Graph Classification via Hierarchical Structure Learning with Multi-Scale Convolution and Pooling

Wei Liu and Qiang Guo

***Abstract*—Graph classification is a fundamental task in the field of machine learning with applications ranging from bioinformatics to social network analysis. However, existing methods suffer from excessive computational complexity and information loss. To address these issues, in this paper, we propose a multi-scale structural learning model for graph classification named MSSGC, which consists of two phases: a pooling phase and a classification phase. In the pooling phase, we leverage multi-scale information aggregation to retain more original graph features while reducing the computational complexity. This is achieved through a multi-scale convolution technique called MSConv, which efficiently passes graph messages and incorporates structural features into node representations. By selecting the most informative nodes in the coarsened graph, we generate discriminative graph representations that effectively utilize semantic and topological information. In the classification phase, MSSGC can employ any off-the-shelf classifiers to predict graph labels. Extensive experimental results on several standard benchmarks demonstrate that the proposed MSSGC outperforms many state-of-the-art graph classification methods.**

***Index Terms*—graph classification, graph structure learning, multi-scale convolution, representation learning.**

I. Introduction

GRAPH-structured data represents entities and their relationships as nodes and edges in a graph. Due to their ability to capture complex network interactions, graphs are widely used in applications such as social media analysis, biological network interpretation, and routing optimization in transportation networks. Graph classification aims to categorize data with complex structures composed of nodes and edges, which holds significant importance in various domains [1]–[5]. For instance, in the field of bioinformatics, graph classification aids in the identification of mutagenicity, toxicity, and anticancer activity of compound molecules by categorizing molecular graphs [6]. Similarly, it plays a vital role in predicting whether a protein is an enzyme by classifying protein networks [7]. However, graph-structured data differs from regularly linked structures in text and image data, rendering the common CNNs-based and RNNs-based methods [8]–[10] is unsuitable for direct application to graph structure data. The fundamental challenge in graph classification lies in effectively modeling and capturing the complex and diverse information inherent in the data, which demands specialized techniques that can appropriately handle and learn from such complexity.

Existing graph classification methods typically apply layer-by-layer pooling techniques to aggregate information from multiple higher-level structures, thereby generating a graph representation. Two prominent types of hierarchical pooling approaches have been presented in the literature: (1) Graph collapse pooling, which maps each node to multiple clusters in a probabilistic manner. This means that every node in the graph is associated with a set of probabilities, indicating the likelihood of that node belonging to a particular cluster. Such soft assignment allows for a more nuanced understanding of the node's role and connectivity in the graph. Motivated by this, the proposed method can effectively collapse the graph structure, facilitating the extraction of more abstract representations. (2) Top-K pooling, which utilizes a score function to determine the importance scores of individual nodes. By selecting the top-K nodes based on their scores, it can condense the graph representation while preserving the most significant information. This hierarchical pooling plays a crucial role in enhancing graph classification by capturing essential information from the graph structures at different levels of abstraction. In the top-K pooling, a selection is made to retain the top-K percent of important nodes along with their associated connectivity relationships. Compared with the collapse pooling, this pooling operation stands out from existing pooling techniques by selectively forwarding graph structure information to subsequent layers based on original data, significantly reducing the computational burden while maintaining informative graph representations. However, the continual discarding of nodes inevitably leads to the loss of valuable information. Although compressing multiple complex topological data into a one-dimensional vector may enhance the information content of the graph representation in the top-K method, it becomes impossible to preserve the overall raw information, as achieved in graph collapse pooling.

Mitigating the limitation of the top-K pooling in terms of information loss under low-configuration conditions is an important challenge. For the top-K pooling method, adequate dissemination of the information contained in the dropped nodes is an effective way to preserve node information. To achieve this goal, we adopt the multi-scale information aggregation strategy. As shown in Fig. 1, the multi-scale

Manuscript received December 11, 2024; revised March 3, 2025.

This work was supported in part by the Key R&D Program of Shandong Province under Grant No. 2024TSGC0109, in part by the Humanities and Social Sciences in Universities of Shandong Province under Grant No. J16WJ04, and in part by the Science and Technology Innovation Program for Distinguished Young Scholars of Shandong Province Higher Education Institutions under Grant 2019KJN045.

W. Liu is a Lecturer of the School of Business Administration, Shandong University of Finance and Economics, Jinan, 250014, China (e-mail: vivian.liu@sdufe.edu.cn).

Q. Guo is a Professor of the School of Computing and Artificial Intelligence, Shandong University of Finance and Economics, Jinan, 250014, China (corresponding author to provide phone: 86-0531-88525944; e-mail: guoqiang@sdufe.edu.cn).

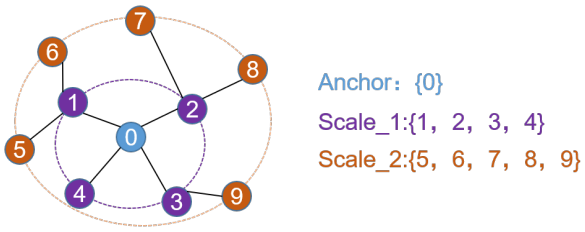

Fig. 1. Multi-scale information in graph structures. Taking the node labeled 0 as an anchor node, nodes 1, 2, 3, and 4 constitute its first-order topological neighboring nodes. The first-order scale encompasses the topological connectivity relationships among these nodes. Similarly, nodes 5, 6, 7, 8, and 9 denote the second-order linking nodes of the anchor node.

topological information expresses the structural information of the graph. In each scale of information, the nodes are pooled with different orders of neighbor relationships. Therefore, we can use the final aggregation of all scales of structural information to preserve the original features as much as possible.

Furthermore, many existing methods tend to overlook the importance of graph structural learning when generating graph representations. The local topology of the graph structure contains crucial information within the subgraphs and plays an important role in understanding the overall graph structure. Global pooling approaches generally aggregate all nodes simultaneously, resulting in a graph representation that can not effectively preserve the structural information and also obscure local features. On the other hand, hierarchical pooling techniques aim to extract hidden messages by generating high-level graphs. The widely-used method is the SAGPool [11], in which a self-attention mechanism is introduced to generate coarsened graphs. Specifically, it utilizes spectral graph convolution and Laplace operators to facilitate information transfer and learn node features and graph topology simultaneously. However, SAGPool is computationally expensive due to its reliance on the spectral technique. To incorporate structural features into node semantics efficiently, we present a multi-scale convolution method called MSConv. Unlike the spectral method, MSConv is computationally less intensive while still effectively passing graph messages to generate nodes and topological structural embeddings. Additionally, it provides a measure of node importance, which aids in condensing the graph size at higher levels of abstraction. In this way, MSConv plays a dual role in capturing graph information and incorporating structural features into the node representations of the multi-scale pooling (MSPool) method.

In this paper, we propose a multi-scale structural learning model for graph classification named MSSGC, which can make full use of the semantic information of nodes and the structural topology embedding to generate the graph representation and consequently boost the performance of graph classification. Specifically, MSSGC consists of two phases. The first phase (i.e., the MSPool phase) is to obtain the discriminative graph representations that retain overall original graph information in a multi-scale manner by using hierarchical pooling. To this end, we scale down the size of the graph by selecting the most informative part of the nodes, and design a multi-scale spatial convolution that plays the roles of message aggregation and importance scores generation for the next coarsened graph. The embeddings provided in each pooling layer are combined with a balance factor to obtain the final representation of the graph. In the second phase (i.e., the classification phase), a classifier is performed to predict the labels of graphs and optimize the model parameters. By incorporating graph structural pooling and convolution into the model, we empower MSSGC to capture multi-scale relationships and extract more robust representations, thereby improving classification performance. Integrating the aforementioned strategies allows MSSGC to effectively leverage the inherent structural information present in the graph, enabling more powerful and discriminative feature extraction.

