Multi-View Block Matrix-Based Graph Convolutional Network

Kaibiao Lin, Runze Chen, Jinpo Chen*, Ping Lu and Fan Yang*

***Abstract*—In recent years, graph convolutional networks (GCNs) have gained significant attention for graph embedding learning. However, the efficacy of GCNs and their variants is often constrained by the implicit homophily assumption, which presumes nodes in close proximity to exhibit similar features. This assumption becomes particularly limiting in the context of complex heterogeneous graphs, as it restricts GCNs' ability to aggregate diverse node information. Addressing this, we propose a novel aggregation mechanism adept at discerning homophilic from heterophilic neighborhoods, thus achieving effective "classification aggregation". More specifically, this study combines block modeling with the aggregation process, allowing the GCNs to learn the aggregation rules across various classes of neighbors automatically. On this basis, a multi-view module is designed to extract semantic information from each view, and the final embedding is obtained by aggregating the information from all views through the attention module. Experiments indicated that the proposed method performs optimally across both heterogeneous and homogeneous graph datasets.**

***Index Terms*—Multi-View Networks, Block Matrix, Graph Convolutional Networks, Graph Neural Networks.**

I. Introduction

GRAPH network data processing traditionally relies on matrix decomposition methods. However, these methods struggle to scale with the expanding size and increasing sparsity of real-world networks. Against this backdrop, graph representation learning has gained prominence. It transforms graph nodes into low-dimensional vectors, allowing adaptable application of these representations in various data mining tasks. This adaptability significantly improves the model's accuracy and reliability for different downstream tasks. Currently, research predominantly focuses on learning representations for homogeneous networks. Notable methods include random walk-based graph representations, which gather topological graph information by traversing neighboring nodes. Models like DeepWalk [1], LINE [2], and Node2vec [3] are the most representative of this domain. In more detail, DeepWalk combined skip-gram and random walk to derive network node embeddings. Additionally, LINE excelled in sparse graphs by employing first-order and second-order neighbor similarities for precise node representation learning. Meanwhile, Node2vec merged depth-first and breadth-first sampling to capture local and preserve global topological information within the embedding space.

In recent years, inspired by recurrent neural networks and convolutional neural networks, neural network-based graph representation learning has become a hot research topic. In this context, GCNs [4] stand out for their effectiveness in heterogeneous graph representation learning. The essence of these models lies in feature aggregation. More specifically, in GCNs, a node's embedding is refined by aggregating feature information from its neighbors through convolutional operations, and then tailored for specific downstream tasks under partial label supervision. The great success of GCNs can be largely attributed to the integration of topological structure with node feature information, facilitating the learning of node embeddings within a supervised learning framework.

Although widely employed, GCNs and their variants [5,6] have two main limitations. First, real-world networks, such as social networks [7,8], physical systems [9,10], traffic networks [11,12], citation networks [13,14], recommender systems [15,16,17], and knowledge graphs [18,19] are often heterogeneous graphs. These networks have diverse nodes and edges, representing distinct entities and relationships. Adapting GCNs for such diversity is challenging, as it demands multi-dimensional information extraction and fusion for accurate node embeddings. Previous GCN applications [20] typically process only one view of these networks or test each one to identify the best fit. These methods fail to leverage the complete, multifaceted nature of

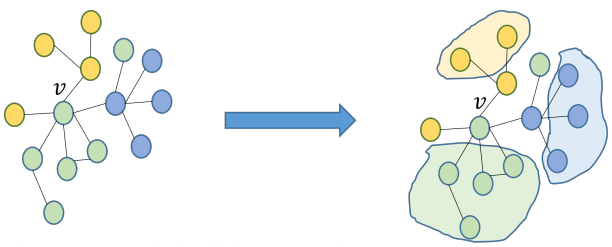


Fig. 1. Process of classifying aggregation.

Manuscript received November 18, 2023; revised April 11, 2024.

This work was supported in part by the Science Foundation of Fujian Province (No. 2021J011188), Xiamen Overseas Returnees Program under Grant (XM202017206), and the XMUT Scientific Research Project (No. YKJCX2021079).

