.4	70.9 ± 0.7	79.2 ± 0.3
N-GCN [4]	$\boldsymbol{X}$, $\boldsymbol{A}$, $\boldsymbol{Y}$	83.0 ± 0.5	72.2 ± 0.5	88.0 ± 0.3	48.9 ± 0.4	71.3 ± 0.2	79.5 ± 0.4
MixHop [17]	$\boldsymbol{X}$, $\boldsymbol{A}$, $\boldsymbol{Y}$	81.9 ± 0.4	71.4 ± 0.6	87.9 ± 0.7	40.6 ± 0.6	70.9 ± 0.3	80.8 ± 0.2
DGCN [12]	$\boldsymbol{X}$, $\boldsymbol{A}$, $\boldsymbol{Y}$	84.1 ± 0.3	73.3 ± 0.1	90.2 ± 0.2	48.9 ± 0.4	72.3 ± 0.2	80.2 ± 0.3
MAGCN [10]	$\boldsymbol{X}$, $\boldsymbol{A}$, $\boldsymbol{Y}$	84.5 ± 0.5	73.3 ± 0.3	90.6 ± 0.3	50.4 ± 0.5	72.5 ± 0.3	80.6 ± 0.8
LA-GCN [2]	$\boldsymbol{X}$, $\boldsymbol{A}$, $\boldsymbol{Y}$	84.6 ± 0.7	**74.7 ± 0.5**	89.9 ± 0.4	48.3 ± 0.6	72.0 ± 0.5	81.7 ± 0.7
OURS	$\boldsymbol{X}$, $\boldsymbol{A}$, $\boldsymbol{Y}$	**84.6 ± 0.4**	73.4 ± 0.5	**91.8 ± 0.5**	**52.7 ± 0.3**	**74.9 ± 0.5**	81.6 ± 0.7

Table 6
ACC of graph classification tasks (%). (Bold: best)

Method	Datasets	
	MNIST	CIFAR10
MLP	95.3 ± 0.1	56.3 ± 0.2
GCN [29]	90.1 ± 0.1	54.1 ± 0.4
GAT [5]	95.5 ± 0.2	64.2 ± 0.5
OURS-UN	97.5 ± 0.2	67.3 ± 0.4
OURS	**97.8** ± 0.3	**68.1** ± 0.5

Table 7
ACC (%) of LoGo-GNN and its variants. (Bold: best)

Method	Datasets					
	Cora	Citeseer	ACM	Chameleon	DBLP	Pubmed
Multi perspectives-GCN+MLP	84.2	72.4	90.1	49.5	72.1	80.3
Multi perspectives-GAT+GAT	**84.7**	73.3	**92.0**	**52.9**	**75.2**	**81.6**
Global-GCN+GCN	84.5	73.2	91.6	51.7	74.5	81.2
OURS-UN-w/o	82.5	71.5	90.4	51.2	72.8	80.5
OURS-w/o	84.3	73.3	91.5	51.3	73.1	80.9
OURS-UN	83.5	71.7	91.6	51.5	73.8	80.9
OURS	84.6	**73.4**	91.8	52.7	74.9	**81.6**

(1) We apply a standard MLP (Multi perspectives-GCN+MLP) or Graph Attention Networks (multi perspectives-GAT+GAT) for final classification to further verify the advantage of the LoGo-GNN architecture. The results show that the model that introduces graph attention networks (GAT) achieves a better performance than the other models (multi perspectives-GCN+MLP, **OURS**). The introduction of graph attention networks (GAT) is an effective way to learn the topology of fused node embeddings. Additionally, the performance of **OURS** is better than multi perspectives-GCN+MLP. We explain this empirical finding by the fact that fused node embeddings do not significantly influence their raw graph relations. Interestingly, the process of perturbation augmentation introduces different topological information, which makes performing convolution operations based on the raw graph relations on the fused node embeddings effective.

(2) LoGo-GNN achieves a better performance than Global-GCN+GCN. This result highlights the importance of local information when global information is limited or incomplete (label information is limited). Specifically, in most cases, models that introduce both local and global perspectives perform better than models that only introduce the global encoder perspective.

(3) It is important to learn complementary information from various perspectives. We can see from Table 7 that the LoGo-GNN (**OURS-UN** and **OURS**) trained using the global loss function outperforms the LoGo-GNN trained only using the object loss $\mathcal{L}_o$. This also confirms our analysis in designing the global loss function.

4.2.2. Perturbation augmentation study (Q4)

We only use the best performing LoGo-GNN based on semi-supervised learning architecture (**OURS**) for the following experiment to simplify the analysis and conduct the tasks of node classification and semantic similarity analysis on the Cora dataset to further validate the effectiveness of proposed perturbation augmentation strategy.