The main contributions are summarized as follows:

- We propose an effective and multi-scale structural learning model for graph classification (MSSGC). To address the issue of information loss in the top-K pooling method, we employ the multi-scale technique to retain raw information. The topological embedding of each scale is regarded as additional information.
- We design a multi-scale spatial graph convolution (MSConv) for graph learning. The fusion of multi-scale information allows for a more robust and effective analysis of the graph, resulting in the enhanced performance of downstream tasks.
- Extensive experimental results on graph classification tasks demonstrate the effectiveness of the proposed model compared to other state-of-the-art approaches. Additional experimental results also show the interpretability and robustness of our MSSGC model.

The rest of the paper is organized as follows. We first review the related work in Section II, and then describe the proposed MSSGC model in Section III. Experimental results and discussions are analyzed in Section IV. Finally, we conclude this paper with future directions in Section V.

II. Related Work

A. Graph Neural Networks

The field of graph neural networks (GNNs) can be broadly categorized into two main branches: spectral methods and spatial ones. Spectral methods employ spectral filters to define the graph volume from the perspective of graph signal processing. A seminal work by Bruna *et al.* [12] introduced convolution operations defined in the Fourier transform domain for graphs. However, the computational overhead associated with these methods presents challenges when scaling up to accommodate large graphs. To address this concern, Defferrard *et al.* [13] enhanced the efficiency by approximating K-polynomial filters through Chebyshev expansion. In [14], graph convolutional network (GCN) was introduced by expanding on the ChebNet concept, which simplifies the Chebyshev polynomial into a first-order approximation of localized spectral filters. On the other hand, spatial methods can directly aggregate information through topological relations between nodes. GraphSAGE [15] is an inductive representation learning method capable of generalizing to previously unseen nodes by aggregating neighborhood content information. Inspired by the attention mechanism, graph attention network (GAT) [16] was introduced with attention mechanisms to aggregate neighborhood representations with

varying weights. However, current graph learning methods are generally suitable for node classification tasks, but ignore the graph structure learning.

In addition, GNNs have various applications from image processing to complex data structure analysis and neuroimaging. [17] employs the sliding window method to partition the original signals into multiple segments, aiming at achieving the dynamically functional connectivity analysis. It also captures global temporal patterns and local temporal patterns, respectively. [18] describes a variety of non-pairwise relations in spam detection tasks. [19] fine-tunes the multi-scale features designed to capture common information among all scale features as well as private or complementary information for each scale feature. These works represent significant advancements in the field of GNNs by fully exploiting different levels of information.

B. Graph Structure Learning

The field of graph structure learning (GSL) has gained significant attention, as discussed in the literature [20]. The primary goal is to simultaneously learn optimized graph structures and corresponding representations [21]. Numerous studies have delved into this core objective, exploring some specific methods and techniques aimed at improving the quality of graph structures and enhancing the associated representations.

Graph pooling operation plays a pivotal role in extending the concept of pooling to graph data, where nodes are organized into diverse community structures, enabling the learning of a hierarchical graph structure [22]. Meanwhile, hierarchical structure in turn enhances the process of acquiring more effective representations. Various graph pooling techniques have been introduced to tackle this task. For instance, DiffPool [23] generates a coarsened adjacency matrix that captures the strength of connectivity between different clusters. For further attention-based node selection, SAGPool [11] constructs a node hierarchy by employing a self-attention network to selectively drop nodes. Considering the embedding similarity reduction, HGP-SL [24] identifies and removes nodes whose current embeddings closely resemble those aggregated from neighboring nodes, thereby generating hierarchical representations. Besides, GSL has a natural application in rendering GNN models more explainable. In such cases, GSL is typically utilized as a post-processing step following conventional graph representation learning. In order to gain insights about the behaviors of GNN models, PGExplainer [25] identifies crucial subgraph structures that influence representation learning.

C. Graph Classification

Existing methods for graph classification can roughly be grouped into three categories: kernel-based methods, pooling methods, and graph convolution-based methods.

The core of kernel-based methods is the similarity calculation by decomposing graphs into certain substructures and comparing the substructures on various graphs in graph classification. In the method with the shortest path [26], the graph is decomposed into some paths, and the shortest paths are compared according to their length and the labels of the endpoints. In [27], the Weisfeiler-Lehman kernel transforms the initial graph into a series of graphs. In these graphs, node attributes encapsulate both topological and labeling details.

Pooling-based methods are another commonly used alternative for graph classification, which compress graphs into lower-dimensional representations without compromising the structural and attribute information. Some representative methods like DiffPool [23] focus on the cluster-based coarsening strategy, while SAGPool [11] adopts attention mechanisms to select informative nodes. HGP-SL [24] leverages node embedding similarities for pooling, while other methods such as TopKPool [28] retain the most significant nodes based on their embeddings. Global pooling approaches, including Mean pooling and Max pooling, offer a holistic view by aggregating node features across the entire graph.

When applying GNNs to the graph classification task, the convolution and pooling operators are important as two main components of GNNs. The convolution operator extracts the representation of graphs using structural information and node features, while the pooling operator coarsens the structure of the initial graph. Although recent advances have shown significant improvements in the performance of deep learning for graph classification, how to jointly learn structural information in graphs is still under investigation. This work attempts to address this issue.

III. Proposed Method

In this section, we first briefly introduce the problem statement. Then, we present the overview of the proposed multi-scale structure graph classification model, i.e., MSSGC. After that, we discuss the MSPool method in detail, which acts as the individual components of MSSGC and generates high-level abstract coarsen graphs. Finally, we analyze the spatial and time complexity of MSSGC.

A. Problem Statement

Consider a set of graphs $G = \{\mathcal{G}_1, \mathcal{G}_2, \ldots, \mathcal{G}_n\}$ with varying numbers of nodes (n_i) and edges (e_i). Each graph $\mathcal{G}_i$ consists of a vertex set $\mathcal{V}_i$, an edge set $\mathcal{E}_i$, and a node feature matrix $X \in \mathbb{R}^{n_i \times d}$, where d is the dimension of node attributes. The adjacency matrix $A \in \mathbb{R}^{n_i \times n_i}$ describes edge connections. For each graph in the l-th pooling layer as $\mathcal{G}^l$ with n^l nodes, its corresponding adjacency and hidden representation matrices are represented as $A^l \in \mathbb{R}^{n^l \times n^l}$ and $X^l \in \mathbb{R}^{n^l \times d}$, respectively. The features in the pooling layers are fused to generate the final embedding $Z_i \in \mathbb{R}^{1 \times w}$ of the graph $\mathcal{G}_i$, where w is the embedding's dimension. A label matrix $Y \in \mathbb{R}^{n \times c}$ classifies each graph, in whiche $Y_{ij} = 1$ if $\mathcal{G}_i$ is in class j, else $Y_{ij} = 0$. As graphs pass through layers in a network and undergo pooling, their structure and node count may change in different layers. For ease of presentation, we formally define our problem as follows:

Input: Given a set of graphs G with their labels Y, the number of graph neural network layers K_1, the number of pooling layers K_2 and the pooling ratio r.

Output: Predict the unknown graph label of $\mathcal{G}_i$.

Besides, we summarize important notations and definitions throughout this paper in Table I.