Kaibiao Lin is a professor and master tutor of Department of Computer Science and Technology, Xiamen University of Technology, Xiamen 361024, China (e-mail: 2010110706@t.xmut.edu.cn).

Runze Chen is a postgraduate of Department of Computer Science and Technology, Xiamen University of Technology, Xiamen 361024, China (e-mail: 2122031199@s.xmut.edu.cn).

Jinpo Chen is a postgraduate of Department of Computer Science and Technology, Xiamen University of Technology, Xiamen 361024, China (corresponding author to provide e-mail: 18959237151@163.com).

Ping Lu is an associate professor and master tutor of School of Economic and Management, Xiamen University of Technology, Xiamen 361024, China (e-mail: 2011990101@t.xmut.edu.cn).

Fan Yang is an associate professor and master tutor of Department of Automation, Xiamen University, Xiamen 361005, China (corresponding author to provide e-mail: yang@xmu.edu.cn).

data in heterogeneous networks. Second, GCNs and their variants typically assume that neighboring nodes share similar representations and class affiliations. This assumption leads GCNs to treat all neighbors equally during aggregation, which can dilute the quality of the resulting node representations. Fig. 1 indicates that there are three types of nodes in the network, and direct aggregation of the central node v without processing introduces much noise from blue and yellow nodes, which is unsuitable for learning node embedding. In this regard, designing aggregation mechanisms that allow the central node to adaptively discern between homophilic and heterophilic neighbors is required, thereby facilitating the node's capability to achieve classifying aggregation. Such mechanisms are designed to enhance the influence of homophilic nodes during the aggregation process while simultaneously reducing noise from other types of nodes.

To enhance GCNs' aggregation mechanism, we introduce the multi-view block matrix-based graph convolution algorithm, MVGCN. This algorithm incorporates a block matrix within the GCN framework to implement block-guided classifying aggregation, which enables the automatic learning of specific aggregation rules for neighbors belonging to different classes. By incorporating block modeling into the aggregation process, GCN can selectively aggregate information by distinguishing between neighbors based on homophily degrees. Further, we design a multi-view module, where each view represents a neighbor graph under various semantics. Corresponding weighted neighbor matrices are derived from the block matrix. Then, the final embedding is obtained by aggregating information from all views using the attention module. This process not only merges insights from different views but also selectively integrates the most relevant information at the node level based on category proximity. This study has several contributions:

(1) For heterogeneous graph data with multiple semantic graph structures, a multi-view fusion method is proposed at the semantic level, eliminating the drawback that GCNs can only use a single view.

(2) At the node level, we introduce block matrix-guided classify aggregation. This approach addresses the shortcomings of GCNs in directly aggregating neighbors and tackles the challenges posed by the homophily assumption.

(3) MVGCN outperforms previous baselines in node classification, node clustering, visualization, and ablation experiments on real datasets (both homogeneous and heterogeneous graph data).

II. Related Works

A. Homology and Heterology

Homology refers to the fact that interlinked nodes typically belong to the same class or possess similar characteristics, e.g., friends who follow each other in a social network can share similar interests, and papers that cite each other belong to the same field. However, real-world networks often exhibit heterogeneity, linking nodes from diverse classes or with distinct characteristics. For example, different types of amino acids in protein structures are more likely to constitute links, and most people in dating networks prefer to communicate with those of the opposite sex.

To counter the limitations of the homology assumption, strategies generally fall into two categories. The first is higher-order neighbor aggregation. Models like H2GCN [21] and HAN [22] proposed that first-order neighbors might exhibit heterogeneity, but aggregating higher-order neighbors tended to reveal homogeneity. This method effectively addressed direct neighborhood heterogeneity. The second category is weighted aggregation. Approaches such as GPR-GNN [23] and GAT [24] assigned weights to neighbors based on node similarity or the relevance of neighbor information. During aggregation, graph neural networks gave more weight to similar neighbors, enhanced the relevance of node information and reduced noise from less similar neighbors.