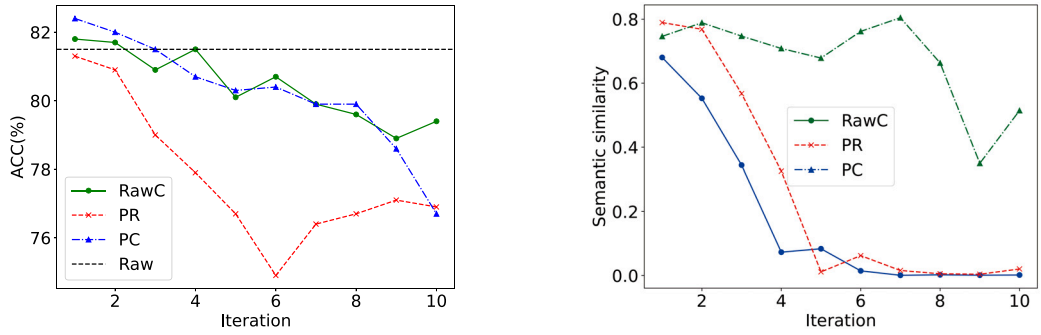
First, we use different aggregation methods to obtain the augmentation graphs. We also introduce the aggregation method of GCN [29] into graph augmentation as a comparison (please refer to Eq. (14)). The descriptions of different input graphs are shown in Table 4. Additionally, we use the broken line chart to describe the representation ability of input graphs (PR, PC, RawC, Raw) after each aggregation (iteration) and compare the semantic similarity between the augmentation graphs and the input graph Raw. Specifically, each augmentation graph and input graph learn their representations through a shared GCN encoder, and their semantic similarity is measured by comparing the learned graph representations. The results are shown in Fig. 3. It can be observed that performing multiple iterations of perturbation augmentation can significantly change the semantic similarity with the raw graph. In addition, perturbation augmentation graphs based on other topological relationships have better expressive power in the case of small iterations. It is worth noting that the semantic similarity between the augmentation graphs and the raw graph is smaller when using other topologies for perturbation augmentation. Based on the analysis above, an augmentation graph with a significant representation ability and small semantic similarity is more effective in providing supplementary information during the information fusion process. Therefore, we are more inclined to choose the augmentation graphs based on other topology relations.

$$\tilde{\boldsymbol{x}}_i = \hat{\boldsymbol{A}}_{i,i}\boldsymbol{x}_i + \sum_{\boldsymbol{x}_j \in \mathcal{N}(\boldsymbol{x}_i)} \hat{\boldsymbol{A}}_{i,j}\boldsymbol{x}_j. \tag{14}$$

We also compare the performance of the GCN encoder based on the raw graph (Raw) and different perturbation augmentation graphs (PR, PC, PK), as shown in Table 8 and Table 9. It is clear that the GCN encoder based on different perturbation augmentation graphs performs better than the GCN encoder based on the raw graph, which further proves the effectiveness of our proposed perturbation augmentation strategy.

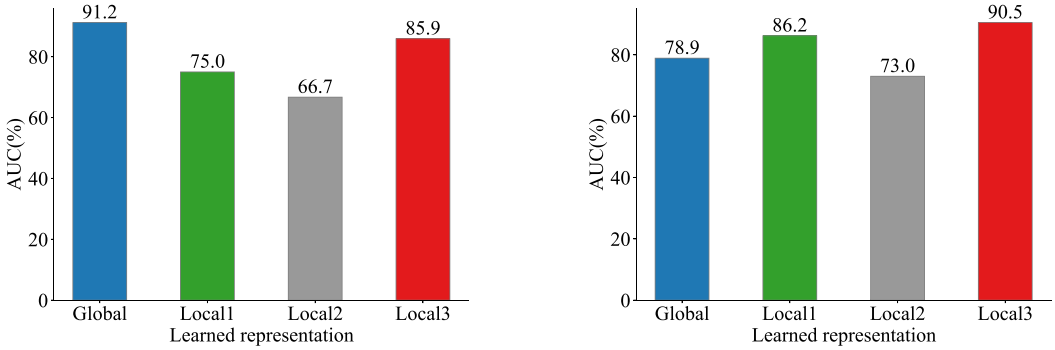
4.2.3. Robustness analysis (Q3)

As LoGo-GNN based on unsupervised learning architecture (**OURS-UN**) and LoGo-GNN based on semi-supervised learning architecture (**OURS**) have the same fusion architecture, we only use the best



(a) Classification accuracy (ACC) (b) Semantic similarity

Fig. 3. Representation ability of input graphs and semantic similarity between the perturbation augmentation graphs and the raw input graph on the Cora dataset.



