Vertex Entanglement-Enhanced Graph Contrastive Learning for Social Networks

Xinlan Xu, Fei Hao, Bo Li, Hongying Zhang, Wangyang Yu, and Geyong Min, *Member, IEEE*

***Abstract*—Graph contrastive learning can extract high-quality representations from graph-structured data, providing an effective way to analyze social networks. However, existing graph contrastive learning methods with both random augmentations and advanced adaptive strategies struggle to effectively preserve key structural semantics and eliminate graph noise, which ultimately limits representation quality. To this end, this article proposes a vertex entanglement-enhanced graph contrastive learning (VEGCL), which utilizes vertex entanglement (VE) as a novel metric that quantifies node importance from a global structural perspective to guide the removal of unimportant nodes and edges for generating augmented views. Specifically, VE is initially employed to selectively remove less important nodes, generating the first augmented view. Subsequently, the original graph is transformed into a line graph, after which VE is again applied to remove less critical edges, producing the second augmented view. Following augmentation, model training is guided by the Info Noise Contrastive Estimation loss function to enhance embedding quality. To verify the effectiveness of the proposed approach, node classification experiments are conducted on five commonly used benchmark datasets including Cora, PubMed, CiteSeer, Coauthor-CS, and LastFM Asia. The experimental results demonstrate that the proposed method improves the performance of node classification by $0.5\% \sim 2.3\%$, and $0.04\% \sim 5\%$, in terms of accuracy and F1 score compared with the existing baseline methods.**

***Index Terms*—Data augmentation, graph contrastive learning, graph representation learning, vertex entanglement (VE).**

I. Introduction

In the era of rapid advancements in artificial intelligence and sensing technologies, social intelligence, defined as the ability to model, analyze, and predict human-driven relational dynamics, has emerged as a cornerstone of understanding complex societal systems. At the core of this paradigm lie graph-structured data, which encodes the complex connections within social ecosystems: from user interactions in online platforms [1] to collaborative relationships in academic citation networks [2], [3], and preference linkages in recommendation systems [4], [5]. As a critical subset of social networks, citation networks depict academic dependencies and knowledge dissemination. In such networks, nodes represent researchers or publications, while edges denote knowledge influence. Effectively analyzing such graph data is pivotal for developing foundation models that can decode social intelligence, enabling applications ranging from identifying key opinion leaders in online communities to uncovering emerging research frontiers in academic networks. As a novel method of graph representation learning, graph contrastive learning (GCL) aims to learn high-quality representation of graph data through a contrastive learning framework [6], [7], [8]. GCL can effectively capture both local and global structural features in graphs, which provides a solid foundation for various graph-related downstream tasks such as node classification [9], [10], node clustering [11], [12] and link prediction [13], [14].

There has been a lot of research focusing on GCL. For example, Wu et al. [15] proposed three random augmentation strategies of node removal, edge removal, and random walk to generate augmented views, which change the structure of the graph in different ways. Yu et al. [16] generated two augmented views by adding random uniform noise to the original node representations to improve model efficiency. Zhu et al. [17] generated augmented views by randomly removing edges and masking node attributes. However, these traditional GCL methods generally rely on random strategies when generating augmented views, which can easily disrupt important structural semantics in graph data. Although recent research has attempted to address this issue via adaptive approaches, significant limitations persist. For example, while methods like AutoGCL [18] introduced adaptive augmentation mechanisms, they often remain confined to node-level operations and lack the ability to explore richer structural variations through edge-level perturbation. Methods such as CONVERT [19] and MA-GCL [10] primarily focus on augmentation at the embedding

Received 23 April 2025; revised 27 November 2025 and 21 January 2026; accepted 29 January 2026. This work was supported in part by the National Natural Science Foundation of China under Grant 62477029 and Grant 61702317, in part by the Ministry of Education Humanities and Social Sciences Research Youth Fund Project under Grant 22YJCZH046, and in part by the Fundamental Research Funds for the Central Universities under Grant GK202505028. *(Corresponding author: Fei Hao.)*

Xinlan Xu, Fei Hao, and Wangyang Yu are with the School of Artificial Intelligence and Computer Science, Shaanxi Normal University, Xi'an 710119, China (e-mail: feehao@gmail.com).

Bo Li is with the School of Computer and Software Engineering, Xihua University, Chengdu 610039, China.

Hongying Zhang is with the School of Mathematics and Statistics, Xi'an Jiaotong University, Xi'an 710049, China.

Geyong Min is with the Department of Computer Science, University of Exeter, Exeter EX4 4QJ, U.K.

Digital Object Identifier 10.1109/TCSS.2026.3660378

space and model level, rather than the original graph structure. This limitation prevents them from fundamentally mitigating the interference from redundant edges and noisy nodes in the original graph, resulting in residual noise information within the learned representations.

To address the aforementioned issues, this article proposes a novel VEGCL method. The core of our approach lies in vertex entanglement (VE), defined as a perturbation of spectral entropy, which serves as a graph metric designed to quantify the global importance of nodes and edges from a holistic perspective. This approach leverages VE to assess the importance of nodes and edges in the original graph, selectively removing those with minimal impact on the global structure to preserve critical structural semantics. This effectively reduces interference from redundant information. Consequently, it generates effective augmented views without compromising critical semantic information. The model is then trained using the info noise contrastive estimation (InfoNCE) loss function to obtain more robust and discriminative embeddings. The major contributions of this article are summarized as follows.

1) **Introducing VE as a Novel Global Node Importance Measure:** To address the inherent local limitations of traditional centrality metrics, we introduce VE. Based on graph Laplacian spectral entropy, VE quantifies node importance from a global functional perspective. Both theoretical analysis and experiments demonstrate that VE provides unique, nonredundant information compared to traditional metrics, offering a new and essential dimension for network analysis.
2) **Proposing a Novel VEGCL Method:** We address the issues of structural semantics disruption caused by existing augmentation methods and inherent structural noise in the original graph by introducing VE, which quantifies global network functionality to guide the augmentation process. We innovatively employ line graph transformation to convert edge evaluation into node evaluation, enabling the global functionality metric VE to be consistently applied for ranking both nodes and edges. This achieves precise removal of redundant nodes and edges, maximizing preservation of the graph's original semantic integrity while enhancing view diversity.
3) **Evaluation of the Effectiveness of VEGCL:** We apply the VEGCL algorithm to the node classification task for evaluating its performance. The experimental results show that the VEGCL algorithm is significantly better in ACC and F1 scores compared with the existing baseline methods. In particular, the VEGCL algorithm improves on the ACC by 2.3% and on the F1 score by 5% compared to the previously optimal methods on the LastFM Asia dataset. In addition, the ablation study is conducted for evaluating the importance of graph data augmentation. Experimentally, our proposed VEGCL on the CiteSeer dataset obtains a 3% improvement in ACC and a 2.67% improvement in F1 compared to a model that uses only one strategy to generate augmented views.

The remainder of this article is organized as follows. In Section II, we present the related work in detail. Then, the problem statement is provided in Section III. In Section IV, we propose a VEGCL method. After that, we validate the feasibility of the method proposed in this article through experiments and analyze the results in Section V. Finally, Section VI summarizes this article and presents the direction of future work.

II. Related Work

GCL is an approach based on a self-supervised learning framework dedicated to learning representations of graph data [20]. In recent years, the literatures on GCL has mainly focused on two aspects of graph data augmentation methods and loss function design.

The most important principle of graph data augmentation is not to fundamentally change the structure information and text information of the graph, so as to avoid affecting the performance of downstream tasks [21], [22]. Preserving critical information during graph augmentation can be achieved by quantifying node importance. Numerous studies have proposed methods for quantifying node importance, including early traditional centrality measures such as degree centrality, as well as more recent approaches like collective network entanglement (CNE) [23] and VE [24]. In recent years, graph data augmentation approaches have been extensively studied [25]. For example, Thakoor et al. [26] used random augmentation to generate augmented views for contrastive learning. Li et al. [27] proposed to encode the data by employing two different graph neural networks (GNNs) to construct augmented views where node representations have differences. Zou et al. [28] and Xie et al. [29] proposed performing GCL on multiple views to fully exploit the diversity and complexity of graph data. Duan et al. [30] and Qin et al. [31] developed frameworks for generating augmented views at different topological scales or from diverse feature perspectives, enhancing anomaly detection capabilities by contrasting these multifaceted representations. Liu et al. [32] improved clustering performance by combining a Gaussian noise augmentation strategy in embedding space with a structural contrast loss.

The essential of loss function design is the way the sample pairs are constructed [33], [34]. Research work on GCL for sample pairs has been widely studied in recent years [35], [36], [37]. For example, Zhang et al. [38] generated negative sample nodes with high hardness for anchor nodes through multisample mixing to enhance graph comparison learning. Lin et al. [39] clustered semantically similar nodes together based on a semantic perspective, with the center of the cluster serving as the prototype of the group. When an anchor node is given, nodes in other clusters serve as negative samples. Tang et al. [40] obtained multiple positive samples by random sampling and added weight coefficients to the loss function to measure the importance of positive and negative samples in different graph data, so as to improve the utilization of positive samples. Zhu et al. [41] partitioned the heterogeneous graph into individual subgraphs via metapaths and performed hard-negative sample construction based on graph structure in each subgraph.