B. Meta-Paths and Multi-Views

Fig. 2 (a) depicts an example of a heterogeneous graph network consisting of users, movies, directors, and actors. Building on this foundation, the concept of a meta-path emerges as a crucial tool for capturing semantic information in heterogeneous graphs. Specifically, the interaction between users and movies represents a dual relationship of watching and being watched. Similarly, the connection between directors and movies shows the roles of directing and being directed, while actors and movies are linked through participation, with actors taking part in movies and movies featuring these actors. A meta-path is a concept for capturing semantic information in heterogeneous graphs. The meta-path relationship between nodes A_1 and A_{l+1} is defined as $A_1 \xrightarrow{R_1} A_2 \xrightarrow{R_2} \ldots \xrightarrow{R_l} A_{l+1}$,which describes the complex relationship $R = R_1 \circ R_2 \circ \ldots \circ R_l$ between the node types A_1 and A_{l+1}. Here, $\circ$ signifies the relationship composition operator is a compound operator on the relationship. Meta-path-based methods are the most widely used methods for heterogeneous graph embedding. They are used to identify semantically related neighborhoods. By doing so, these meta-paths, which are ordered and symmetric, facilitate a deeper understanding of node connections. For example, in Fig. 2 (b), meta-paths like User-Movie-User (UMU), User-Movie-Director-Movie-User (UMDMU), and User-Movie-Actor-Movie-User (UMAMU) describe different relationships between users. In more detail, UMU searches for users who watch the same movie. UMDMU connects users through a director's

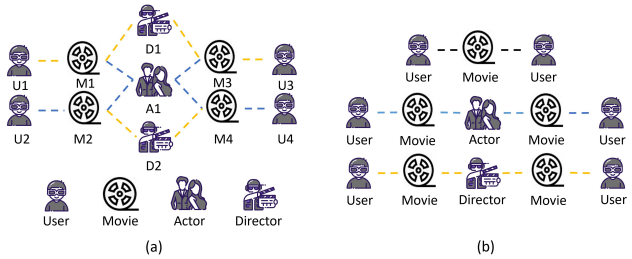


Fig. 2. Heterogeneous networks and meta-paths. Here, 'U' stands for user, 'M' for movie, 'A' for actor, and 'D' for director. 'UMU' indicates the meta-path user-movie-user.