(a) AUC of unattacked inputs (b) AUC of attacked inputs

Fig. 4. The performance of VGAE reconstructed graphs for the learned embeddings based on different perspectives on the Cora dataset. Unattacked inputs represent using representations learned from each unattacked perspective as inputs. Attacked inputs represent using representations learned from each attacked perspective as inputs, where we randomly delete some edges with an 80% ratio to get the modified graph as the input of each perspective encoder.

Table 8
ACC (%) of GCN encoder based on different perturbation augmentation graphs. (Bold: best)

Input	Datasets					
	Cora	Citeseer	ACM	Chameleon	DBLP	Pubmed
RAW	81.5	70.4	87.8	47.6	70.2	79.0
PR	81.3	70.5	88.1	46.9	71.2	78.3
PC	**82.4**	**71.4**	**88.7**	**49.6**	**71.8**	80.1
PK	82.2	70.7	87.5	48.3	70.0	**80.5**

Table 9
ACC (%) of LoGo-GNN based on different graph combinations. (Bold: best)

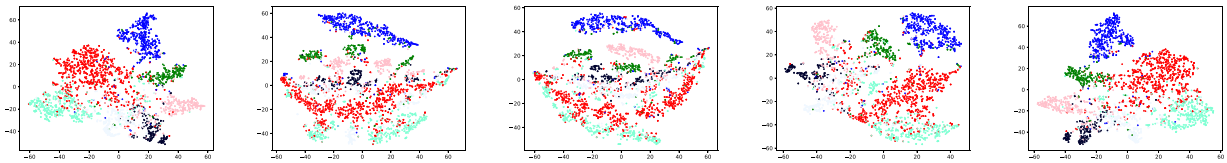
Input	Datasets					
	Cora	Citeseer	ACM	Chameleon	DBLP	Pubmed
Raw+PR+PC	83.7	71.5	91.0	50.6	72.9	80.4
Raw+PR+PK	83.3	71.1	91.2	50.2	71.9	81.0
Raw+PC+PK	**84.6**	**73.4**	**91.8**	**52.7**	**74.9**	**81.6**

performing LoGo-GNN based on semi-supervised learning architecture (**OURS**) for the following experiment to simplify the analysis.

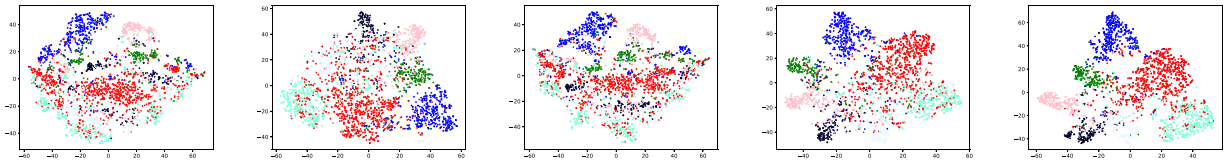
First, we use visualization and graph reconstruction methods to measure the expressive power of each perspective. We use visualization and graph reconstruction methods to measure the expressive power of each perspective. Specifically, we use VGAE [30] to reconstruct the representations learned by each perspective encoder on the Cora dataset and use the AUC (area under the curve of ROC) metric to plot cylindrical shapes. The results are shown in Fig. 4. The representations learned by each perspective encoder are also plotted using t-SNE [9], as shown in Fig. 5.

Additionally, we use the broken line chart to evaluate the semantic similarity between the global perspective embedding and the local perspective embeddings based on different training epochs ({1,5,10,20,40, 100}) on the Cora dataset. Specifically, each perspective embedding learns its representations through a shared GCN encoder, and their semantic similarity is measured by comparing the learned representations. The results are shown in Fig. 6. The description of each perspective is detailed in Table 4. It is clear that the expressive power (visual distribution and graph construction ability) of the representations learned by each perspective encoder is affected when the input graphs are attacked. Fused representation from local to global perspectives has stronger expressive power (as shown in Fig. 5) in noisy environments, i.e., LoGo-GNN is robust. The representation learned from a global perspective is not optimal in this scenario. On the other hand, the representation learned from the local perspective has better expressive power. Moreover, the difference in the learned representations between each local perspective and the global perspective increases with the proposed contrastive loss $\mathcal{L}_c$, which validates the importance of introducing local perspectives and also verifies our earlier analysis. Our aim is to maximize the differences in the representations learned by the encoders from different perspectives while ensuring that certain perspective encoders have exceptional learning abilities. It is worth noting that it does not always lead to a significant difference in the representation learned by encoders from different perspectives as the training epoch increases. This is attributed to the fact that the objective function is constrained by both contrast loss and object loss simultaneously.