From the above mentioned research status on GCL, the field has made remarkable achievements in the two core directions of

graph data augmentation and loss function design. In graph data augmentation, researchers have proposed various innovative approaches to create augmented views that achieve the goal of improving model efficiency and reducing the reliance on manual experience. In loss function design, researchers have explored various ways of constructing sample pairs to improve the performance of GCL. However, many GCL methods still face challenges during data augmentation, primarily because they struggle to effectively preserve critical structural semantics while simultaneously eliminating structural noise from the original graph, which ultimately compromises the quality of the learned representations. In addition, although existing methods have introduced innovations in sample pair construction, they generally suffer from limitations in scalability and stability, such as high computational overhead or complex hyperparameter tuning. To address these challenges, we propose VEGCL. This method employs a VE-based data augmentation strategy, which preserves crucial semantic information in the graph structure while mitigating noise interference from the original graph. Furthermore, we select the InfoNCE [34] loss for model optimization due to its efficient negative sample processing and tunable temperature mechanism, which collectively enable more efficient and stable model optimization.

III. Problem Statement

Given a graph $G = (V, E)$, where $V = \{v_1, v_2, \ldots, v_n\}$ denotes the set of nodes, $E = \{e_1, e_2, \ldots, e_m\}$ denotes the set of edges and $E \in V \times V$. The adjacency matrix is denoted as $A \in R^{n \times n}$, where $A_{ij} = 1$ when there exists an edge between node v_i and node v_j. Otherwise, $A_{ij} = 0$. Our goal is to learn a GNN encoder $f(\cdot)$ to obtain a better node representation vector h_i. Where $h_i \in H$ denotes the vector representation of node $v_i \in V$. We define the representation learning of a node as follows:

$$H = f(\cdot). \tag{1}$$

Specifically, two augmented views are first generated based on the input graph G. Then, a GNN encoder is used to encode the two augmented views to obtain the node representations. Finally, an optimization algorithm is used to minimize the contrastive loss function $J = (1/2N) \sum_{i=1}^{N} [\ell(h_i(1)) + \ell(h_i(2))]$, where N is the total number of nodes, i denotes the node index, $\ell(h_i(1))$ and $\ell(h_i(2))$ represent the loss of node i in the two augmented views, respectively. The optimization objective is to find the parameter θ that minimizes this loss

$$\widetilde{\theta} = \arg\min_{\theta} J. \tag{2}$$

Equation (2) represents the value of the model parameter θ when the loss function J reaches its minimum value as $\widetilde{\theta}$.

Table I provides a detailed overview of the major notations used in this article.

TABLE I
Major Notations Used in This Article

Notation	Descriptions
G	Original graph
V	Node set of graph G
E	Edge set of graph G
A	Adjacency matrix of graph G
$f(\cdot)$	GNN encoder
J	Contrastive loss
θ	Model parameter
$L(G)$	Line graph
LV	Node set of line graph $L(G)$
LE	Edge set of line graph $L(G)$
$S_\tau(\cdot)$	Spectral entropy
L	Laplacian matrix
D	Degree matrix
ρ	Density operator
Z	Partition function
$E_\tau(v)$	VE value of node v in original graph
G_v	Perturbed graph
p_v	Probability of deleting node v
p_e	Probability of deleting edge e
p_c	Hyper-parameter
p_s	Cut-off probability
τ	Temperature parameter
$\widetilde{G}_1, \widetilde{G}_2$	Augumented views
$\widetilde{E}_\tau(v)$	VE value of node v in line graph
N	Number of nodes

IV. Methodology

We propose a VEGCL method, which tends to retain important nodes and edges and delete insignificant nodes and edges in the process of generating augmented views. The overall framework of the VEGCL is shown in Fig. 1.

To be more specific, it first applies VE to the input graph G to obtain the importance of nodes, and deletes insignificant nodes to get an augmented view. Then, it converts the original input graph G into a line graph $L(G)$, and uses VE on the line graph to obtain the importance of edges, and deletes insignificant edges to obtain another augmented view. After that, the two augmented views are input to the encoder, which is then passed through a nonlinear projection layer and projectively embedded to obtain a low-dimensional node representation vector. Finally, the loss function is used to train the GNN encoder and its parameters are optimized so that the resulting encoder has better node representation.

A. Graph Data Enhancement Based on VE

This section introduces two graph data augmentation methods based on VE: node removal and edge removal. For node removal, VE is directly applied to identify and remove less significant nodes. For edge removal, the original graph is first transformed into its line-graph representation. VE is subsequently employed to evaluate and remove unimportant edges.

1) Node Removal Based on VE: For each node, VE is used to calculate its importance. Then, based on this importance, less important nodes are removed along with their connected edges. The details are as follows.

VE is a measure of node influence that captures global topological information about the network. It reveals the importance of nodes in the network. Specifically, VE measures the impact of localized changes in a single node on the spectral entropy of the network.

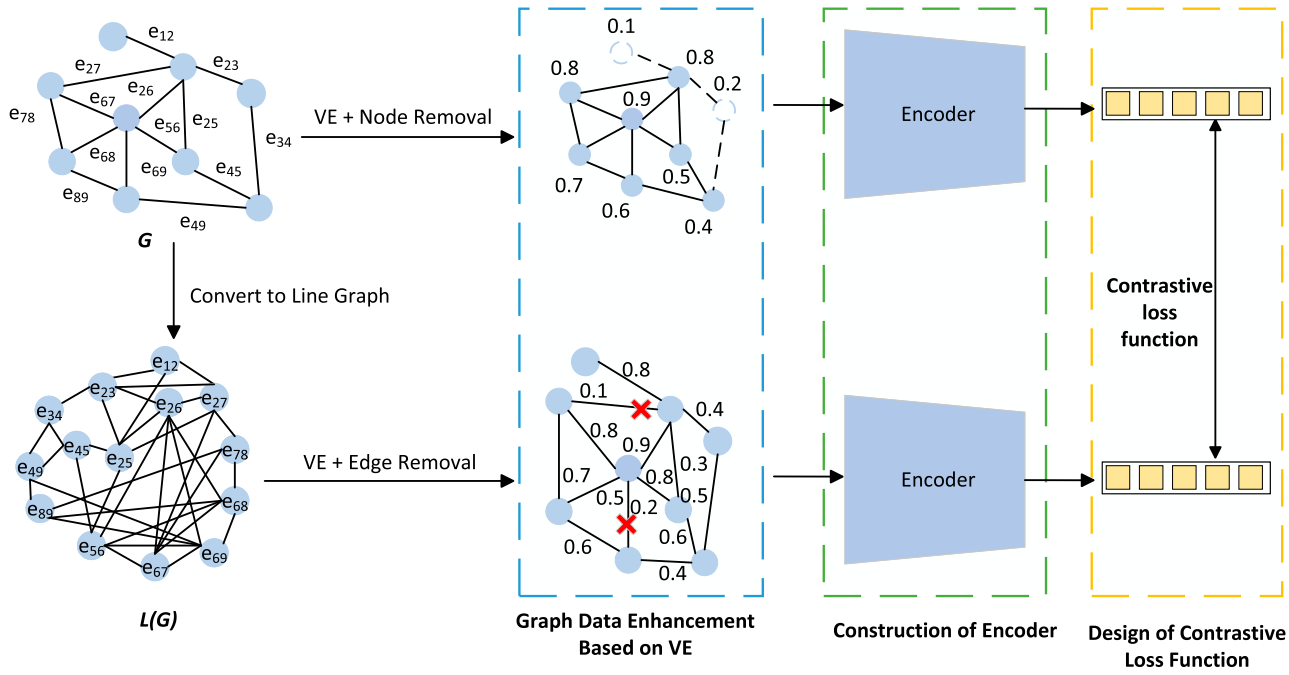

Fig. 1. VEGCL framework.

Spectral entropy [42] is defined as follows:

$$S_\tau(\rho) = -tr(\rho \log_2 \rho) \tag{3}$$

where the density matrix $\rho = (e^{-\tau L}/Z)$, $Z = tr(e^{-\tau L}) = \sum_{i=1}^{N} e^{-\tau \lambda_i(L)}$ of the network G is the partition function of the system and the parameter τ encodes the diffusion propagation time. $e^{-\tau L}$ is the diffusion propagator, $L = D - A$ is the Laplace matrix, D is the degree matrix of network G, and A is the adjacency matrix of network G. For simplicity, we use the notation $S_\tau(G) = S_\tau(\rho)$ to denote the spectral entropy of network G.

Therefore, VE is defined as follows:

$$E_\tau(v) = S_\tau(G_v) - S_\tau(G) \tag{4}$$

where G denotes the original network, G_v denotes the network after specific perturbation of node v in the original network, $S_\tau(G)$ denotes the spectral entropy of the original network, and $S_\tau(G_v)$ denotes the spectral entropy of the network after specific perturbation of node v in the original network. As can be seen from (4), $E_\tau(v) \le 0$.

*Theorem 1:*Let $E_\tau(v) = S_\tau(G_v) - S_\tau(G)$ denote the value of VE, then $E_\tau(v) \le 0$.