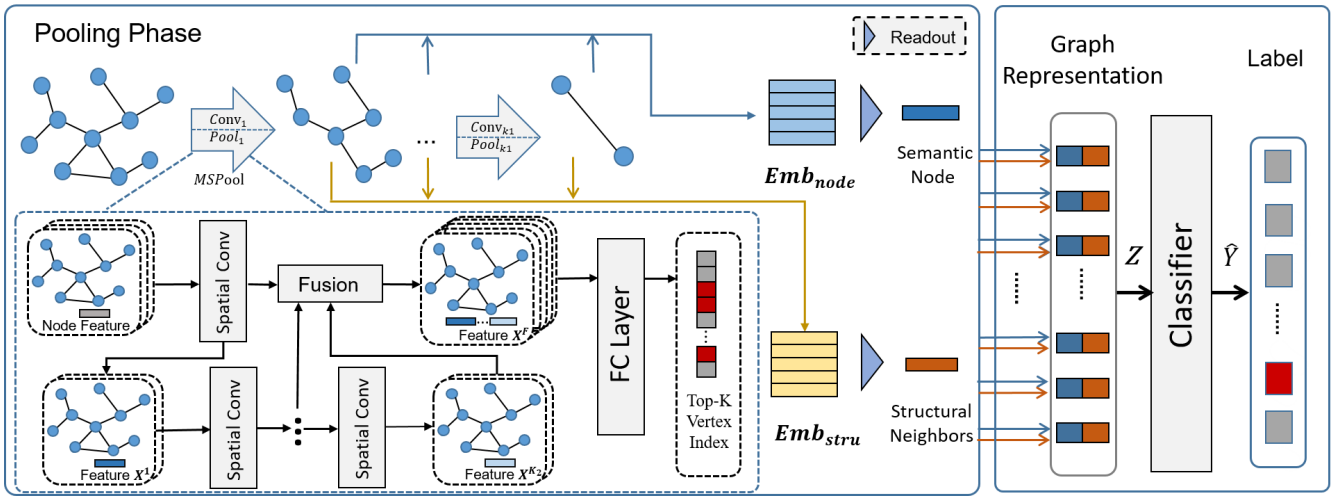

Fig. 2. The MSSGC model unfolds in two targeted phases: a pooling phase and a classification phase. The pooling phase leverages a shared multi-scale graph convolutional network across diverse graphs, condensing their structure according to a specific ratio to form coarsened graphs. These graphs, refined through K_1 pooling layers, become reservoirs of significant topological and semantic insights. Then the model performs the classification phase, where these insights are synthesized into representations that guide precise label assignment.

TABLE I
NOTATIONS AND DEFINITIONS

Notation	Definition
$X^{(t+1)}$	The feature matrix of nodes in the $(t+1)$-th graph convolution layer.
D_v	The diagonal matrix of vertex degrees.
N^l	The node features from the l-th pooling graph.
S^l	The structural features from the l-th pooling graph.
$X^{(l+1)}$	The feature matrix of nodes of the $(l+1)$-th pooling graph.
$A^{(l+1)}$	A indicates the connection strength between vertexes. The $A^{(l+1)}$ indicates the connection relation of the $(l+1)$-th pooling graph.
K_1	The number of convolution layers.
K_2	The number of pooling layers.
r	The pooling ratio.
Z	The embedding of the graphs.
Y	The label matrix of graphs.

B. Model Overview

The whole model contains two phases: pooling and classification. The pooling phase is for coarsening and extracting the discriminative graph representation with multi-scale structural information. As for extracting the structural information in graphs which remains a significant challenge in graph classification tasks, we exploit the multi-scale neighborhood topological structure in the pooling phase to supplement the semantic information of nodes. Finally, the classification phase optimizes the parameters by label classification learning.

For a given graph structure with the adjacency matrix and node features, a graph embedding can be generated after pooling, which contains both semantic and structural information. Such discriminative representations can be fed into the following classifier to recognize the label of the graph. At each pooling layer, we utilize the spatial graph convolution to generate multi-scale features of nodes. These features from different scales are combined and used to judge the importance of the nodes. The coarsen graph in the next layer retains the nodes with the top-K percent importance score and their connecting relationships. At last, the graph representation is obtained by concatenating the node features and the aggregated features of multi-scale neighborhoods in each pooling layer, in which a factor is introduced to balance the importance of output in the intermediate pooling layers. In the classification phase, the discriminative graph embeddings are fed into any classification models (e.g., a fully connected network), and the loss function can help to optimize the parameters in the model. Fig. 2 illustrates the proposed MSSGC model.

C. Structural Pooling

In the phase of structural pooling, the model is provided with the graph structure, which includes the adjacency matrix and node features. A powerful graph representation is generated, which will be used in the subsequent classification phase. In the following, we first explain how to measure the importance of nodes for coarsening the graph in the next layer using the graph MSConv operation, and then discuss how to integrate the information from the pooling process into the graph representation.

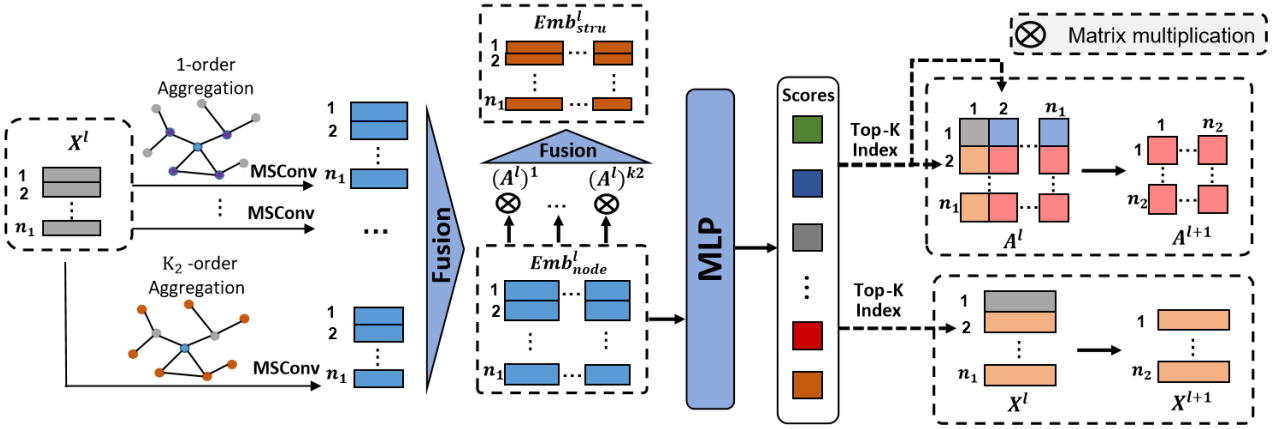

Fig. 3. The graph structure coarsening process leverages the MSConv to create node features at various scales, capturing the graph's complex multi-scale structure. By synthesizing structural features from these multi-scale node interactions, the method abstracts the graph in the next pooling layer, keeping only significant nodes and connections based on top-K feature scores. The proposed framework streamlines the graph by preserving essential nodes and relationships, thus reducing complexity while retaining core structural information for effective analysis.

1) Multi-Scale Pooling for Graph Structure: In the MSPool process, the node and structure embeddings of the graph structure are acquired by our model. The importance of different nodes is then measured, and the top-K nodes with their connecting relationships are selected to be retained for buiding a new high-level graph structure. The specific data flow of the structure coarsening process is shown in Fig. 3.

To fully capture the multi-scale topological structure and generate accurate features of the graph, we jointly learn graph features from different graph convolution layers. Considering the excellent computational efficiency and general scalability of spatial GNNs [29], we exploit a convolutional operation of information by specifying the spatial message-aggregation strategy to retain the original information in graph structure, which can be formulated as

$$\begin{cases} X^{(t+1)} & = \sigma\left(X^{t} \cdot \Theta^{t}\right), \\ X^{(t+1)} & = \left(\sum_{\beta \in \mathcal{N}_v} Agg_v^t\left(w_\beta \cdot x_\beta^{t+1}\right)\right) / \left|\mathcal{D}_v\right|, \end{cases} \tag{1}$$

where $X^{(t+1)}$ denotes the node features in the $(t+1)$-th layer. $Agg_v^t(\cdot)$ acts as the vertex message and hyperedge update function through an aggregation operation. $\Theta^t \in \mathbb{R}^{C^t \times C^{t+1}}$ are the trainable parameters for the layer $t \in [0, K_2]$. These parameters can be refined during the training phase. w_β is a weight linked to the node β. If w_β is not predefined, its value defaults to be 1. $\sigma(\cdot)$ is a non-linear activation function, such as the $ReLU(\cdot)$ function. $|\mathcal{D}_v|$ represents the degree of node v. Dividing with $|\mathcal{D}_v|$ aims to normalize the node features.