Finally, the performance of LoGo-GNN, GCN [29] and Global-GNN (LoGo-GNN with global encoder perspective) are tested when dealing with some uncertainty issues in node classification tasks to demonstrate

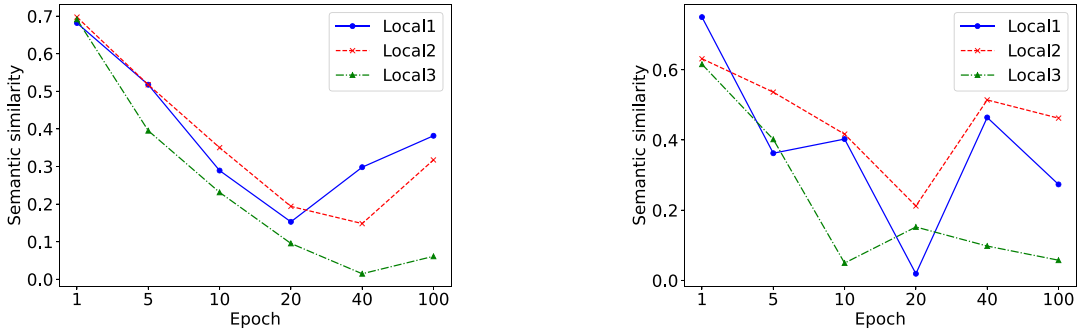


(a) Unattacked Global (b) Unattacked Local1 (c) Unattacked Local2 (d) Unattacked Local3 (e) Unattacked Local-Global



(f) Attacked Global (g) Attacked Local1 (h) Attacked Local2 (i) Attacked Local3 (j) Attacked Local-Global

Fig. 5. Visualization of the learned embeddings based on different perspectives on the Cora dataset. Unattacked * represent using representations learned from each unattacked perspective as inputs. Attacked * represent using representations learned from each attacked perspective as inputs, where we randomly delete some edges with an 80% ratio to get the modified graph as the input of each perspective encoder.



(a) Unattacked inputs (b) Attacked inputs

Fig. 6. Semantic similarity between the local perspective embeddings and the global perspective embedding on the Cora dataset. Unattacked inputs represent using representations learned from each unattacked perspective as inputs. Attacked inputs represent using representations learned from each attacked perspective as inputs, where we randomly delete some edges with an 80% ratio to get the modified graph as the input of each perspective encoder.

the robustness of our proposed model. We consider four types of uncertainty issues: low label rates, random topology attack, node feature mask, and node noise attack, which can lead to potential perturbations and affect classification performance. We only use the Cora dataset for experiments, with the number of labeled nodes being {14, 21, 28, 54}, attach ratios and mask ratios set to {0.2, 0.4, 0.6, 0.8}, and noise level set to {0.001, 0.01, 0.1, 1}. Specifically for the random topology attack/node feature mask, we randomly delete some edges/node features with a given ratio to obtain the modified graph/nodes and evaluate the classification accuracy on the (dirty) test set. Gaussian noise is added to the node features to disrupt the original feature values and affect the predicted results of the model for node noise attack. The parameter noise level represents the coefficient for the added noise. The experimental results are shown in Fig. 7. The following conclusions can be made:

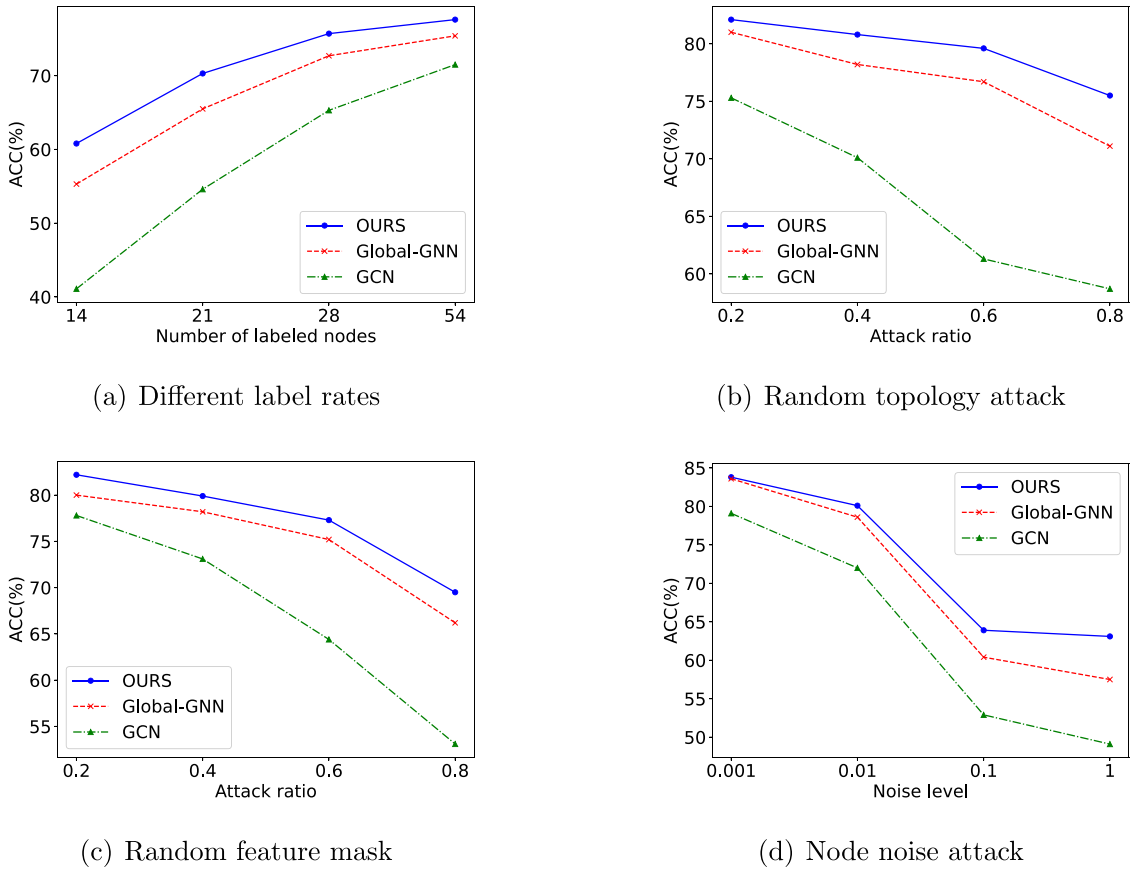
(1) It can be concluded from Fig. 7(a) that while the baseline performance rapidly deteriorates with a decrease in label rate, **OURS** performs exceptionally well even at extremely low label rates. Moreover, in scenarios where there are information constraints, considering both local and global information proves to be more effective in capturing key information. This is because **OURS** outperforms Global-GNN significantly, especially when the label rate is extremely low.

(2) It is clear from Fig. 7(b) that **OURS** outperforms the baselines at higher attack ratios. This is because **OURS** plays a critical role in learning from the global and local encoder perspectives. Overall, the performance of all methods decays rapidly with respect to the random attack ratio.

(3) Similar to the above analysis, it can be concluded from Figs. 7(c)–(d) that **OURS** consistently outperforms GCN and Global-GNN. Overall, the performance of all methods decays rapidly with respect to the mask ratio and noise level.

4.3. Performance on node clustering (Q5)

We also evaluate the performance of the proposed method on unsupervised graph representation learning. After training, we perform clustering tasks on the fused node embeddings $\boldsymbol{H}^{(1)}$ using the K-means algorithm over the Cora, Citeseer, Pubmed datasets, where two metrics are used for evaluation: normalized mutual information (NMI), and adjusted random index (ARI), with the mean and standard error of the metrics presented in Table 10. We select some representative unsupervised models and graph contrastive learning models as baselines, namely, K-means, Deepwalk [37], GraphCL [31], IGCL [36], GCA [20], GAE [30], and VGAE [30]. To verify the flexibility of LoGo-GNN, we use the loss functions of GCA [20], IGCL [36], and GraphCL [31]



(a) Different label rates

(b) Random topology attack

(c) Random feature mask

(d) Node noise attack

Fig. 7. Test performance comparison of GCN, Global-GNN, and **OURS** on Cora.

Table 10
Node clustering results for methods with different inputs, where $\boldsymbol{A}$ and $\boldsymbol{X}$ are the adjacency matrix and feature matrix (Bold: best)