Proof: To prove $E_\tau(v) \le 0$, we need to prove that $S_\tau(G_v) \le S_\tau(G)$. According to the definition of spectral entropy, we can obtain

$$S_\tau(G) = -tr(\rho \log_2 \rho) = -\sum_{i=1}^{N} \lambda_i(\rho) \log_2 \lambda_i(\rho)$$

$$S_\tau(G_v) = -tr(\rho' \log_2 \rho') = -\sum_{i=1}^{N} \lambda_i'(\rho) \log_2 \lambda_i'(\rho).$$

Since $(e^{-\tau \lambda_i'(L)}/Z') \ge (e^{-\tau \lambda_i(L)}/Z)$ and $\lambda_i(\rho) = (e^{-\tau \lambda_i(L)}/Z)$, we can obtain that $\lambda_i'(\rho) \ge \lambda_i(\rho)$. Where $(e^{-\tau \lambda_i'(L)}/Z') \ge (e^{-\tau \lambda_i(L)}/Z)$ has been proved in [24]. This means that $S_\tau(G_v) \le S_\tau(G)$. Hence, $E_\tau(v) = S_\tau(G_v) - S_\tau(G) \le 0$.

Therefore, the smaller the value of $E_\tau(v)$, the greater the influence of node v on the network connectivity and the stronger the entanglement. That is, the smaller the value of $E_\tau(v)$, it proves that the node is more important and less likely to be removed. Conversely, the larger the value of $E_\tau(v)$, the more likely it is to be removed.

To obtain the augmented view, the probability of a subset $\widetilde{V}$ sampled from the original node set V is defined as follows:

$$P\{v \in \widetilde{V}\} = 1 - p_v \tag{5}$$

where $v \in V$, p_v represents the probability of deleting a node v. That is, nodes are retained with a probability of $1 - p_v$ to form a modified subset $\widetilde{V}$ to generate the augmented view. The importance of the nodes is reflected by p_v to realize the deletion of insignificant nodes while retaining important nodes in the augmented view.

The node importance is calculated based on VE, since the value of $E_\tau(v)$ is negative and a smaller value of $E_\tau(v)$ corresponds to a more important node. Therefore, we perform an inverse operation on $E_\tau(v)$ and normalize it to finally obtain the formula for p_v as shown in (7). Since the inverse operation is performed on $E_\tau(v)$, a higher value of $E_\tau'(v)$ indicates a higher importance of the node

$$E_\tau'(v) = -E_\tau(v) \tag{6}$$

$$p_v = \min\left\{\frac{E_\tau'(v) - E_\tau'(v)_{\min}}{E_\tau'(v)_{\max} - E_\tau'(v)_{\min}} \cdot p_{c_1}, p_{s_1}\right\}. \tag{7}$$

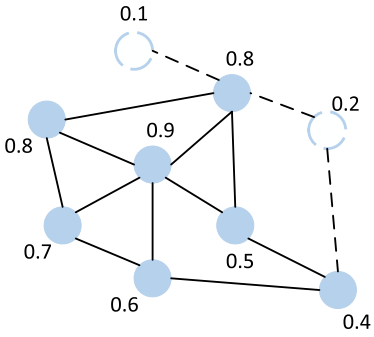

Fig. 2. Framework for node removal.

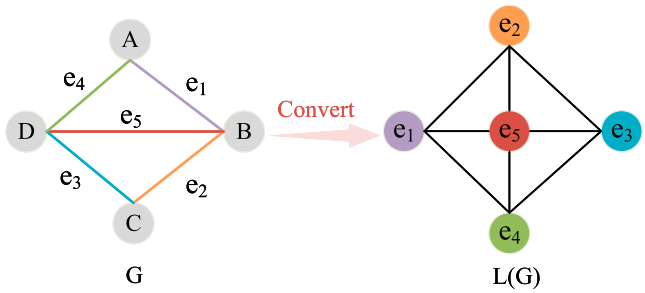

Fig. 3. Line graph conversion process.

Here, p_{c_1} is a hyperparameter intended to control the overall probability of deleting a node. $p_{s_1} < 1$ is the cutoff probability of the node, which is used to limit the maximum value of the probability, because too high a deletion probability can severely damage the graph structure. We employ a coarse-to-fine tuning strategy, which begins with a broad grid search to identify promising parameter ranges and then proceeds with Bayesian optimization for precise refinement within these intervals.

Fig. 2 illustrates the node removal framework, where the numerical values represent node importance calculated based on VE. Assuming we set the node removal threshold to 0.2, all nodes with importance below this threshold and their connected edges will be removed from the graph.

2) Edge Removal Based on VE: For edges, the original graph is first converted to a line graph and then uses VE, so as to reflect the importance of edges in the network. Edges with high importance are retained, while edges with less importance are more likely to be removed. The detailed steps are as follows.

At first, the original graph is converted to a line graph. Given a graph $G = (V, E)$, then $L(G) = (LV, LE)$ is a line graph representing the adjacencies of edges in the graph G. $L(G)$ can be constructed by converting nodes in the original graph G to edges in $L(G)$, and edges in the original graph G to nodes in $L(G)$. Moreover, if two nodes in $L(G)$ have the same common endpoint in the original graph G, then those two nodes in $L(G)$ have an edge. For example, Fig. 3 illustrates the process of converting an original graph G into its line graph $L(G)$. G consists of four nodes A, B, C, and D, interconnected by five edges: the purple edge e_1 connects A and B, the orange edge e_2 connects B and C, the blue edge e_3 connects C and D, the green edge e_4 connects D and A, and the red edge e_5 connects B and D. Through the line graph conversion process, each edge in G is mapped to a node in $L(G)$, meaning e_1 to e_5 correspond to the five nodes in $L(G)$, respectively. The connections between nodes in $L(G)$ are determined by the adjacency of edges in G: if two original edges share a common node, their corresponding nodes in $L(G)$ are connected. Specifically, since edge e_5 is adjacent to e_1, e_2, e_3, and e_4 at nodes B or D, node e_5 in $L(G)$ establishes connections with nodes e_1, e_2, e_3, and e_4. The connections between other nodes similarly follow this adjacency rule, ultimately forming the $L(G)$ comprising five nodes.

Then, the value of VE of the line graph is calculated using (8) to obtain the importance of the nodes in the line graph which is the importance of the edges in the original graph

$$\widetilde{E}_\tau(v) = S_\tau(L(G_v)) - S_\tau(L(G)). \tag{8}$$

Here, $L(G)$ denotes a line graph, $L(G_v)$ denotes a graph after a specific perturbation of a node v in the line graph, $S_\tau(L(G))$ denotes the spectral entropy of the line graph, and $S_\tau(L(G_v))$ denotes the spectral entropy of the graph after a specific perturbation of a node v in the line graph. Because the value of $\widetilde{E}_\tau(v)$ is negative, the larger the value of $\widetilde{E}_\tau(v)$ represents the less important the node is and the more likely it is to be removed. In other words, the larger the absolute value of $\widetilde{E}_\tau(v)$, the more important the node is and the less likely it is to be removed.

Similarly, the concept of $S_\tau(L(G)) = S_\tau(\rho)$ is employed to represent the spectral entropy of a line graph $L(G)$. That is, $S_\tau(L(G)) = -tr(\rho \log_2 \rho)$.

Finally, the probability of sampling a modified subset $\widetilde{E}$ from the original set of edges E is defined as follows:

$$P\{e \in \widetilde{E}\} = 1 - p_e \tag{9}$$

where $e \in E$, p_e represents the probability of deleting an edge e. That is, edges are retained with a probability of $1 - p_e$ to form a modified subset $\widetilde{E}$ to generate the augmented view. The importance of the edges is reflected by p_e to realize the deletion of insignificant edges while retaining important edges in the augmented view.

Similarly, since the computed value of $\widetilde{E}_\tau(v)$ is negative, and smaller values of $\widetilde{E}_\tau(v)$ correspond to more important edges. Therefore, we perform an inverse operation on $\widetilde{E}_\tau(v)$ and normalize it to finally obtain the formula for p_e as shown in (11). Since the inverse operation is performed on $\widetilde{E}_\tau(v)$, a higher value of $\widetilde{E}'_\tau(v)$ indicates a higher importance of the edge

$$\widetilde{E}'_\tau(v) = -\widetilde{E}_\tau(v) \tag{10}$$

$$p_e = \min\left\{\frac{\widetilde{E}'_\tau(v) - \widetilde{E}'_\tau(v)_{\min}}{\widetilde{E}'_\tau(v)_{\max} - \widetilde{E}'_\tau(v)_{\min}} \cdot p_{c_2}, p_{s_2}\right\}. \tag{11}$$

Here, p_{c_2} is a hyperparameter intended to control the overall probability of deleting an edge. $p_{s_2} < 1$ is the cutoff probability of the edge, which is used to limit the maximum value of the probability, because too high deletion probability can severely damage the graph structure. We employ a coarse-to-fine tuning

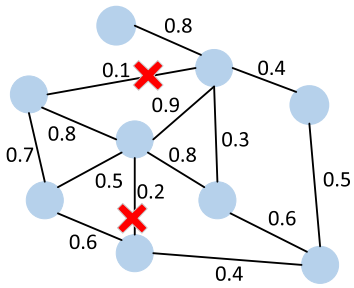

Fig. 4. Framework for edge removal.