After convolution once, the features of each node aggregate the information of the first-order neighbors around it. When the number of the spatial convolution layers is K_2, each node feature in the output $X^{(t+1)}$ aggregates the information of its K_2-order neighbors. To obtain the final node feature representation X_F, the multi-order information is combined together. The MSConv method that convolutes the node features and then concatenates the node features at different scales can be represented by the following operation

$$X_F = [X^1 \| \ldots \| X^{K_2}], \tag{2}$$

where $\|$ denotes the concatenation operation that is a simple yet effective approach for feature fusion [30].

Based on the above MSConv, the node embeddings N^l in the l-th layer are pulled with the graph structure by the MSConv operation as follows

$$N^l = \text{MSConv}\left(A^l, X^l\right), \tag{3}$$

where $A^l \in \{0,1\}^{n^l \times n^l}$ $(l \in [0, K_1 - 1])$ is the adjacency matrix of the graph. Note that X^l is initialized by the input node features, and n^l denotes the number of nodes in the l-th layer.

In addition to the semantic features of nodes, the multi-scale topology of the graph contains rich structural information. The information of node neighbors at each scale also contains local topological structure information of the graph. To better extract the overall information about the graph structure, the local embeddings with multi-scale structural information are aggregated by learning with vector splicing

$$\begin{aligned} S^{l+1} = Concat(&W_1^{l+1}(A^{(l+1)})^1 N^{l+1}, \ldots, \\ &W_{K_2}^{l+1}(A^{(l+1)})^{K_2} N^{l+1}), \end{aligned} \tag{4}$$

where $(A^{(l+1)})^{K_2}$ signifies the K_2-order connections or relationships among nodes, being beneficial to encapsulate indirect relationships in the graph. S^{l+1} delineates the computation of the structural embedding in the $(l+1)$-th layer. It aggregates the influences from various neighborhood orders from 1-st to K_1-th of the nodes by weighing and concatenating their respective node embeddings, thereby producing a richer representation that captures both local and distant contextual information within the graph.

Furthermore, the attention learning mechanism at different scales according to the importance of neighboring nodes allows for the generation of more accurate structural embeddings. Considering the different importance of neighbors of the node, our model employs an attention coefficient matrix W to aggregate information from various neighbors. The attention coefficient of the node j 's feature to the node i can be calculated as

$$w_{ij}^{K_2} = \frac{\exp\left(\text{LeakyReLU}\left([x_i \| x_j]\right)\right)}{\sum_{a \in \mathcal{N}_i^{K_2}} \exp\left(\text{LeakyReLU}\left([x_i \| x_a]\right)\right)}, \tag{5}$$

where $\text{LeakyReLU}(\cdot\|\cdot)$ is an activation function, x denotes the row elements of X, and $\mathcal{N}_i^{K_2}$ represents the K_2-order neighbors of the node i.

In order to generate the coarsened graph structure for the $(l+1)$-th layer, we only retain the nodes of percent r and their associated information in the graph structure in the l-th layer. Then a fully connected network (i.e., MLP) is adopted to learn the importance of different nodes, and the index subset of top-ranked nodes is selected as follows

$$\begin{aligned} Scores^l &= MLP(N^l), \\ k^l &= \max\left(1, \lfloor r * n^l \rfloor\right), \\ idx &= f_{TopK}(Scores^l, k^l), \end{aligned} \tag{6}$$

where r denotes the pooling ratio that has an impact on the graph's structure across different pooling layers. The function $f_{TopK}(\cdot)$ sorts and yields the indices of the top-K nodes. $\lfloor \cdot \rfloor$ represents the floor operation that rounds down to the nearest whole number.

After obtaining the node index for the coarsened graph, we can select node features and adjacency relationships that should be preserved as follows

$$\begin{aligned} A^{l+1} &= A^l(\mathrm{idx}, \mathrm{idx}), \\ X^{l+1} &= X^l(\mathrm{idx}, :). \end{aligned} \tag{7}$$

where $A^l(\mathrm{idx}, \mathrm{idx})$ and $X^l(\mathrm{idx}, :)$ represent the row or column extraction to form the adjacency matrix and node feature matrix for the induced subgraph. Thus, $A^{l+1} \in \mathbb{R}^{n^{l+1} \times n^{l+1}}$ and $X^{l+1} \in \mathbb{R}^{n^{l+1} \times d}$ indicate the graph structure and node feature information of the $(l+1)$-th layer.

2) Readout of Graph Representation: Following K_1 iterations of graph convolution and pooling, we obtain K_1 refined subgraphs. For a more powerful graph-level representation, we have also crafted a readout function that adeptly amalgamates the embeddings from each subgraph at each pooling layer. Specifically, the max-readout with embeddings of nodes and hyperedges in each subgraph is used, i.e.,

$$\begin{aligned} Emb^{l+1} &= \mathrm{Readout}\left(N^{l+1}, S^{l+1}\right) \\ &= Concat\left(\max_{q=1}^{d} N^{l+1}(:, q), \max_{q=1}^{d} S^{l+1}(:, q)\right), \end{aligned} \tag{8}$$

where the embedding Emb^{l+1} of the whole graph in the $(l+1)$-th layer is formed by extracting the maximum value of each d embedding dimension of the node features and structural features in the graph through a $\mathrm{Readout}$ function. Subsequently, we aggregate the readout outputs from various levels to construct the final graph representation as follows:

$$Z^{l+1} = \begin{cases} Concat(Emb^{l+1}, Z^l) & \text{if } \lambda = 1 \\ (1-\lambda) Emb^{l+1} + \lambda Z^l & \text{if } \lambda \neq 1 \end{cases}, \tag{9}$$

where $Z^0 = \phi$, and λ is a balance factor for controlling the extent of information that is preserved at the previous pooling layers.

D. Classification

At last, we feed the graph representations Z from Eq. (9) into a classifier to achieve the classification result as follows

$$\hat{Y} = \mathrm{F}(Z), \tag{10}$$

where $\mathrm{F}(\cdot)$ can be any off-the-shelf classifiers that can be used to handle the classification task, such as fully connected networks, support vector machine [31], and decision tree [32].

To train our MSSGC, the common cross-entropy [33] of predictions over the labels is used as the loss function

$$\mathcal{L} = -\sum_{i \in L} \sum_{j=1}^{c} Y_{ij} \log \hat{Y}_{ij}, \tag{11}$$

where Y_{ij} is the ground truth, and $\hat{Y}_{ij}$ represents the predicted probability of the graph belonging to the class j.

E. Complexity Analysis

We theoretically analyze the computational complexity of our proposed MSSGC. Let N be the number of nodes in the graph, and each node has C features. Assuming each node has M neighbors. The computational complexity of a graph convolutional layer is $\mathcal{O}(NMC)$. For the MSConv with K_2 convolution layers, its computational complexity is $\mathcal{O}\left(K_2 NMC\right)$. For each scale, the attention weights between the node and its neighbors need to be computed, resulting in a time complexity of approximately $\mathcal{O}\left(K_1 NMC\right)$ with K_1 scales. Besides, the cost of calculating the node and structure embeddings by MSConv is $\mathcal{O}\left(2 * K_2 NMC + 2 * K_1 NMC\right)$.

Due to K_1 and K_2 being constants and $M, C < N$ in most cases, the total time complexity of our method is approximately equal to $\mathcal{O}\left(N^3\right)$. As for the spatial complexity, the proposed MSSGC takes a constant number of two-dimensional arrays to store information on the graph, so it is approximately equal to $\mathcal{O}\left(N^2\right)$.