Method	Training data	Cora		Citeseer		Pubmed	
		NMI	ARI	NMI	ARI	NMI	ARI
K-mean	$\boldsymbol{X}$	32.1	23.0	30.5	27.9	0.1	0.2
DeepWalk	$\boldsymbol{X}$	32.7	24.3	8.8	9.2	27.9	29.9
GAE	$\boldsymbol{X}$, $\boldsymbol{A}$	42.9	34.7	17.6	12.4	27.7	27.9
VGAE	$\boldsymbol{X}$, $\boldsymbol{A}$	23.9	17.5	15.6	9.3	22.9	21.3
GRAGE	$\boldsymbol{X}$, $\boldsymbol{A}$	54.4 ± 1.2	43.4 ± 3.1	35.2 ± 1.8	34.1 ± 1.7	30 ± 3.4	29.5 ± 2.1
OURS	$\boldsymbol{X}$, $\boldsymbol{A}$	55.9 ± 2.4	47.5 ± 2.4	39.5 ± 2.4	35.3 ± 2.4	32.7 ± 2.6	33.9 ± 2.1
GraphCL	$\boldsymbol{X}$, $\boldsymbol{A}$	55.3 ± 3.2	54.5 ± 2.7	42.1 ± 3.9	44.1 ± 1.8	34.1 ± 3.3	34.2 ± 2.8
OURS(GraphCL)	$\boldsymbol{X}$, $\boldsymbol{A}$	57.2 ± 2.1	56.3 ± 2.2	45.3 ± 1.9	44.5 ± 2.2	35.8 ± 2.4	34.0 ± 1.4
IGCL	$\boldsymbol{X}$, $\boldsymbol{A}$	56.6 ± 3.0	53.9 ± 2.0	43.5 ± 1.7	**45.2 ± 1.9**	33.4 ± 2.6	32.9 ± 1.1
OURS(IGCL)	$\boldsymbol{X}$, $\boldsymbol{A}$	**58.2** ± 1.2	**55.5** ± 2.3	**45.5** ± 2.4	45.0 ± 2.7	**37.7** ± 2.5	**35.9** ± 1.9

as the $\mathcal{I}$ loss (**OURS**, **OURS(GraphCL)**, **OURS(IGCL)**) in the loss function $\mathcal{L}_o$, respectively. It can be observed that LoGo-GNN achieves comparable results with other state-of-the-art models on most datasets. Taking the Pubmed dataset as an example, the ARI of **OURS(IGCL)** increased by 9.11% compared to IGCL. This reflects that LoGo-GNN is also highly competitive in unsupervised clustering tasks. Notably, the performance of LoGo-GNN heavily depends on the selection of $\mathcal{I}$ loss, which indirectly reflects the flexibility of LoGo-GNN.

4.4. Parameter sensitivity (Q6)

A sensitivity analysis is undertaken on the critical hyperparameter η to analyze the sensitivity of the hyperparameters. We train LoGo-GNN using η values approximately ranging from 0.05 to 0.95 in increments of 0.05. We empirically find that η values within the range of 0.4 to 0.8 yield satisfactory results, which show that model performance can be improved by paying appropriate attention to the representation relationships of learning from different encoder perspectives. In addition, we conducted an evaluation on how the hyperparameters λ_1 and λ_2 impact the performance of the model. Specifically, we set η at a fixed value and varied the values of λ_1 and λ_2 from 0 to 0.9. Then, we displayed the results of all node classifications on a 3D bar chart, which is illustrated in Fig. 8. It can be observed that the effect of self-supervised learning will be better when both λ_1 and λ_2 have higher values.

5. Conclusion

In this study, we introduce collaborative graph neural networks for augmented graphs (LoGo-GNN), a novel architecture that addresses the limitations of existing graph neural networks (GNNs) in capturing the internal local collaboration among input graphs. By employing a perturbation augmentation strategy, it generates multiple input graphs

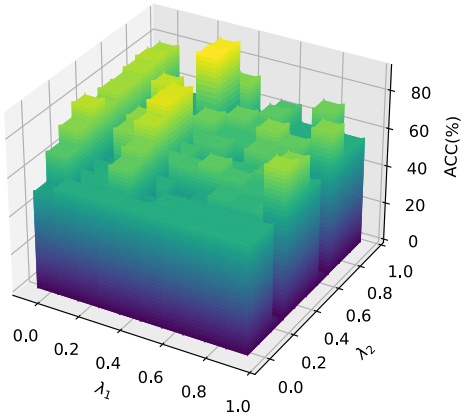


Fig. 8. The performance of **OURS-UN** with varying hyperparameters in node classification on the Cora dataset in terms of ACC (%).

and strategically pairs them, facilitating collaboration across different scales and enabling representation learning from both local and global encoder perspectives. It is worth noting that the loss function of LoGo-GNN includes a novel complementary loss and an object loss to guide the efficient node embedding fusion and collaboration between different perspectives.

The strengths of our method over existing works lie in its greater effectiveness and robustness in handling multiple augmented graphs. LoGo-GNN stands out as an effective and robust solution, offering a comprehensive approach from a local-to-global perspective. Theoretical analysis conducted under ideal conditions and related experiments demonstrate the expressive power of LoGo-GNN and validate its effectiveness and robustness.