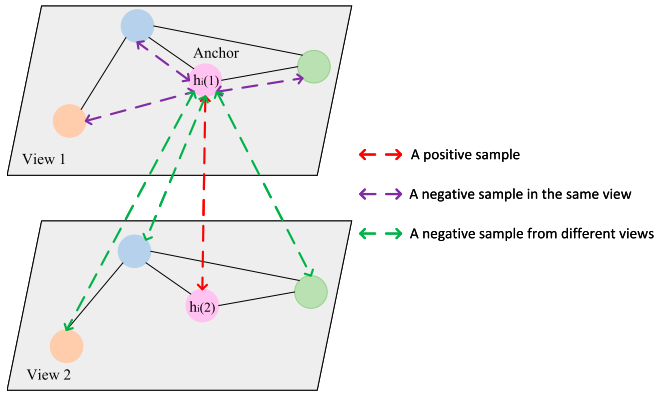

Fig. 5. InfoNCE loss function.

strategy, which begins with a broad grid search to identify promising parameter ranges and then proceeds with Bayesian optimization for precise refinement within these intervals.

The edge removal framework is illustrated in Fig. 4, where the annotated values correspond to the importance of each edge. The core goal of edge removal is to eliminate edges with low importance in the graph. Taking Fig. 4 as an example, if the threshold is set to 0.2, all edges with importance lower than this value will be removed.

B. Construction of Encoder

In this section, we introduce GNN to build the encoder. The augmented view is encoded by using a GNN encoder to obtain node representation vectors [43].

Given a graph $G = (V, E)$, the GNN expects to learn a vector representation h_i for each node $v_i \in V$. For a k-layer GNN, its k_{th} layer can be formalized according to the message passing framework as follows:

$$m_i^{(k)} = AGG^{(k)}(\{h_j^{k-1} : j \in N_i\}) \tag{12}$$

$$h_i^{(k)} = COMB^{(k)}(\{h_i^{(k-1)}, m_i^{(k)}\}). \tag{13}$$

Here, $h_i^{(k)}$ denotes the representation of node v_i at kth layer and $h_i^{(0)} = x_i$. $m_i^{(k)}$ represents the messages received by node v_i at kth layer and N_i denotes the set of neighboring nodes of node v_i. $AGG^{(k)}$ and $COMB^{(k)}$ denote the message aggregation function and the representation updating function at kth layer, respectively, which use different function designs for different implementations. $h_i = h_i^{(K)}$ is the final representation of the node after passing through the K layers of the GNN.

In this article, a two-layer graph convolutional network is used for the specific implementation.

C. Design of Contrastive Loss Function

In this section, we present the loss function used in our approach. We denote the augmented view obtained by node removal based on VE as view 1, and the augmented view obtained by edge removal based on VE as view 2. We then utilize these two views for contrastive learning. Specifically, we input both views into a shared graph encoder to obtain node embeddings across different views. Our contrastive learning framework employs the classic InfoNCE loss, whose core principle treats embedding of the same node across different augmented views as positive pairs, while considering all others as negative samples. However, since view 1 is generated through VE-based node removal, the node set in this view constitutes a subset of the original node set, leading to a node dimension mismatch with view 2. Consequently, a node dimension mismatch issue arises between view 1 and view 2. To address this issue, a critical adaptation is made to the traditional InfoNCE loss. The adaptation involves a preprocessing step to align the nodes that are common to both views, thereby restricting the subsequent loss computation to only this aligned subset of nodes. Specifically, we define the modified loss function as follows: As shown in Fig. 5, given two views, the i_{th} node $h_i(1)$ of view 1 is selected as the anchor, and its contrastive loss $\ell(h_i(1))$ is defined as follows:

$$\ell(h_i(1)) = -\log \frac{e^{s_{ii}^{12}/\tau}}{e^{s_{ii}^{12}/\tau} + \sum_{j \neq i}(e^{s_{ij}^{11}} + e^{s_{ij}^{12}})/\tau}. \tag{14}$$

Here, $s_{ij}^{ab} = \theta(h_i(a), h_j(b)) = s(g(h_i(a)), g(h_j(b)))$ is the similarity measure function, where $s(\cdot, \cdot)$ represents the cosine similarity and $g(\cdot)$ represents a nonlinear mapping that enhances the expressive power of the discriminant function. τ is the temperature parameter to control the similarity scale. Because the two views are symmetric, the i_{th} node $h_i(2)$ of view 2 is chosen as the anchor, which can be similarly defined according to (14) for its contrastive loss $\ell(h_i(2))$. Each node in view 1 and view 2 is then selected as an anchor in turn, which ultimately obtains a total contrastive loss is defined as follows:

$$J = \frac{1}{2M} \sum_{i=1}^{M} [\ell(h_i(1)) + \ell(h_i(2))]. \tag{15}$$

where M denotes the subset of retained nodes.

Finally, the encoder is trained by the designed loss function to optimize the parameters. Specifically, a gradient descent optimization algorithm [44] is used to minimize the loss function so as to optimize the parameters of the encoder. The final encoder obtained will have better node representation.

D. The VEGCL Algorithm

In this section, we present the pseudocode of the VEGCL algorithm as shown in Algorithm 1.

Algorithm 1: VEGCL Algorithm

Input: Graph G
Output: The trained GNN encoder $f(\cdot)$
1 **Initialize** GNN encoder $f(\cdot)$;
2 Compute the node importance and edge importance using VE;
3 **for** *epoch* $\leftarrow 1, 2, \cdots$ **do**
4 **for** *each nodes or edges in graph* **do**
5 **if** *node importance* $<$ *threshold* **then**
6 Remove the node from the graph;
7 Update the augumented view $\widetilde{G}_1$;
8 **if** *edge importance* $<$ *threshold* **then**
9 Remove the edge from the graph;
10 Update the augumented view $\widetilde{G}_2$;
11 **for** *each augumented view* **do**
12 The node representation vector is obtained using the encoder $f(\cdot)$
13 Calculate the contrastive loss J using Eq.(15);
14 Update the encoder parameters using a gradient descent optimization algorithm to minimize J.
15 **return** $f(\cdot)$

The algorithm first initializes the GNN encoder $f(\cdot)$ and then calculates the node importance according to (4) and the edge importance according to (8) (line 1-line 2). This is then used as a basis to generate two augmented views $\widetilde{G}_1$ and $\widetilde{G}_2$ by removing unimportant nodes and edges from the graph (line 4- line 10). These two augmented views are then input into the shared GNN encoder $f(\cdot)$ to obtain the corresponding node representation vectors, respectively (line 11-line 12). After that, the contrast loss J is computed according to (15) (line 13). Then, the model is trained by minimizing the contrast loss J until the model becomes converged (line 14). Finally, return the trained GNN encoder $f(\cdot)$ (line 15).

V. Experiments

This section first studies the effectiveness of VE for evaluating the importance of nodes in the input graph. The proposed algorithm is then applied to the node classification task to evaluate its performance.

A. Experiment Setup

1) Experimental Environment: The server is the Dell PowerEdge R740; the CPU is 5220R 2.2G, 24C/48T (24 core 48 processes) $*$ 2; the GPU is NVIDIA GRX 3090; the memory is 384G; the operating system is Ubuntu 18.04.6, Python 3.8.8, Pytorch 1.8.1, torch-geometric 2.0.1, cdlib 0.2.6, networkx 2.5.1, and numpy 1.22.4.

2) Datasets: We select four citation network datasets including Cora, CiteSeer, Coauthor-CS and PubMed[1] along with one social network dataset called LastFM Asia[2] for the experiments.

TABLE II
Details of Datasets

Dataset	Nodes	Edges	Features	Classes
Cora	2708	10 556	1433	7
CiteSeer	3327	9104	3703	6
Coauthor-CS	18 333	163 788	6805	15
PubMed	19 717	88 648	500	3
LastFM Asia	7624	27 806	7842	18

Table II shows the detailed statistical information of the datasets. It includes the number of nodes, the number of edges, the number of features, and the number of classes.

3) Evaluation Metrics: We use accuracy (ACC) and F1 score to evaluate the performance of node classification.

a) *ACC* is defined as follows [45].

$$ACC = \frac{TP + TN}{TP + FP + FN + TN} \tag{16}$$

Here, TP denotes the number of positive classes predicted as positive, which is the number of correctly predicted positive samples, FN denotes the number of positive classes predicted as negative, which is the number of incorrectly predicted positive samples, FP denotes the number of negative classes predicted as positive, which is the number of incorrectly predicted negative samples, and TN denotes the number of negative classes predicted as negative, which is the number of correctly predicted negative samples. The higher the ACC, the more effective the model is.

b) *F1 Score* is defined as follows [46].

$$F1 = \frac{2 * P * R}{P + R} \tag{17}$$

Here, $P = TP/(TP + FP)$ and $R = TP/(TP + FN)$.

B. Comparison System

To validate the effectiveness of VEGCL, we conduct comprehensive comparative experiments. The selected benchmark methods include three categories: random augmentation methods, learning-based augmentation or architectural innovation methods, and importance-driven augmentation methods.