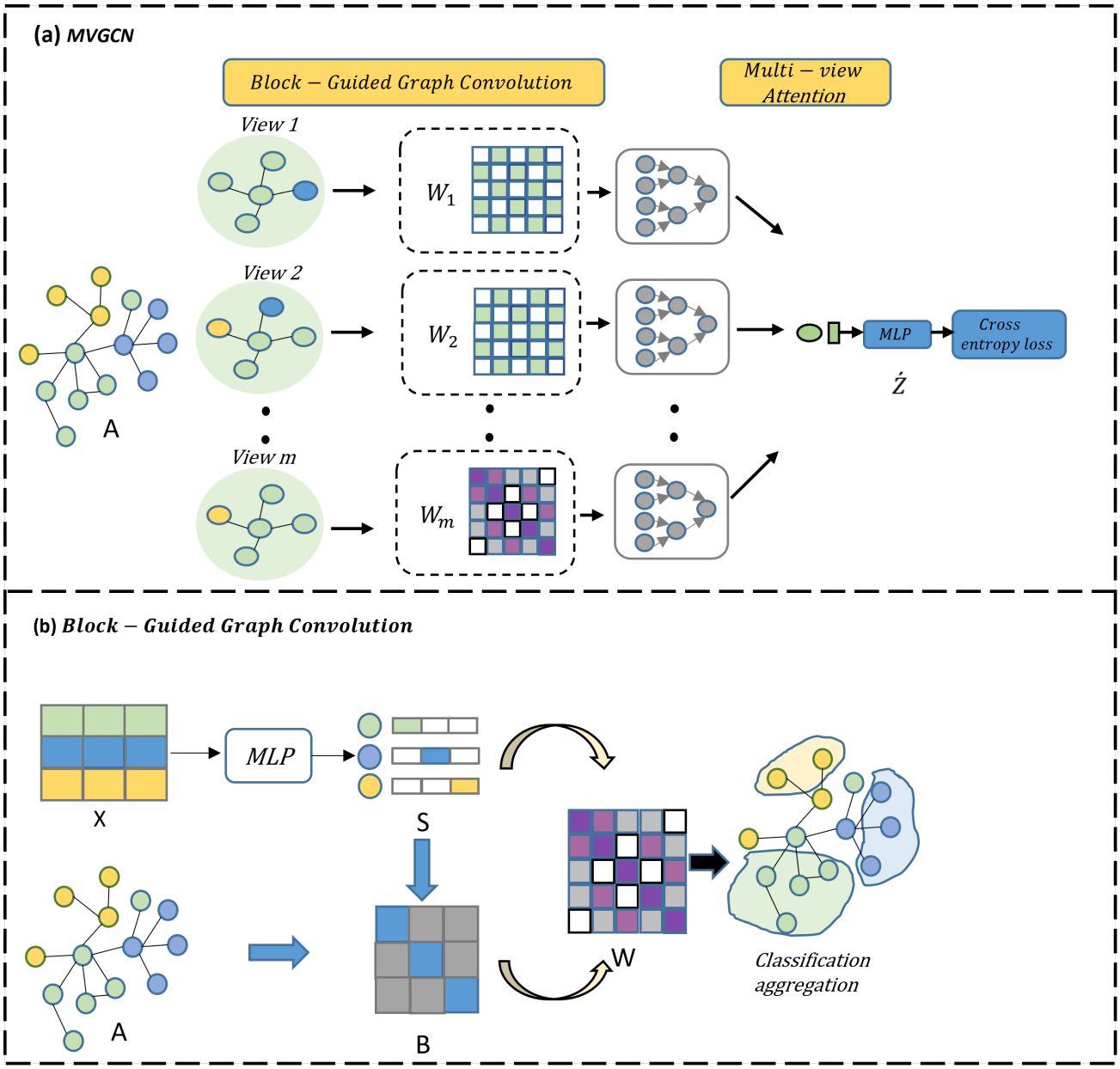


Fig. 3. The MVGCN model: Panel (a) comprises a block-guided graph convolutional network and a multi-view fusion module. This architecture distinctively processes diverse views through block matrix-guided graph convolutional networks. The final embedding is attained by integrating the embedding representations of each view via a view-level attention fusion mechanism. Panel (b) utilizes an MLP to generate soft labels for nodes. These labels are subsequently integrated with the network's topological structure to ascertain the block matrices. Finally, the weight matrices are calculated based on the soft labels and the block matrices, culminating in classification aggregation.

filmography. UMAMU aligns users with a preference for the same actor. Hence, meta-paths effectively serve as guides to higher-order nearest neighbors, enhancing node relationship understanding through inherent semantic significance.

In heterogeneous graphs, a node's local structure varies with different relationship types. Different meta-paths can reveal specific structural views of the network, and using multiple meta-paths simultaneously can decompose the heterogeneous graph into multi-view subnetworks. Fusing these multi-view perspectives, which considers diverse semantic information, can significantly enhance graph information utilization. graph information utilization. For instance, SemiGNN [25] developed a hierarchical attention mechanism. This approach assigned weights to different views and classified user fraud detection through multi-view data analysis. MV-GNN [26] proposed a cross-dependent information transfer scheme. It enhanced the information channel for atomic and chemically bonded views and introduced a shared self-attentive readout component and a divergence loss stabilization training process. Moreover, MAGCN [27] designed path encoders to capture the consistency of geometric relationships and probability distributions across views to find consistent clustered embedding spaces for multi-view adaptively.