IV. Experiments

In this section, we conduct experiments on public datasets to assess the performance and resilience of the proposed MSSGC. We want to answer three questions from experiments: (1) How does MSSGC compare to the state-of-the-arts for the simple graph classification task, including GNN-based methods, kernel-based methods, and pooling-based methods? (2) What makes our model more effective than others? (3) How to analyze the robustness of MSConv with very few training samples? Correspondingly, we conduct the following experiments: (1) Comparing the graph classification accuracy of MSSGC with state-of-the-art (SOTA) methods; (2) Conducting ablation studies to investigate the impact of different components in MSSGC; (3) Analyzing the robustness of MSConv with very few training samples.

A. Datasets

Six commonly used public benchmarks are applied to graph classification tasks for empirical studies. Specifically, we will examine the Mutagenicity, MUTAG, PROTEINS, D&D, IMDB-BINARY and IMDB-MULTI datasets. Diverse ranges are covered by these datasets, including protein, chemical graph and social network classification.

Mutagenicity [34] is a chemical compound dataset of drugs, which consists of two classes: mutagen and non-mutagen. Small molecules dataset MUTAG [35] comprises 188 mutagenic aromatics, with nodes and edges respectively depicting atoms and chemical bonds. PROTEINS and D&D [36] are two protein graph datasets, in which nodes indicate amino acids, edges link nodes that are less than 6 angstroms apart, and the label shows whether a protein is an enzyme

TABLE II
DETAILS FOR THE DATASETS USED IN THE EXPERIMENTS.

Dataset	Mutagenicity	MUTAG	PROTEINS	D&D	IMDB-B	IMDB-M
Classes	2	2	2	2	2	3
Num_Graphs	4337	188	1113	1168	1000	1500

or not. In the social network datasets, IMDB-BINARY and IMDB-MULTI [37] are relational datasets that consist of a network of actors or actresses who played roles in movies in IMDB. A performer or actress is represented by a node, and an edge links two nodes when they feature in the same film. Table II lists the details for the above datasets. Following previous work [24], we randomly split each dataset into three parts: 80% as the training set, 10% as the validation set, and the remaining 10% as the test set.

B. Settings

The settings of compared methods, training details and evaluation are as follows:

Compared Methods. The competing methods encompass three categories: graph kernel methods, graph neural networks (GNNs), and graph pooling models. Specifically, graph kernel methods use carefully designed kernels to perform graph classification based on graph pattern recognition. Two classical algorithms, namely Shortest-Path Kernel (SP) [26] and Weisfeiler-Lehman Kernel (WL) [27], are chosen as baselines. GNNs are designed to learn meaningful node-level representations, and the addition of readout functions is applied in the node representations to extract graph information globally. The following three methods fall under GNNs: GCN [14], GraphSAGE [15], which learns node embeddings through sampling and aggregation, and GAT [16]. The graph pooling group consists of numerous models that generate graph-level representation with coarsened operators. ECC [38], Set2Set [39], and DGCNN [40] are three novel global graph pooling algorithms. Besides, three hierarchical graph pooling models, i.e., DiffPool [23], SAGPool [11], EdgePool [41], CDR [42], and OMG [43], are also employed as baselines.

Training Details. The learning rate is searched in $\{0.01, 0.001$,1e-4,1e-5$\}$, and the pooling ratio r is selected in $[0.1, 0.9]$ and the balance factor λ is in $[0, 1]$. Considering the oversmooth of convolution operations [14] and practical computational limitations, we set the number of convolution layers $K_1 \in [1, 6]$ in each pooling layer and the number of pooling layers $K_2 \in [1, 6]$. A fully connected layer is employed as the final classifier, followed by a softmax function. LeakyRelu [44] is chosen as the activation function used in the MSConv operation. We perform our model 500 epochs with the Adam optimizer [45]. The optimal combinations of main hyperparameters for each dataset are provided in Table III.

Evaluation. For more comprehensive evaluations, we perform 20-fold cross-validation to evaluate the performance of all models and report the accuracy averaged over 20 folds for all the above datasets.

C. Graph Classification Performance and Discussion

To demonstrate the effectiveness of MSSGC, we examine it in a wide range of general graph classification datasets. Experimental results on the datasets are reported in Table IV, in which the best results are bolded and the second best results are underlined. From these results, we have the following observations.

Performance. MSSGC outperforms the competing methods greatly. In particular, on the datasets of Mutagenicity, MUTAG, PROTEINS, D&D, IMDB-B and IMDB-M, the accuracy of our model increases over that of the previous SOTA methods by 0.42%, 1.12%, 0.91%, 1.94%, 1.48%, 1.69%, respectively. The proposed method achieves superior performance across all test datasets, while the competing methods show the inconsistent results and variability. MSSGC's notable improvement of 3.16% over the Set2Set (the previous best-performing graph pooling method) on the IMDB-M dataset is a clear testament to its generalization ability.

Discussion. Graph classification can be roughly grouped into three categories: kernel-based methods, graph convolution-based methods, and pooling methods. The core of kernel-based methods is the similarity calculation by decomposing graphs into certain substructures and comparing the substructures. In the kernel method with SP [26], the graph is decomposed into some paths, and the shortest paths are compared according to their length and the labels of the endpoints. WL [27] transforms the initial graph into a series of graphs, whose node attributes encapsulate both topological and labeling details. The graph neural network-based approaches (e.g., GCN [14], GraphSAGE [15], and GAT [16]) do not fully utilize the information of the high-level hierarchy when they aggregate all the node features globally at once to form the graph features. The pooling-based methods contain the convolution operator and the pooling operator. The convolution operator extracts the representation of the graph using structural information and node features, while the pooling operator coarsens the structure of the initial graph. The pooling method is another commonly used alternative for graph classification, allowing for the compression of graphs into lower-dimensional representations without compromising the structural and attribute information. The global methods, such as ECC [38], Set2Set [39], and DGCNN [40], also do not fully exploit the information in the graph structure. And hierarchical graph pooling techniques like DiffPool [23] focus on the cluster-based coarsening, while SAGPool [11] adopts the attention mechanisms to select informative nodes. Besides, EdgePool [41] retains the significant edges based on their embeddings. Both CDR [42] and OMG [43] focus on the robustness of the model and the resistance to label noise effects, respectively.

Compared to other methodologies that might overlook the latent interconnections among nodes, our model stands out by actively utilizing graph multi-scale structures to learn and model node representations. Within our model, a shared convolutional pooling network is systematically trained across

TABLE III
THE OPTIMAL COMBINATIONS OF MAIN HYPERPARAMETERS FOR EACH DATASET.

Dataset	Mutagenicity	MUTAG	PROTEINS	D&D	IMDB-B	IMDB-M
K_1	2	3	2	3	2	3
K_2	3	3	4	2	3	3
r	0.5	0.6	0.8	0.8	0.5	0.5
λ	1	1	0.5	0.5	0.2	0.1

TABLE IV
CLASSIFICATION RESULTS OF DIFFERENT METHODS ON ALL DATASETS.

Methods	Mutagenicity	MUTAG	PROTEINS	D&D	IMDB-B	IMDB-M
SP	71.63±2.19	80.45±1.26	75.71±2.73	78.72±3.89	64.2±4.36	42.39±2.11
WL	80.32±1.71	82.05±0.36	76.16±3.99	76.44±2.35	73.8±3.97	49.33±4.75
GCN	79.81±1.58	87.20±5.11	75.17±3.63	73.26±4.46	70.01±2.28	51.20±5.13
GraphSAGE	78.75±1.18	79.80±13.94	74.01±4.27	75.78±3.91	68.87±4.53	49.90±5.04
GAT	78.89±2.05	89.40±6.18	75.72±4.01	77.30±3.68	65.14±5.18	47.80±3.10
ECC	71.89±1.21	88.33±5.34	72.33±3.47	73.68±3.85	67.70±2.83	43.48±3.13
Set2Set	80.84±0.67	86.78±7.33	79.33±0.84	70.83±0.84	71.00±7.54	49.73±4.19
DGCNN	80.41±1.02	85.8±1.94	79.99±0.44	70.06±1.21	69.20±3.09	45.60±3.42
DiffPool	80.68±0.67	88.87±6.75	79.90±2.95	78.61±1.32	68.40±6.18	45.62±3.41
SAGPool	79.72±0.79	93.32±4.16	78.28±4.03	78.35±3.57	72.80±2.39	49.43±2.68
EdgePool	81.41±0.88	92.27± 5.79	82.38±0.82	79.20±2.61	71.96±3.19	49.24±3.47
CDR	79.63±1.58	91.52± 4.58	80.19±2.51	80.58±4.16	69.24±3.81	47.41±2.58
OMG	81.05±1.36	91.74± 5.27	81.86±1.38	81.68±3.15	70.83±2.50	48.63±3.69
MSSGC	**81.85±3.32**	**94.45±4.80**	**83.31±3.07**	**83.65±3.39**	**74.35±3.13**	**52.93±3.46**

TABLE V
ABLATION RESULTS OF OUR METHOD ON THE IMDB-MULTI DATASET.