1) Random Augmentation Methods:

a) *GRACE* [17] generates two augmented views by randomly removing edges and randomly masking node attributes, and then trains the model by using the contrastive loss function to maximize the consistency of node embeddings in the two views.

b) *gCooL* [9] generates two views through random data augmentation, then simultaneously employs both node-level and community-level contrastive learning to learn node representations.

[1]https://github.com/shchur/gnn-benchmark/raw/master/data/npz/

[2]https://snap.stanford.edu/data/feather-lastfm-social.html

c) *BGRL* [26] generates augmented views through random augmentation and learns node representations by predicting the output of the target encoder via an online encoder, which eliminates the need for negative samples.

2) Learning-Based Augmentation or Architectural Innovation Methods:

a) *MA-GCL* [10] constructs contrastive views by designing an asymmetric, randomized, and shuffled graph encoder architecture.

b) *GAug* [22] learns graph class homogeneity through the graph auto-encoder to guide edge addition and deletion operations based on class homogeneity.

c) *CTAug* [8] improves the performance of GCL by introducing a preservation mechanism for cohesive subgraphs during topology augmentation and graph encoding.

3) Importance-Driven Augmentation Methods:

a) *GCA* [34] generates two augmented views by removing insignificant edges and masking insignificant node attributes, and then the model is trained using a contrastive loss function to maximize the consistency of the node embeddings in the two views.

b) *CSGCL* [12] takes into account the community structure and the variability of the influence of different communities on the whole network. It adopts a graph node and edge augmentation strategy based on community strength to generate two augmented views. And then the similarity between each node in the InfoNCE loss function is gradually fine-tuned according to the community strength to compose the final loss function to train the model.

c) *DCMSL* [14] considers the differences between both community strength and nodes to mask node attributes and remove edges to generate augmented views. Then a multiscale graph contrastive loss function is designed, which conducts contrastive learning from two different perspectives of nodes and communities.

d) *VEGCL* is our proposed method which generates two augmented views by removing insignificant nodes and edges using a graph data augmentation based on VE method, and then trains the model using the InfoNCE loss function to maximize the consistency of the node embeddings in the two views.

e) *VEGCL-Renyi Entropy (VEGCL-RE)* is a variant of our proposed method, and the difference with our proposed method is that the method measures the importance of nodes and edges by utilizing the effect of local changes in individual nodes on the Renyi entropy of the network. Renyi entropy is defined as follows:

$$R_\alpha(G) = \frac{1}{1-\alpha}\log_2\left(tr(\rho^\alpha)\right) \tag{18}$$

where tr denotes the trace of the matrix, parameter $\alpha \in (0,1)\cup(1,\infty)$, G denotes the original network. $R_\alpha(G)$ denotes the Renyi entropy of the original network, ρ denotes the density matrix of the network G and $\rho = (e^{-\tau L}/Z)$, $Z = tr(e^{-\tau L}) = \sum_{i=1}^{N} e^{-\tau\lambda_i(L)}$ is the distribution function of the system, and the parameter τ encodes the diffusion propagation time. $e^{-\tau L}$ is the diffusion propagator, $L = D - A$ is the Laplacian matrix, D is the degree matrix of the network G, and A is the neighborhood matrix of the network G.

C. Experimental Results

1) Exp1: The Effectiveness of VE: We use the Pearson correlation coefficient (PCC) to quantify the linear correlation between degree centrality, betweenness centrality, closeness centrality, and VE. Its formula is defined as follows:

$$r = \frac{\sum_{i=1}^{n}[(X_i-\bar{X})(Y_i-\bar{Y})]}{\sqrt{\sum_{i=1}^{n}[(X_i-\bar{X})^2}\sqrt{\sum_{i=1}^{n}[(Y_i-\bar{Y})^2}} \tag{19}$$

where X and Y represent two variables respectively, n is the number of samples, X_i and Y_i are the observations at point i corresponding to variable X and Y, $\bar{X}$ is the average of X samples and $\bar{Y}$ is the average of Y samples.

The range of values for PCC is $[-1, 1]$. Negative values mean that as one variable increases the other decreases, positive values mean that as one variable increases so does the other, and 0 means that the increase or decrease of one variable has no effect on the value of the other.

Fig. 6(a)–(e) present the correlation analysis results of four centrality metrics across five datasets. Overall, VE exhibits relatively low correlations with traditional centrality metrics, indicating that its global perspective based on spectral entropy perturbations can capture novel information beyond what is captured by traditional metrics focusing on local connectivity or path statistics. However, this correlation varies across different datasets. Specifically, in networks with high clustering coefficients such as Cora, VE shows low correlations with all three traditional centrality measures, and the nodes identified with high VE values are primarily distributed at bridge positions between communities rather than being highly connected nodes with dense local links. In contrast, in networks like CiteSeer with degree distributions closer to power law distributions and stronger structural heterogeneity, VE demonstrates high correlations with degree and betweenness centrality, while maintaining a low correlation with closeness centrality. This discrepancy primarily stems from network topological heterogeneity. VE essentially measures the impact of nodes on network robustness, multiscale information propagation efficiency, and global spectral properties, rather than merely assessing local connection density or shortest path control capability. Consequently, in highly modular and structurally uniform networks, VE distinctly separates from traditional metrics. In networks with power law heterogeneity, although VE partially overlaps with degree and betweenness centrality in measurement, it retains its independent analytical perspective, providing a unique supplement to the evaluation of node importance in networks. This conclusion is intuitively validated by the visualization of node spatial distribution. Taking the Cora dataset as an example, Fig. 6(f)–(i) illustrate the node importance distributions of degree centrality, betweenness centrality, closeness centrality, and

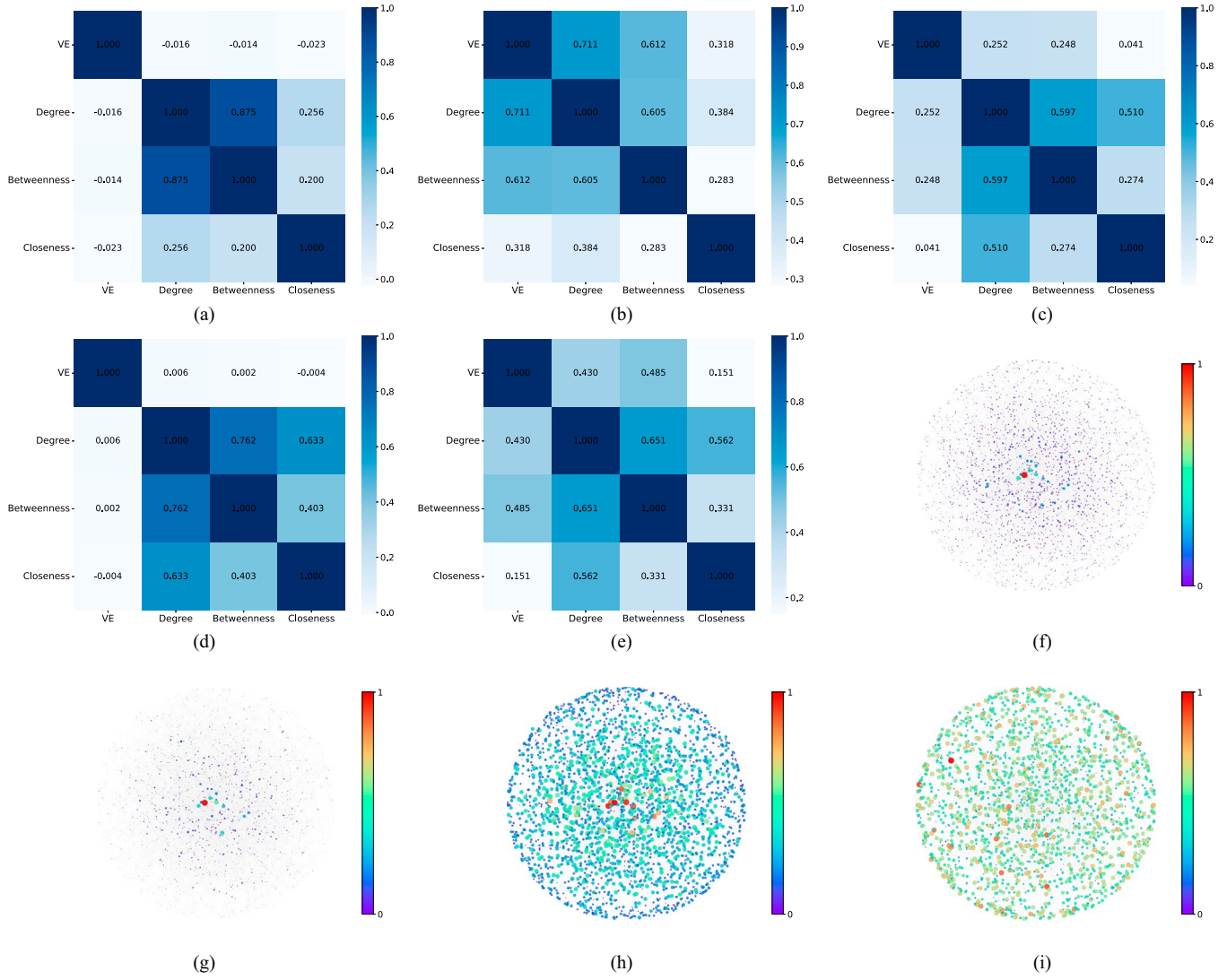

Fig. 6. Correlation analysis results: (a)–(e) present the PCC heat map between degree centrality, betweenness centrality, closeness centrality, and VE across five datasets; and (f)–(i) illustrate the visualization of node importance distribution of degree centrality, closeness centrality, betweenness centrality, and VE on the Cora dataset. (a) Cora. (b) CiteSeer. (c) PubMed. (d) Coauthor-CS. (e) LastFMAsia. (f) Degree. (g) Betweenness. (h) Closeness. (i) VE.