However, the work primarily focuses on generating a local-to-global perspective, where information from each perspective undergoes simple unified processing. Moreover, the model's time complexity increases non-linearly with the number of perspectives. In the future, we plan to implement dynamic sampling of different perspective information and consider extracting hierarchical information from each perspective rather than simply unifying the information from each perspective.

CRediT authorship contribution statement

Qihang Guo: Writing – review & editing, Writing – original draft, Methodology. **Xibei Yang:** Writing – review & editing, Supervision, Resources. **Ming Li:** Writing – review & editing, Supervision. **Yuhua Qian:** Supervision, Investigation.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgments

Xibei Yang acknowledged the supports from the National Natural Science Foundation of China (No. 62076111). Ming Li acknowledged the supports from the Jinhua Science and Technology Plan (No. 2023-3-003a).

References

[1] Z. Wu, S. Pan, F. Chen, et al., A comprehensive survey on graph neural networks, IEEE Trans. Neural Netw. Learn. Syst. 32 (1) (2021) 4–24.
[2] S. Liu, R. Ying, H. Dong, et al., Local augmentation for graph neural networks, in: Proceedings of International Conference on Machine Learning, ICML 2022, Baltimore, Maryland, USA, 2022.
[3] J. Wang, J. Liang, K. Yao, et al., Graph convolutional autoencoders with co-learning of graph structure and node attributes, Pattern Recognit. 121 (2022) 108215.
[4] S. Abu-El-Haija, A. Kapoor, B. Perozzi, et al., N-GCN: multi-scale graph convolution for semi-supervised node classification, in: Proceedings of the 35th Conference on Uncertainty in Artificial Intelligence, UAI 2019, Tel Aviv, Israel, 2019.
[5] P. Velickovic, G. Cucurull, A. Casanova, et al., Graph attention networks, in: Proceedings of 6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, 2018.
[6] L. Bai, Y. Zhao, J. Liang, Self-supervised spectral clustering with exemplar constraints, Pattern Recognit. 132 (2022) 108975.
[7] M. Li, L. Zhang, L. Cui, et al., BLoG: Bootstrapped graph representation learning with local and global regularization for recommendation, Pattern Recognit. 144 (2023) 109874.
[8] Q. Guo, X. Yang, F. Zhang, et al., Perturbation-augmented graph convolutional networks: A graph contrastive learning architecture for effective node classification tasks, Eng. Appl. Artif. Intell. 129 (2024) 107616.
[9] X. Wang, M. Zhu, D. Bo, et al., AM-GCN: adaptive multi-channel graph convolutional networks, in: Proceedings of 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD 2020, CA, USA, 2020.
[10] K. Yao, J. Liang, J. Liang, et al., Multi-view graph convolutional networks with attention mechanism, Artificial Intelligence 307 (2022) 103708.
[11] L. Zhang, H. Song, N. Aletras, et al., Node-feature convolution for graph convolutional networks, Pattern Recognit. 128 (2022) 108661.
[12] T. Jin, H. Dai, L. Cao, et al., Deepwalk-aware graph convolutional networks, Sci. China Inf. Sci. 65 (5) (2022) 1–15.
[13] J. Bi, F. Dornaika, Sample-weighted fused graph-based semi-supervised learning on multi-view data, Inf. Fusion 104 (2024) 102175.
[14] L. Zhong, J. Lu, Z. Chen, et al., Adaptive multi-channel contrastive graph convolutional network with graph and feature fusion, Inform. Sci. 658 (2024) 120012.
[15] Y. Yang, H. Lv, N. Chen, A survey on ensemble learning under the era of deep learning, Artif. Intell. Rev. 56 (6) (2023) 5545–5589.
[16] V.P. Dwivedi, C.K. Joshi, A.T. Luu, et al., Benchmarking graph neural networks, J. Mach. Learn. Res. 24 (2023) 43:1–43:48.
[17] S. Abu-El-Haija, B. Perozzi, A. Kapoor, et al., MixHop: Higher-order graph convolutional architectures via sparsified neighborhood mixing, in: Proceedings of the 36th International Conference on Machine Learning, ICML 2019, Long Beach, California, USA, 2019.
[18] J. Wang, J. Liang, J. Cui, et al., Semi-supervised learning with mixed-order graph convolutional networks, Inform. Sci. 573 (2021) 171–181.
[19] J. Qiu, Q. Chen, Y. Dong, et al., GCC: graph contrastive coding for graph neural network pre-training, in: Proceedings of 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD 2020, CA, USA, 2020.
[20] Y. Zhu, Y. Xu, F. Yu, et al., Graph contrastive learning with adaptive augmentation, in: Proceedings of the Web Conference 2021, WWW 2021, Slovenia, 2021.
[21] P. Velickovic, W. Fedus, W.L. Hamilton, et al., Deep graph infomax, in: Proceedings of 7th International Conference on Learning Representations, ICLR 2019, New Orleans, la, USA, 2019.
[22] X. Shen, D. Sun, S. Pan, et al., Neighbor contrastive learning on learnable graph augmentation, in: B. Williams, Y. Chen, J. Neville (Eds.), Proceedings of 37th AAAI Conference on Artificial Intelligence, AAAI 2023, Washington, DC, USA, 2023, pp. 9782–9791.
[23] Y. Liu, M. Jin, S. Pan, et al., Graph self-supervised learning: A survey, IEEE Trans. Knowl. Data Eng. 35 (6) (2023) 5879–5900.
[24] Z. Li, Y. Zhao, Y. Zhang, et al., Multi-relational graph attention networks for knowledge graph completion, Knowl.-Based Syst. 251 (2022) 109262.
[25] Y. Liu, Y. Zhao, X. Wang, et al., Multi-scale subgraph contrastive learning, in: Proceedings of the 32th International Joint Conference on Artificial Intelligence, IJCAI 2023, Macao, SAR, China, 2023.
[26] J. Guo, K. Huang, X. Yi, et al., Learning disentangled graph convolutional networks locally and globally, IEEE Trans. Neural Netw. Learn. Syst. 35 (3) (2024) 3640–3651.