Construction	Scale-1		Scale-2		Scale-3		Scale-4	
Emb_S	TRUE	FALSE	TRUE	FALSE	TRUE	FALSE	TRUE	FALSE
Readout_Max	49.55±2.69	40.57±2.01	51.21±2.72	44.23±2.54	52.03±2.36	44.36±2.17	**52.89±3.53**	45.02±3.07
Readout_Mean	48.63±5.58	39.43±5.51	48.68±4.89	42.23±3.79	50.25±4.46	43.13±4.98	50.96±4.27	43.83±4.15
Readout_Concat	44.66±4.89	38.13±5.19	45.73±4.41	40.66±4.87	47.86±3.72	42.56±4.29	48.13±3.28	42.97±3.89

all graph structures, which not only compresses the overall number of nodes but also integrates diverse multi-scale neighborhood configurations to produce more comprehensive pooled graphs. By leveraging the intrinsic structure of the graph, our model can effectively model the multi-scale correlations that exist among nodes, which results in the extraction of a variety of hidden relationships that might remain undiscovered.

Furthermore, the incorporation of multi-scale information acts as a counterbalance, making up for any information that could be lost in directly applying the top-K method. A distinguishing feature of MSSGC during the pooling phase is its meticulous attention on both the immediate local structures found within the graph and the broader multi-scale topological structures. The dual focus ensures that the resultant representation of the graph is not just comprehensive but also highly discriminative, setting it apart from other models and enhancing its applicability in various scenarios.

D. Ablation Study

To better understand the performance of MSSGC for graph classification, several ablation experiments are conducted to identify the critical components of the model in the IMDB-MULTI dataset. They aim to analyze the impact of the following components on the overall performance of the model: (1) *Scale Range*. To explore the effect of different ranges of scale, we consider the following four settings for comparison here: Scale-1, Scale-2, Scale-3 and Scale-4. (2) *Readout Method*. We choose three readout methods of node information and local structure, i.e., Max, Mean, and Concatenate. (3) *Topological Structure*. To verify the effectiveness with topological information of the multi-scale neighborhood structure, we compare the graph representation with and without multi-scale neighborhood structure embeddings.

The ablation results on IMDB-MULTI are provided in Table V. Notably, the best performance is observed at Scale-4 with the Readout_Max method ($52.89 \pm 3.53\%$), indicating the effectiveness of our multi-scale approach and the importance of high-order scales in capturing relevant features for classification. It can be found that no matter what the range of scale in graph structure is selected and what way of readout functions, the results with multi-scale neighborhood structure embeddings are always better than those without multi-scale neighborhood structure embeddings. This means that the multi-scale representation method has a better performance compared with only using single-layer features. Furthermore, from the results of different readout methods, we can observe that the Readout_Mean method cannot obtain the best performance. Its performance is very oscillating

when this readout method is used in the feature selection, and the fluctuation in the accuracy range is much larger than that of only using the Readout_Max method.

The superior results achieved by our proposed framework can be credited to the following key factors. Regarding the graph structure, the adaptability in the graph's scale offers a versatile modeling ability, enabling the capture of multi-scale correlations effectively, which is particularly beneficial in modeling local topological structures through multi-scale neighborhood structure embeddings. Such an approach proves advantageous for crafting a more distinctive graph representation. These multi-scale correlations offer a more significant boost in the learning process compared to the raw relationships, since they adeptly discern the underlying relationships within the data, offering a more nuanced understanding. Depending on the sparsity of the data, there is a flexibility in determining the appropriate scale for constructing the graph structure, ensuring optimal results based on the dataset at hand. Turning our attention to the readout method for representation, the diminished outcomes observed with the Readout_Mean method suggest that aggregating values of all nodes in each dimension may not be as effective as focusing on their maximum values for representation. While the mean takes into account the entirety of node values, it often overlooks specific data points that might become overshadowed amidst a plethora of features, thereby diluting the overall representation's effectiveness.

E. Effectiveness and Robustness Analysis of MSConv

1) Settings: In this subsection, we will discuss the effectiveness and robustness of the MSConv component used in MSSGC. Here, we perform the experiments in the node classification task. So the settings of datasets and compared methods are different from the above experiments.

Datasets. In our experiments, we choose four public benchmarks that fall into two primary categories: (1) Publication Citation Network. This includes Citeseer and Pubmed as used in [46]. Within these two datasets, each academic paper is represented by a vertex, which is linked with an initial feature using the bag of words model. In this case, the main task is to predict the specific category to which a given paper pertains. (2) Social Media Network. This encompasses Github and Facebook as in [47]. These datasets highlight the interconnections between various websites. The vertex features for these networks are derived from the lexical content present on each website.

The detailed statistical attributes of these datasets are depicted as follows. In the Citation Network category: Citeseer comprises 6 classes with a total of $3,327$ nodes and $4,732$ edges. Each node in this dataset has $3,703$ features associated with it. PubMed has 3 distinct classes. It is a larger dataset with $19,717$ nodes connected by $44,338$ edges, and each node is described by 500 features. For the Social Media Network category: Github is structured into 2 classes. It is a sizable dataset with $37,700$ nodes and a significant number of $289,003$ edges. Every node here is characterized by $4,005$ features. Facebook, with 4 classes, contains $22,470$ nodes and $171,002$ edges. Each node in this dataset is equipped with $4,714$ features. In each dataset, we randomly select $5/10$ samples from each category for training purposes and another 5 samples from each category for validation. The remaining samples are allocated for testing in our experiments.

TABLE VI
ACCURACY RESULTS WHEN TRAINING WITH 5 SAMPLES PER CATEGORY, THE HIGHEST ACCURACY OUTCOMES ARE MARKED IN BOLD.

Method	Citeseer	Pumbed	Github	Facebook
GCN	58.62	69.38	72.18	57.71
GAT	60.21	70.51	72.96	55.82
GraphSAGE	56.48	68.4	72.78	56.14
GIN	55.40	68.28	72.01	54.50
MSConv	**60.54**	**70.69**	**73.25**	**58.69**

TABLE VII
ACCURACY RESULTS WHEN TRAINING WITH 10 SAMPLES PER CATEGORY, THE HIGHEST ACCURACY OUTCOMES ARE MARKED IN BOLD.

Method	Citeseer	Pumbed	Github	Facebook
GCN	63.83	73.05	77.37	65.77
GAT	64.72	73.78	77.39	63.82
GraphSAGE	59.32	72.3	75.98	64.13
GIN	59.64	72.59	74.59	61.35
MSConv	**64.99**	**74.19**	**78.41**	**66.82**

Compared Methods. Four typical SOTA methods, including spectral method (GCN [14]) and spatial methods (GraphSAGE [15], GAT [16]), and GIN [48]), are selected for comparisons.

2) Results and Discussions: The results from experiments on all four datasets can be found in Table VI and Table VII. Specifically, Table VI corresponds to an experiment with only 5 samples in the training, and the corresponding experiment in Table VII has 10 training samples. In these settings, the results unequivocally demonstrate that our model consistently outperforms other competitors across all four datasets, verifing the effectiveness and robustness of our proposed MSConv method.