VE, respectively. As shown in Fig. 6(f)–(h), key nodes identified by traditional centrality metrics tend to be closely connected to each other, which form clusters in specific regions of the network. In contrast, the VE in Fig. 6(i) reveals a completely different pattern in that the important nodes are not clustered in local regions but are evenly distributed across the network. This significant difference in spatial distribution further corroborates that VE can identify structural bridge nodes that maintain network connectivity on a global scale, rather than merely discovering core nodes in locally dense regions. This provides compelling visual evidence for the core conclusion that VE offers a novel perspective independent of traditional centrality metrics.

2) Exp2: Node Classification: We experimentally validated our proposed method on five datasets and evaluated the performance of our method by two metrics, including ACC and F1 score. The results are shown in Tables III and IV.

TABLE III
ACC OF NODE CLASSIFICATION

	Cora	PubMed	CiteSeer	Coauthor-CS	LastFM Asia
GCA	79.9	84.9	70.2	90.5	77.5
GRACE	78.3	85.1	55.7	90.2	80.9
CSGCL	74.8	84.4	69.3	90.3	79.9
DCMSL	80.9	83.2	63.4	91.2	73.1
gCooL	79.9	84.2	68.3	91.3	78.5
GAug	81.7	77.6	67.7	90.9	79.9
BGRL	73.1	81.7	60.6	90.5	71.8
MA-GCL	82.6	84.8	69.8	91.2	79.4
CTAug	82.3	85.2	70.7	91.4	81.0
VEGCL-RE	82.7	85.9	71.2	90.8	83.1
VEGCL	**83.1**	**86.2**	**71.5**	**91.5**	**83.3**

Note: The bold represents the best ACC for each dataset.

According to the results in Tables III and IV, the VEGCL algorithm shows excellent performance in node classification tasks on different datasets. Our method significantly outperforms other baseline methods in overall performance.

TABLE IV
F1 OF NODE CLASSIFICATION

	Cora	PubMed	CiteSeer	Coauthor-CS	LastFM Asia
GCA	79.94	83.77	69.02	90.46	62.05
GRACE	77.94	83.26	57.78	89.37	68.43
CSGCL	77.95	84.04	64.21	89.04	77.33
DCMSL	80.77	82.61	63.35	**91.22**	70.08
gCooL	78.69	83.84	64.01	89.07	64.57
GAug	80.94	77.46	64.76	87.69	64.56
BGRL	71.56	81.31	55.80	87.71	55.06
MA-GCL	79.81	84.34	62.25	88.47	63.37
CTAug	80.57	84.67	64.13	88.61	63.85
VEGCL-RE	80.24	84.39	68.87	90.28	82.03
VEGCL	**81.04**	**84.78**	**69.06**	90.53	**82.33**

Note: The bold represents the best F1 for each dataset.

Specifically, the VEGCL improves on the ACC by 0.5% and on the F1 score by 0.1% compared to the previously optimal method on the Cora dataset. Similarly, the VEGCL improves 1% in ACC and 0.11% in F1 score on the PubMed dataset. The results also show that the VEGCL improves 0.8% in ACC and a slight increase of 0.04% in F1 score on the CiteSeer dataset. In addition, on the LastFM Asia dataset, the VEGCL improves ACC and F1 score by 2.3% and 5%, respectively. On the Coauthor-CS dataset, the VEGCL algorithm increases the ACC by 0.1%. However, the DCMSL algorithm performs slightly better on the F1 score, which may be due to the category imbalance in the Coauthor-CS dataset. Specifically, the reason why the DCMSL algorithm performs better on the F1 score may be due to the fact that it has a higher sensitivity to those fewer category nodes in the dataset, and thus is able to more accurately identify these nodes, despite the fact that they make up a smaller percentage of the overall dataset.

These results consistently demonstrate the robustness and effectiveness of the VEGCL algorithm in improving node classification ACC and F1 score. In addition, our method also slightly outperforms the variant method VEGCL-RE in these two metrics, which is probably due to the fact that our method employs the effect of local changes in individual nodes on the spectral entropy of the network to measure the importance of nodes and edges. Among them, spectral entropy serves as a global metric that captures the topological features of the entire network. This global perspective is crucial for identifying key nodes in the network and helps to understand and analyze the network structure more comprehensively.

3) Exp3: Parameter Analysis: Without loss of generality, we conduct experiments on two datasets of Cora and CiteSeer to explore in depth the specific impact of these parameters on the performance of the node classification task by adjusting some key parameters. Based on the experimental results on these two datasets, performance trends on other datasets can be predicted. The experimental results are shown in Fig. 7.

As can be seen from Fig. 7, the node classification performance is worse for datasets of larger size. Therefore, it can be predicted that the node classification performance of the PubMed, Coauthor-CS and LastFM Asia datasets is even lower because these three datasets are larger in size. In Fig. 7, the results in (a) show that the Cora dataset achieves the highest ACC when the edge removal threshold is set to 0.8, while the

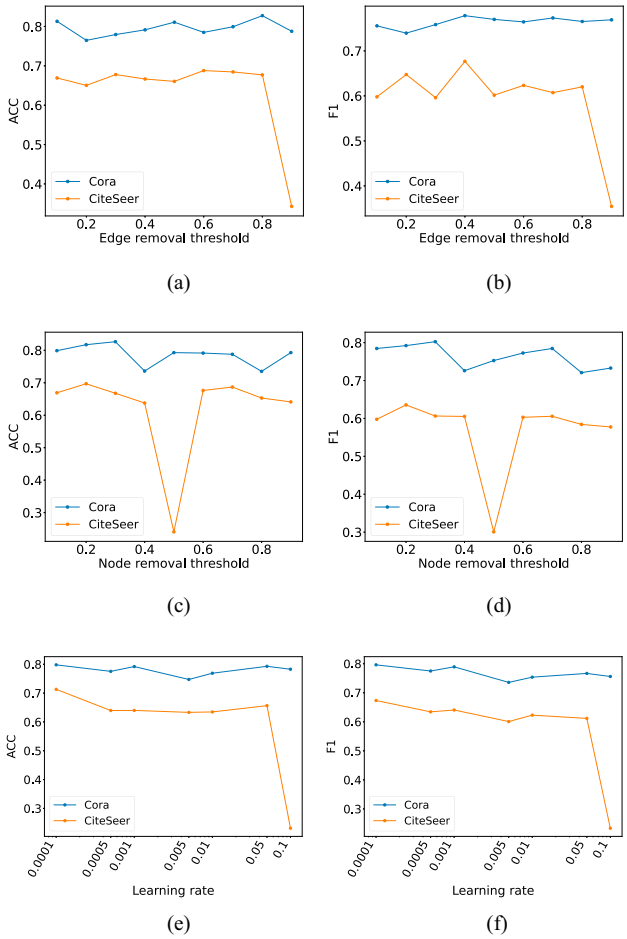

Fig. 7. Influence of different parameters on model ACC and F1 score: (a) influence of different edge removal threshold on model ACC; (b) influence of different edge removal threshold on model F1; (c) influence of different node removal threshold on model ACC; (d) influence of different node removal threshold on model F1; (e) influence of different learning rate on model ACC; and (f) influence of different learning rate on model F1.

CiteSeer dataset performs best when the threshold is about 0.6. This indicates that the edge removal threshold has a significant effect on the model ACC for different datasets. Next, the results of (b) indicate that the F1 score is optimized when the edge removal threshold is about 0.4 on both Cora and CiteSeer datasets. This finding emphasizes the importance of an appropriate edge removal threshold in improving the F1 score of the model. From (a) and (b), the performance of node classification drops extremely fast when the edge removal threshold of CiteSeer dataset reaches 0.9, which is attributed to the fact that too high deletion probability may destroy important structural features in the graph. After that, the results in (c) and (d) show that the Cora dataset has the highest ACC and F1 score at a node removal threshold of about 0.3, while the CiteSeer dataset performs best at a threshold of about 0.2. This discrepancy primarily stems from the inherent differences in the topological characteristics of the two datasets. The relatively sparse structure of CiteSeer makes it more sensitive to node removal, allowing it to enhance generalization capability by appropriately removing unimportant nodes around the

threshold of 0.2. However, when the threshold increases to 0.5, the removal of a large number of nodes disrupts its fragile community structure, making it difficult for the model to effectively distinguish nodes from different academic domains and leading to a significant decline in performance. Notably, when the threshold exceeds 0.5, although the network structure becomes extremely sparse, the model is forced to learn more generalized feature representations, which to some extent mitigates the overfitting problem. Observing the results from (a) to (d), it can be observed that different datasets have different sensitivities to node and edge deletion. Among them, the Cora dataset maintains sufficient structural information after deleting more nodes or edges, which maintains the classification performance. Finally, the results in (e) and (f) show that when the learning rate is set to 0.0001, the model achieves the highest ACC and F1 score on both the Cora and CiteSeer datasets. However, the Cora dataset maintains relatively stable performance across different learning rates, whereas the CiteSeer dataset exhibits significant fluctuations, with a sharp performance drop specifically at a learning rate of 0.1. We argue that the performance fluctuations in CiteSeer can be attributed to its inherent dataset characteristics, such as a sparser graph structure and a more imbalanced class distribution. These factors collectively make the model performance more sensitive to changes in the learning rate.