C. Graph Embedding

Graphs play a crucial role in numerous real-world applications, offering insights into social structures, entity relationships, and communication patterns. This relevance has made graphs a focal point of academic research. Graph embedding learning, which involves distilling graphs' structural and semantic information into low-dimensional

node representations, is increasingly applied to various downstream tasks [28,29]. The graph embedding is evolving through integration with advanced technologies, expanding its application across various fields [30,31]. Several notable contributions in this domain include: Chen et al. [32] first designed a conflict detection-based metric, Totoro. It assessed the topological position of nodes by addressing the topological imbalance caused by marked nodes' asymmetry and inhomogeneity. Building on this, the ReNode method was developed to recalibrate node weights, tackling the challenges of topological imbalance. Moreover, Zhang et al. [33] proposed an effective robustness framework called RoHe. It mitigated perturbation amplification effects and soft attention mechanism issues by pruning unreliable neighbors based on topology and features through attention purifiers. Furthermore, Dou et al. [34] developed a CARE-GNN model to address feature and relationship camouflage in fraudulent networks. It leveraged label awareness to identify information-rich neighbors and reinforcement learning to determine optimal neighbor thresholds. Besides, Liu et al. [35] built the GNN framework GraphConsis to solve the inconsistency problem. It suggests integrating contextual embedding with node features to tackle contextual inconsistency and employs consistency scores to filter neighbors based on feature inconsistency. Additionally, it learns relational attention weights for sampled nodes to manage relational inconsistency.

III. MODEL

Fig. 3 illustrates that MVGCN consists of two modules: the block matrix-guided graph convolutional network, and the multi-view fusion module. For the block matrix-guided graph convolutional network module, the process begins with utilizing Multilayer Perceptron (MLP) to generate the soft labels of nodes. These labels, in conjunction with the network's topology, are used to determine the block matrix. Subsequently, the weight matrix is computed from the soft labels and the block matrix. This matrix is then used to guide the classifying aggregation by assigning different weights based on the network's homogeneity and heterogeneity. For the multi-view fusion module, each view is distinct due to the changes in graph structure neighbors. Hence, within each view, a unique weight matrix is determined to conduct classifying aggregation. The final embedding is derived by fusing the embedding representation of each view with view-level attention.

A. Pre-Training

To mitigate the limitations imposed by the homology assumption on GCNs, this study incorporates a block matrix into the GCN framework. Given the label matrix Y of the nodes and the adjacency matrix A of the graph network, the block matrix [20] is defined as follows:

$$B = (Y^T AY) \oslash (Y^T AE) \tag{1}$$

where E is an all-ones matrix of the same size as Y, and $\oslash$ is a Hadamard division operation. In this paper, blocks are the class of labels in the graph. The block matrix models the probability of linking nodes in any two blocks and $B_{(i,j)}$ is the probability of linking nodes in classes i and j at the node level.

The block matrix is a statistical modeling of the structural rules of the network. In order to obtain the block matrix, it is necessary to know all node labels. However, GCN is a semi-supervised learning model, and only a subset of labels is available. Addressing this, this model leverages known data to infer unknown labels, thus generating 'soft labels' based on node attributes. Specifically, MVGCN employs a multilayer perceptron to transform node attributes into soft labels.

Here, X is the node attribute, MLP is the multilayer perceptron, $\widetilde{S}$ is the output of MLP and $\sigma(\cdot)$ is the activation function. A softmax operation is performed on $\widetilde{S}$ to generate soft labels.

$$\widetilde{S} = \sigma\left(MLP\left(X\right)\right) \tag{2}$$

$$S = softmax\left(\widetilde{S}\right) \tag{3}$$

MVGCN pre-trains the MLP layer with actual labels to ensure the reliability of the soft labels S. The loss L_{MLP} is minimized during the training process, making the soft label S more closely resemble the actual label Y.

Here, S_i is the soft label of node v_i, Y_i is the actual label of node v_i, T_v denotes all nodes in the training sets, and $f(\cdot)$ is the cross-entropy loss function.

$$L_{MLP} = \sum_{v_i \in T_v} f\left(S_i, Y_i\right) \tag{4}$$

Replacing the true labels Y with the soft labels S in (1), we obtain (5). where S is the soft label matrix calculated from MLP, E is the all-ones matrix with the same size as S, and $\oslash$ is the Hadamard division operation.