MSConv outperforms other graph learning methods greatly. In particular, on the datasets of Citeseer, Pumbed, Github, and Facebook, the accuracy of our MSConv increases over that of the previous SOTA methods by 0.27%, 0.41%, 1.02%, 1.05% with 10 training samples, respectively. Notably, our model exhibits even greater performance gains in scenarios with limited 5 or 10 labeled samples during training. It suggests that the proposed model is particularly effective in handling situations with a scarcity of labeled samples. In such case, the labeled data is often limited in many real-world applications, and thus the role of data correlation becomes increasingly significant in the representation learning process. Our model leverages multi-scale data correlation to a great extent, thereby significantly enhancing its robustness. These findings again underscore the effectiveness and reliability of the model in challenging scenarios, where limited labeled data poses a significant obstacle.

3) Visualization: In order to intuitively observe the learning effectiveness of MSConv compared with other SOTA methods, we visualize the node embeddings on the Citeseer dataset by the t-SNE tool, which aims to apply a dimensionality reduction technique to the high-dimensional features obtained from each method to allow for a 2D representation. Fig. 4 displays the results of t-SNE visualization of different methods, where t-SNE is applied to the feature represen-

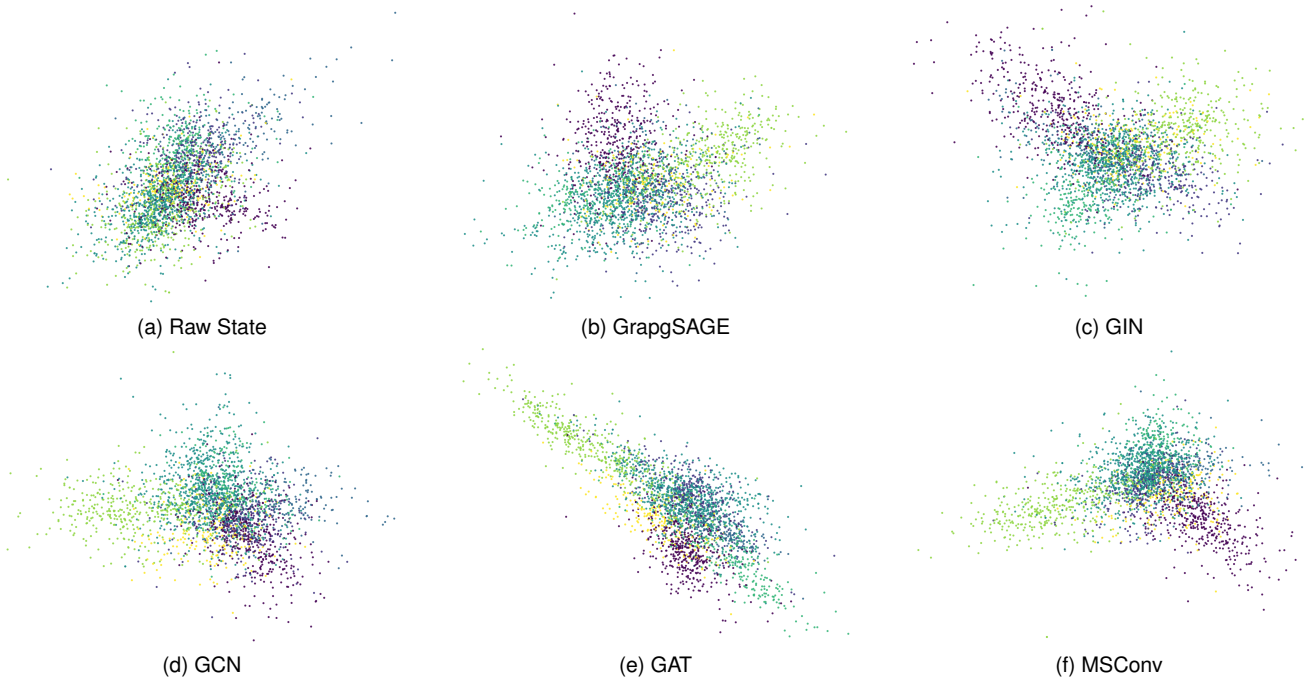

Fig. 4. t-SNE Visualization of the node embeddings in the cases when the training data has 5 or 10 samples on the Citeseer dataset.

tations of nodes or graphs as obtained by different GNN models, including a raw state (original features without any transformation), GraphSAGE, GIN, GCN, GAT, and our proposed MSConv. Each point represents a node in the dataset, and colors denote different categories or clusters within the data. It can be observed from the results that compared with other competing methods, the proposed MSConv method yields discernible clustering, which provides a qualitative confirmation of our method's efficacy in extreme scenarios.

V. Conclusion and Future Work

In this work, we introduce the MSSGC model for the graph classification task by leveraging multi-scale convolution, i.e., MSConv. The proposed method can create the hierarchical graph representations that effectively capture the essential topological and structural information of the graphs. By employing a selective node retention strategy at each pooling layer, our method maintains the most critical nodes, thereby simplifying the graph structures without losing significant information. The simplified structures are enriched by cross-scale semantic and local structural embeddings, providing a solid foundation for a variety of downstream tasks. Our thorough evaluation across several benchmarks validate the MSSGC's superior efficacy over current leading methods, underscoring our model's exceptional capability to understand complex graph data nuances.

Future research can explore to achieve a balance between the abstraction levels of pooled structures and their original forms, aiming to reduce information entropy. Such advancements may unlock new potentials of our MSSGC, enhancing its performance and applicability in more complex scenarios. Besides, many graphs have low-rank and sparse properties [49], [50]. Thus, various structured low-rank constraints [51]–[53] can be imposed on these graphs for better performance robustness and generalization.

References

[1] K. Lin, R. Chen, J. Chen, P. Lu, and F. Yang, "Multi-view block matrix-based graph convolutional network," *Engineering Letters*, vol. 32, no. 6, pp1073-1082, 2024.

[2] K. Wu, H. Dai, S. Wang, and C. Liu, "Point cloud classification network based on dynamic graph convolution," *Engineering Letters*, vol. 31, no. 4, pp1859-1866, 2023.

[3] S. Li, Z. Li, J. Wu, J. Miao, Y. Bai, X. Yu, and K. Li, "Attention feature fusion graph convolutional network for target-oriented opinion words extraction," *Engineering Letters*, vol. 31, no. 3, pp1273-1280, 2023.

[4] Q. Wang, R. He, C. Zhu, and H. Rao, "Short-time traffic flow prediction based on high-order graph convolutional networks," *IAENG International Journal of Computer Science*, vol. 51, no. 10, pp1612-1626, 2024.

[5] Y. Han and H. Dai, "Research on network traffic classification based on graph neural network," *IAENG International Journal of Computer Science*, vol. 51, no. 12, pp2043-2050, 2024.

[6] P. Mahé and J.-P. Vert, "Graph kernels based on tree patterns for molecules," *Machine Learning*, vol. 75, no. 1, pp3-35, 2009.

[7] A. Fout, J. Byrd, B. Shariat, and A. Ben-Hur, "Protein interface prediction using graph convolutional networks," in *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[8] S. Hui, Q. Guo, X. Geng, and C. Zhang, "Multi-guidance CNNs for salient object detection," *ACM Transactions on Multimedia Computing, Communications and Applications*, vol. 19, no. 3, article 117, 2023.

[9] Q. Qiao, M. Qu, W. Wang, B. Jiang, and Q. Guo, "Effective global context integration for lightweight 3D medical image segmentation," *IEEE Transactions on Circuits and Systems for Video Technology*, pp1-14, 2024.

[10] Q. Guo, L. Fang, R. Wang, and C. Zhang, "Multivariate time series forecasting using multiscale recurrent networks with scale attention and cross-scale guidance," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 36, no. 1, pp540-554, 2025.

[11] J. Lee, I. Lee, and J. Kang, "Self-attention graph pooling," in *Proceedings of International Conference on Machine Learning*, pp3734-3743, 2019.