In conclusion, our method is relatively sensitive to hyperparameters. To help researchers quickly deploy this method in new datasets, we recommend a cascaded hyperparameter tuning strategy. This strategy first identifies a promising range of parameters through broad grid search, and then employs Bayesian optimization for precise iteration within this range to achieve a balance between efficiency and ACC.

4) Exp4: Ablation Studies: We perform ablation studies on key module of VEGCL. '-NR' denotes the generation of augmented views by node removal only, and '-ER' denotes the generation of augmented views by edge removal only.

The results are shown in Tables V and VI, where we can see that the scheme using both node removal and edge removal improves the model performance on all datasets. We attribute this performance improvement to the complementary nature of the two augmentation strategies, which collectively promote a more robust representation learning process. Specifically, '-NR' enhances robustness by enabling the model to focus on the relational structure among core nodes through the removal of unimportant nodes in the graph. '-ER' achieves this by removing redundant or noisy edges, allowing the model to concentrate on the core structural relationships between nodes. This reduces interference from irrelevant information and improves the quality of node representations. By combining these two strategies, VEGCL generates more diverse augmented views for contrastive learning. This approach not only overcomes the limitations of a single augmentation strategy but also guides the model to learn node embeddings with dual invariance which are stable against both unimportant edge perturbations introduced by '-ER' and unimportant node perturbations caused by '-NR'. Ultimately, the model learns more robust node representations, leading to significantly enhanced performance on downstream tasks.

TABLE V
ABLATION STUDY ON NODE CLASSIFCATION IN ACC

	Cora	PubMed	CiteSeer	Coauthor-CS	LastFM Asia
VEGCL-NR	82.3	85.4	67.3	89.6	79.8
VEGCL-ER	81.4	84.7	68.5	90.7	81.3
VEGCL	**83.1**	**86.2**	**71.5**	**91.5**	**83.3**

Note: The bold represents the best ACC for each dataset.

TABLE VI
ABLATION STUDY ON NODE CLASSIFCATION IN F1

	Cora	PubMed	CiteSeer	Coauthor-CS	LastFM Asia
VEGCL-NR	80.67	83.19	66.39	88.53	78.44
VEGCL-ER	80.81	82.95	62.60	89.82	80.20
VEGCL	**81.04**	**84.78**	**69.06**	**90.53**	**82.33**

Note: The bold represents the best F1 for each dataset.

D. Complexity Analysis

The computational complexity of VE primarily arises from the following stages: First, constructing the adjacency matrix and the Laplacian matrix L of the network has a time complexity of $O(N^2)$, where N represents the total number of nodes. Subsequently, performing the eigenvalue decomposition of the L to obtain its eigenvalues and eigenvectors with a time complexity of $O(N^3)$. The spectral entropy calculation is based on the density matrix which has a complexity of $O(N)$. Next, to evaluate the importance of each node v, we compute its spectral properties on the perturbed network. The computational complexity for a single node is $O(k_v^2 N)$, where k_v denotes the degree of node v. Therefore, the overall complexity for all nodes is $\sum_v O(k_v^2 N) = O(N \cdot \sum_v k_v^2) = O(N^2\langle k^2\rangle)$, where $\langle k^2\rangle$ is the average of the squared degrees. In summary, the total computational complexity of VE is $O(N^2) + O(N^3) + O(N) + O(N^2\langle k^2\rangle) \approx O(N^3) + O(N^2\langle k^2\rangle)$.

VI. CONCLUSION

In this study, we propose the VEGCL, whose core innovation lies in utilizing VE to guide graph data augmentation. This method effectively preserves critical semantic information in the graph while reducing redundant structural interference. Experiments on multiple benchmark datasets demonstrate that VEGCL outperforms baseline methods in node classification tasks. This work not only enhances the performance of GCL in citation networks but also establishes a methodological foundation for broader social intelligence tasks, such as user behavior analysis, public opinion propagation prediction, and community evolution modeling. However, the VEGCL also has some limitations. For example, it relies on a fixed data augmentation threshold that requires manual adjustment, which may affect its robustness and generalization across different datasets. Therefore, future research can focus on constructing an adaptive threshold mechanism. Potential directions include formulating the threshold as a learnable parameter to be optimized jointly with the model via gradient descent, or adopting meta-learning

strategies to dynamically predict optimal thresholds based on the characteristics of the input graph. Furthermore, exploring VE-based weighting schemes as an alternative to hard removal could better preserve structural information.

Data Availability Statement

The source code is available at https://github.com/xxl-1237/VEGCL.git.

References

[1] M. Rodríguez-Ibánez, A. Casánez-Ventura, F. Castejón-Mateos, and P.-M. Cuenca-Jiménez, "A review on sentiment analysis from social media platforms," *Expert Syst. Appl.*, 2023, Art. no. 119862.

[2] Z. Deng, Y. Dong, and J. Zhu, "Batch virtual adversarial training for graph convolutional networks," *AI Open*, vol. 4, pp. 73–79, 2023.

[3] Y. Fang, X. Tang, L. Shang, D. Fan, D. Zha, and Q. Tan, "LLM4GCL: Can large language model em-power graph contrastive learning?" 2023. [Online]. Available: https://openreview.net/forum?id=wxClzZdjqP

[4] J. Yu, X. Xia, T. Chen, L. Cui, N. Q. V. Hung, and H. Yin, "XSimGCL: Towards extremely simple graph contrastive learning for recommendation," *IEEE Trans. Knowl. Data Eng.*, vol. 36, no. 2, pp. 913–926, Feb. 2024.

[5] Y. Jiang, C. Huang, and L. Huang, "Adaptive graph contrastive learning for recommendation," in *Proc. 29th ACM SIGKDD Conf. Knowl. Dis. Data Min.*, 2023, pp. 4252–4261.

[6] S. Li, W. Li, A. M. Luvembe, and W. Tong, "Graph contrastive learning with feature augmentation for rumor detection," *IEEE Trans. Computat. Social Syst.*, vol. 11, no. 4, pp. 5158–5167, Aug. 2024.

[7] Y. Qian, Y. Zhang, N. Chawla, Y. Ye, and C. Zhang, "Malicious repositories detection with adversarial heterogeneous graph contrastive learning," in *Proc. 31st ACM Int. Conf. Inf. & Knowl. Manage.*, 2022, pp. 1645–1654.

[8] Y. Wu, L. Wang, X. Han, and H.-J. Ye, "Graph contrastive learning with cohesive subgraph awareness," in *Proc. ACM Web Conf.*, 2024, pp. 629–640.

[9] B. Li, B. Jing, and H. Tong, "Graph communal contrastive learning," in *Proc. ACM Web Conf.*, 2022, pp. 1203–1213.

[10] X. Gong, C. Yang, and C. Shi, "MA-GCL: Model augmentation tricks for graph contrastive learning," in *Proc. AAAI Conf. Artif. Intell.*, vol. 37, no. 4, pp. 4284–4292, 2023.

[11] K. Hassani and A. H. Khasahmadi, "Contrastive multi-view representation learning on graphs," in *Int. Conf. Mach. Learn*. PMLR, 2020, pp. 4116–4126.

[12] H. Chen, Z. Zhao, Y. Li, Y. Zou, R. Li, and R. Zhang, "CS-GCL: Community-strength-enhanced graph contrastive learning," 2023, *arXiv:2305.04658*.

[13] Y. You, T. Chen, Z. Wang, and Y. Shen, "Bringing your own view: Graph contrastive learning without prefabricated data augmentations," in *Proc. 15th ACM Int. Conf. Web Search Data Mining*, 2022, pp. 1300–1309.

[14] H. Chen, Y. Li, P. S. Yu, Y. Zou, and R. Li, "DCMSL: Dual influenced community strength-boosted multi-scale graph contrastive learning," *Knowl.-Based Syst.*, vol. 304, 2024, Art. no. 112472.

[15] J. Wu, et al., "Self-supervised graph learning for recommendation," in *Proc. 44th Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2021, pp. 726–735.

[16] J. Yu, H. Yin, X. Xia, T. Chen, L. Cui, and Q. V. H. Nguyen, "Are graph augmentations necessary? simple graph contrastive learning for recommendation," in *Proc. 45th Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2022, pp. 1294–1303.

[17] Y. Zhu, Y. Xu, F. Yu, Q. Liu, S. Wu, and L. Wang, "Deep graph contrastive representation learning," 2020, *arXiv:2006.04131*.

[18] Y. Yin, Q. Wang, S. Huang, H. Xiong, and X. Zhang, "AutoGCL: Automated graph contrastive learning via learnable view generators," in *Proc. AAAI Conf. Artif. Intell.*, vol. 36, no. 8, pp. 8892–8900, 2022.