[27] M.R. Khan, J.E. Blumenstock, Multi-GCN: Graph convolutional networks for multi-view networks, with applications to global poverty, in: Proceedings of 33rd AAAI Conference on Artificial Intelligence, AAAI 2019, Honolulu, Hawaii, USA, 2019, pp. 606–613.
[28] S. Piramuthu, Feed-forward neural networks and feature construction with correlation information: an integrated framework, European J. Oper. Res. 93 (2) (1996) 418–427.
[29] T.N. Kipf, M. Welling, Semi-supervised classification with graph convolutional networks, in: Proceedings of 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, 2017.
[30] T.N. Kipf, M. Welling, Variational graph auto-encoders, 2016, CoRR. doi:abs/1611.07308.
[31] Y. You, T. Chen, Y. Sui, et al., Graph contrastive learning with augmentations, in: Proceedings of 33rd Annual Conference on Neural Information Processing Systems, NeurIPS 2020, 2020.
[32] N. Murata, An integral representation of functions using three-layered networks and their approximation bounds, Neural Netw. 9 (6) (1996) 947–956.
[33] A. Bojchevski, S. Günnemann, Deep Gaussian embedding of graphs: Unsupervised inductive learning via ranking, in: Proceedings of 6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, 2018.
[34] X. Wang, H. Ji, C. Shi, et al., Heterogeneous graph attention network, in: Proceedings of the World Wide Web Conference, WWW 2019, San Francisco, CA, USA, 2019.
[35] H. Pei, B. Wei, K.C. Chang, et al., Geom-GCN: Geometric graph convolutional networks, in: Proceedings of 8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, 2020.
[36] H. Liang, X. Du, B. Zhu, et al., Graph contrastive learning with implicit augmentations, Neural Netw. 163 (2023) 156–164.
[37] B. Perozzi, R. Al-Rfou, S. Skiena, DeepWalk: online learning of social representations, in: Proceedings of 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2014, New York, NY, USA, 2014.
[38] J. Chen, H. ren Fang, Y. Saad, Fast approximate *k*NN graph construction for high dimensional data via recursive lanczos bisection, J. Mach. Learn. Res. 10 (2009) 1989–2012.
[39] D.P. Kingma, J. Ba, Adam: A method for stochastic optimization, in: Proceedings of 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, 2015.
[40] X. Zhang, G. Ding, J. Li, et al., Deep learning empowered MAC protocol identification with squeeze-and-excitation networks, IEEE Trans. Cogn. Commun. Netw. 8 (2) (2022) 683–693.

Qihang Guo He is currently a Ph.D student in the Jiangsu University of Science and Technology. His research focuses on deep learning and graph learning.

Xibei Yang He was born in 1980. He is a Ph.D. and professor. His main research interests include rough set, granular computing, and machine learning.

Ming Li He is a 'Shuang Long Scholar' Distinguished Professor, Institute of Electrical and Electronics Engineers (IEEE) Member, Chinese Association for Artificial Intelligence (CAAI) Member, China Computer Federation (CCF) Member. His main research interests include graph neural networks, graph representation learning, graph data mining.

Yuhua Qian He is a Ph.D., professor, IEEE Member, and ACM Member. His main research interests include intelligent information processing, rough set, granular computing, deep learning, and machine learning.