[12] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, "Spectral networks and locally connected networks on graphs," in *Proceedings of International Conference on Learning Representations*, 2014.

[13] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Advances in Neural Information Processing Systems*, vol. 29, 2016.

[14] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proceedings of International Conference on Learning Representations*, 2017.

[15] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Advances in Neural Information Processing Systems*, vol. 30, 2017.
[16] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," in *Proceedings of International Conference on Learning Representations*, 2018.
[17] R. Hu, L. Peng, J. Gan, X. Shi, X. Zhu, "Complementary graph representation learning for functional neuroimaging identification," in *Proceedings of the 30th ACM International Conference on Multimedia*, pp3385-3393, 2022.
[18] X. Sun, H. Yin, B. Liu, H. Chen, J. Cao, Y. Shao, and N. Q. V. Hung, "Heterogeneous hypergraph embedding for graph classification," in *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, pp725-733, 2021.
[19] R. Hu, Z. Deng, and X. Zhu, "Multi-scale graph fusion for co-saliency detection," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 9, pp7789-7796, 2021.
[20] Z. Zhou, S. Zhou, B. Mao, X. Zhou, J. Chen, Q. Tan, et al., "OpenGSL: A comprehensive benchmark for graph structure learning," *arXiv preprint arXiv:2306.10280*, 2023.
[21] D. Yu, R. Zhang, Z. Jiang, Y. Wu, and Y. Yang, "Graph-revised convolutional network," in *Proceedings of Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp378-393, 2021.
[22] F. Hu, Y. Zhu, S. Wu, L. Wang, and T. Tan, "Hierarchical graph convolutional networks for semi-supervised node classification," in *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pp4532-4539, 2019.
[23] Z. Ying, J. You, C. Morris, X. Ren, W. Hamilton, and J. Leskovec, "Hierarchical graph representation learning with differentiable pooling," in *Advances in Neural Information Processing Systems*, vol. 31, 2018.
[24] Z. Zhang, J. Bu, M. Ester, J. Zhang, C. Yao, Z. Yu, and C. Wang, "Hierarchical graph pooling with structure learning," *arXiv preprint arXiv:1911.05954*, 2019.
[25] D. Luo, W. Cheng, D. Xu, W. Yu, B. Zong, H. Chen, and X. Zhang, "Parameterized explainer for graph neural network," in *Advances in Neural Information Processing Systems*, vol. 33, pp19620-19631, 2020.
[26] K. M. Borgwardt and H.-P. Kriegel, "Shortest-path kernels on graphs," in *Proceedings of the Fifth IEEE International Conference on Data Mining*, pp8-16, 2005.
[27] N. Shervashidze, P. Schweitzer, E.J. Van Leeuwen, K. Mehlhorn, and K.M. Borgwardt, "Weisfeiler-lehman graph kernels," *Journal of Machine Learning Research*, vol. 12, pp2539-2561, 2011.
[28] H. Gao and S. Ji, "Graph u-nets," in *Proceedings of International Conference on Machine Learning*, pp2083-2092, 2019.
[29] N. Yadati, M. Nimishakavi, P. Yadav, V. Nitin, A. Louis, and P. Talukdar, "Hypergcn: A new method for training graph convolutional networks on hypergraphs," in *Advances in Neural Information Processing Systems*, vol. 32, 2019.
[30] U. G. Mangai, S. Samanta, S. Das, and P. R. Chowdhury, "A survey of decision fusion and feature fusion strategies for pattern classification," *IETE Technical Review*, vol. 27, no. 4, pp293-307, 2010.
[31] C. Cortes and V. Vapnik, "Support vector machine," *Machine Learning*, vol. 20, no. 3, pp273-297, 1995.
[32] S. R. Safavian and D. Landgrebe, "A survey of decision tree classifier methodology," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 21, no. 3, pp660-674, 1991.
[33] L. Feng, S. Shu, Z. Lin, F. Lv, L. Li, and B. An, "Can cross entropy loss be robust to label noise?" in *Proceedings of Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, pp2206-2212, 2021.
[34] J. Kazius, R. McGuire, and R. Bursi, "Derivation and validation of toxicophores for mutagenicity prediction," *Journal of Medicinal Chemistry*, vol. 48, no. 1, pp312-320, 2005.
[35] A. K. Debnath, R. L. Lopez de Compadre, G. Debnath, A. J. Shusterman, and C. Hansch, "Structure-activity relationship of mutagenic aromatic and heteroaromatic nitro compounds. Correlation with molecular orbital energies and hydrophobicity," *Journal of Medicinal Chemistry*, vol. 34, no. 2, pp786-797, 1991.
[36] P. D. Dobson and A. J. Doig, "Distinguishing enzyme structures from non-enzymes without alignments," *Journal of Molecular Biology*, vol. 330, no. 4, pp771-783, 2003.
[37] P. Yanardag and S. V. N. Vishwanathan, "Deep graph kernels," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp1365-1374, 2015.
[38] M. Simonovsky and N. Komodakis, "Dynamic edge-conditioned filters in convolutional neural networks on graphs," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp3693-3702, 2017.
[39] O. Vinyals, S. Bengio, and M. Kudlur, "Order matters: Sequence to sequence for sets," in *Proceedings of International Conference on Learning Representations*, 2015.
[40] M. Zhang, Z. Cui, M. Neumann, and Y. Chen, "An end-to-end deep learning architecture for graph classification," in *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, pp4438-4445, 2018.
[41] F. Diehl, "Edge contraction pooling for graph neural networks," *arXiv preprint arXiv:1905.10990*, 2019.
[42] X. Xia, T. Liu, B. Han, C. Gong, N. Wang, Z. Ge, and Y. Chang, "Robust early-learning: Hindering the memorization of noisy labels," in *International Conference on Learning Representations*, 2020.
[43] N. Yin, L. Shen, M. Wang, X. Luo, Z. Luo, and D. Tao, "OMG: Towards effective graph classification against label noise," *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, no. 12, pp12873-12886, 2023.
[44] B. Xu, N. Wang, T. Chen, and M. Li, "Empirical evaluation of rectified activations in convolutional network," *arXiv preprint arXiv:1505.00853*, 2015.
[45] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proceedings of International Conference on Learning Representations*, 2015.
[46] G. Namata, B. London, L. Getoor, B. Huang, and U. Edu, "Query-driven active surveying for collective classification," in *Proceedings of 10th International Workshop on Mining and Learning with Graphs*, 2012.
[47] B. Rozemberczki, C. Allen, and R. Sarkar, "Multi-scale attributed node embedding," *Journal of Complex Networks*, vol. 9, no. 2, cnab014, 2021.
[48] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How powerful are graph neural networks?" in *Proceedings of International Conference on Learning Representations*, 2019.
[49] W. Jin, Y. Ma, X. Liu, X. Tang, S. Wang, and J. Tang, "Graph structure learning for robust graph neural network," in *Proceedings of the 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp66-74, 2020.
[50] D. Luo, W. Cheng, W. Yu, B. Zong, J. Ni, H. Chen, and X. Zhang, "Learning to drop: Robust graph neural network via topological denoising," in *Proceedings of the Fourteenth ACM International Conference on Web Search and Data Mining*, pp779-787, 2021.
[51] Q. Guo, Y. Zhang, S. Qiu, and C. Zhang, "Accelerating patch-based low-rank image restoration using kd-forest and Lanczos approximation," *Information Sciences*, vol. 556, pp177-193, 2021.
[52] X. Geng, Q. Guo, S. Hui, M. Yang, and C. Zhang, "Tensor robust PCA with nonconvex and nonlocal regularization," *Computer Vision and Image Understanding*, vol. 243, article 104007, 2024.
[53] T. Yan and Q. Guo, "Tensor robust principal component analysis via dual lp quasi-norm sparse constraints," *Digital Signal Processing*, vol. 150, article 104520, 2024.