[19] X. Yang et al., "Convert: Contrastive graph clustering with reliable augmentation," in *Proc. 31st ACM Int. Conf. Multimedia*, 2023, pp. 319–327.

[20] J. Shuai et al., "A review-aware graph contrastive learning framework for recommendation," in *Proc. 45th Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2022, pp. 1283–1293.

[21] J. Huang et al., "Adversarial learning data augmentation for graph contrastive learning in recommendation," in *Proc. International Conference on Database Systems for Advanced Applications*. Berlin, Germany: Springer, 2023, pp. 373–388.

[22] T. Zhao, Y. Liu, L. Neves, O. Woodford, M. Jiang, and N. Shah, "Data augmentation for graph neural networks," in *Proc. AAAI Conf. Artif. Intell.*, 2021, vol. 35, no. 12, pp. 11015–11023.

[23] A. Ghavasieh, M. Stella, J. Biamonte, and M. De Domenico, "Unraveling the effects of multiscale network entanglement on empirical systems," *Commun. Phys.*, vol. 4, no. 1, p. 129, 2021.

[24] Y. Huang, H. Wang, X.-L. Ren, and L. Lü, "Identifying key players in complex networks via network entanglement," *Commun. Phys.*, vol. 7, no. 1, p. 19, 2024.

[25] X. Cai, C. Huang, L. Xia, and X. Ren, "LightGCL: Simple yet effective graph contrastive learning for recommendation," 2023, *arXiv:2302.08191*.

[26] S. Thakoor et al., "Large-scale representation learning on graphs via bootstrapping," in *Int. Conf. Learn. Representations*, 2022.

[27] H. Li, J. Cao, J. Zhu, Q. Luo, S. He, and X. Wang, "Augmentation-free graph contrastive learning of invariant-discriminative representations," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 35, no. 8, pp. 11157–11167, Aug. 2024.

[28] D. Zou et al., "Multi-level cross-view contrastive learning for knowledge-aware recommender system," in *Proc. 45th Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2022, pp. 1358–1368.

[29] Y. Xie, J. Jia, C. Wen, D. Li, and M. Li, "Multi-topology contrastive graph representation learning," *Sci. China Inf. Sci.*, vol. 69, no. 2, 2026, Art. no. 122102.

[30] J. Duan et al., "Graph anomaly detection via multi-scale contrastive learning networks with augmented view," in *Proc. AAAI Conf. Artif. Intell.*, 2023, vol. 37, no. 6, pp. 7459–7467.

[31] S. Qin, Y. Luo, J. Zhu, G. Tao, J. Zheng, and Z. Ma, "Anomaly detection on attributed networks via multiview and multiscale contrastive learning," *IEEE Trans. Computat. Social Syst.*, 2024.

[32] Y. Liu et al., "Simple contrastive graph clustering," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 35, no. 10, pp. 13789–13800, Oct. 2023.

[33] J. Xia, L. Wu, G. Wang, J. Chen, and S. Z. Li, "ProGCL: Rethinking hard negative mining in graph contrastive learning," 2021, *arXiv:2110.02027*.

[34] Y. Zhu, Y. Xu, F. Yu, Q. Liu, S. Wu, and L. Wang, "Graph contrastive learning with adaptive augmentation," in *Proc. Web Conf.*, 2021, pp. 2069–2080.

[35] C. Niu, G. Pang, and L. Chen, "Affinity uncertainty-based hard negative mining in graph contrastive learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 35, no. 9, pp. 11681–11691, Sep. 2024.

[36] Z. Lin, C. Tian, Y. Hou, and W. X. Zhao, "Improving graph collaborative filtering with neighborhood-enriched contrastive learning," in *Proc. ACM Web Conf.*, 2022, pp. 2320–2329.

[37] X. Shen, D. Sun, S. Pan, X. Zhou, and L. T. Yang, "Neighbor contrastive learning on learnable graph augmentation," in *Proc. AAAI Conf. Artif. Intell.*, 2023, vol. 37, no. 8, pp. 9782–9791.

[38] S. Zhang et al., "M-Mix: Generating hard negatives via multi-sample mixing for contrastive learning," in *Proc. 28th ACM SIGKDD Conf. Knowl. Discovery Data Min.*, 2022, pp. 2461–2470.

[39] S. Lin et al., "Prototypical graph contrastive learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 35, no. 2, pp. 2747–2758, 2024.

[40] H. Tang, G. Zhao, Y. Wu, and X. Qian, "Multisample-based contrastive loss for top-K recommendation," *IEEE Trans. Multimedia*, vol. 25, pp. 339–351, 2021.

[41] Y. Zhu, Y. Xu, H. Cui, C. Yang, Q. Liu, and S. Wu, "Structure-enhanced heterogeneous graph contrastive learning," in *Proc. SIAM Int. Conf. Data Mining (SDM)*, 2022, pp. 82–90.

[42] M. De Domenico and J. Biamonte, "Spectral entropies as information-theoretic tools for complex network comparison," *Phys. Rev. X*, vol. 6, no. 4, 2016, Art. no. 041062.

[43] G. Yang, M. Li, H. Feng, and X. Zhuang, "Deeper insights into deep graph convolutional networks: Stability and generalization," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 48, no. 2, pp. 1707–1719, Feb. 2026.

[44] K. Chandra, A. Xie, J. Ragan-Kelley, and E. Meijer, "Gradient descent: The ultimate optimizer," in *Advances in Neural Information Processing System*, vol. 35, pp. 8214–8225, 2022.

[45] M. Mimura, "Impact of benign sample size on binary classification accuracy," *Expert Syst. Appl.*, vol. 211, 2023, Art. no. 118630.

[46] K. Takahashi, K. Yamamoto, A. Kuchiba, and T. Koyama, "Confidence interval for micro-averaged F1 and macro-averaged F1 scores," *Appl. Intell.*, vol. 52, no. 5, pp. 4961–4972, 2022.

Xinlan Xu received the B.Sc. degree in computer science and technology from Xihua University, Chengdu, China, in 2024. She is currently working toward the M.Sc. degree in computer science and technology with the School of Artificial Intelligence and Computer Science, Shaanxi Normal University, Xi'an, China.

Her research interests include social networks, data mining, and graph representation learning.

Fei Hao received the Ph.D. degree in computer science and engineering from Soonchunhyang University, Asan, South Korea, in 2016.

From 2020 to 2022, he was a Marie Curie Fellow with the University of Exeter, Exeter, U.K. He is currently a Professor with the School of Artificial Intelligence and Computer Science, Shaanxi Normal University, Xi'an, China. He has published more than 200 papers in the leading international journals and conference proceedings, such as IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, IEEE TRANSACTIONS ON SERVICES COMPUTING, IEEE TRANSACTIONS ON NETWORK SCIENCE AND ENGINEERING, *IEEE Communications Magazine*, IEEE INTERNET COMPUTING, *ACM Transactions on Multimedia Computing, Communications, and Applications*, ACM SPECIAL INTEREST GROUP ON INFORMATION RETRIEVAL, and IEEE GLOBAL COMMUNICATIONS CONFERENCE. His research interests include social computing, soft computing, big data analytics, pervasive computing, and data mining. He is a member of ACM, CCF, and KIPS.

Dr. Hao was a recipient of the Best Paper Award from IEEE GreenCom 2013, the Outstanding Service Award at IEEE DSS 2018, the IEEE SmartData 2017, the IEEE Outstanding Leadership Award at IEEE CPSCom 2013, and the 2015 Chinese Government Award for Outstanding Self-Financed Students Abroad.

Bo Li received the B.S. degree from Shaanxi University of Technology, Hanzhong, China, in 2006, the M.S. degree from Xihua University, Chengdu, China, in 2009, and the Ph.D. degree from Southwest Jiaotong University, Chengdu, China, in 2019, all in computer science.

He is currently an Associate Professor with the School of Computer and Software Engineering, Xihua University. His research interests include data mining and machine learning.

Hongying Zhang received the M.Sc. and Ph.D. degrees in applied mathematics from Xi'an Jiaotong University, Xi'an, China, in 2003 and 2008, respectively.

She is currently a Professor with the School of Mathematics and Statistics, Xi'an Jiaotong University. Her research interests include artificial intelligence, granular computing, and machine learning.

Wangyang Yu received the M.S. degree from Shandong University of Science and Technology, Qingdao, China, in 2009, and the Ph.D. degree from Tongji University, Shanghai, China, in 2014, both in computer software and theory.

He is currently an Associate Professor with the School of Artificial Intelligence and Computer Science, Shaanxi Normal University, Xi'an, China. From 2016 to 2017, he was also a Visiting Scholar with the University of Derby, Derby, U.K. His research interests include theory of Petri nets, formal methods in software engineering and trustworthy software.

Geyong Min (Member, IEEE) received the B.Sc. degree from Huazhong University of Science and Technology, Wuhan, China, in 1995, and the Ph.D. degree from the University of Glasgow, Glasgow, U.K., in 2003, both in computing science.

He is a Professor of high performance computing and networking with the Department of Computer Science, College of Engineering, Mathematics and Physical Sciences, University of Exeter, Exeter, U.K. His research interests include next generation internet, wireless communications, multimedia systems, information security, ubiquitous computing, modelling, and performance engineering.