The block matrix represents the connection pattern between classes. It is used for quantifying the graph's homology ratio. Specifically, a higher frequency of connections between nodes with identical soft labels directly correlates with an elevated homology ratio in the graph.

$$B = \left(S^T AS\right) \oslash \left(S^T AE\right) \tag{5}$$

B. Block Similarity Matrix

The block matrix B describes the likelihood distribution of node connectivity between any two classes. However, this matrix cannot directly guide the GCN to classify aggregation, especially in heterogeneous graphs. In such graphs, edges often link dissimilar classes more frequently than they connect similar ones. To address this, it is necessary to modify the values of elements in the block matrix B to facilitate the propagation of valuable information during the graph convolution process. For this purpose, we introduce the block similarity matrix Q, defined as:

$$Q = BB^T \tag{6}$$

The block similarity matrix Q measures the degree of similarity between each block in B, indicating that classes with similar structures will have more information

dissemination. In addition, nodes in the same class should have more information exchanges. To achieve this, the elements on the main diagonal, represented by $Diag(Q)$, are thus augmented by an enhancement factor α.

$$Diag(Q) \leftarrow \alpha \cdot Diag(Q) \tag{7}$$

C. Graph Convolution Propagation

MVGCN can assign different information dissemination rules for different class combinations based on the block similarity matrix Q. Meanwhile, soft labels can indicate the specific class combination to which any pair of nodes belongs. Thus, the information dissemination process can be guided by the block similarity matrix Q and the soft label matrix S. Specifically, for two nodes V_i and V_j with soft labels $S_i = \{s_i^1, s_i^2, \ldots, s_i^c\}$ and $S_j = \{s_j^1, s_j^2, \ldots, s_j^c\}$, respectively, where c is the number of classes, the node pair (v_i, v_j) has c^2 class combinations and the possibility of each class combination is shown as follows:

$$p\left(\varphi(v_i) = S_r, \varphi(v_j) = S_t\right) = s_i^r s_j^t \tag{8}$$

where $\varphi(\cdot)$ is the mapping function of the node's class it belongs to, and $r, t \in \{1, 2, \ldots, c\}$.

The block similarity matrix Q represents the information propagation probability between any two classes. The greater the similarity between the two classes, the more information should be propagated. The propagation probability w_{ij} between node pairs v_i and v_j can also be regarded as the propagation expectation of the elements in Q.

$$w_{ij} = \sum_{r=1}^{c} \sum_{t=1}^{c} q_{r,t} s_i^r s_j^t \tag{9}$$

Here, $q_{r,t}$ is an element in Q that represents the propagation probability between class r and class t.

The propagation weights between node pairs are jointly guided by the block similarity matrix Q and the soft label matrix S. The graph's propagation weights for all node pairs can then be described as a weight matrix. Equation (10) is the matrix expression of (9).

$$W = SQS^T \tag{10}$$

Next, the weight matrix W is utilized to optimize the topology of the graph network.

$$\tilde{A} = W \odot (A + I) \tag{11}$$

Here, I is the unit matrix, $\odot$ is the Hadamard multiplication operation, and $\tilde{A}$ can be viewed as a weighted adjacency matrix for which a softmax operation is performed to normalize the weights of the edges. Elements $\hat{a}_{i,j}$ in matrix $\hat{A}$ can be calculated as:

$$\hat{a}_{i,j} = \frac{\exp(\tilde{a}_{i,j})}{\sum_{v_k \in N} \exp(\tilde{a}_{i,k})} \tag{12}$$

where $\tilde{a}_{i,j}$ is an element in $\tilde{A}$. In MVGCN, the traditional adjacency matrix A used in the original GCN propagation is substituted with the normalized adjacency matrix $\hat{A}$. This alteration enables the graph convolution operation in MVGCN to perform classify aggregation. Thus, the new graph convolution propagation is defined as follows. Z^l is the node representation at layer l. W_1^l and W_2^l are the learning parameters at layer l. Notably, when l is layer 0, $Z^0 = X$.

$$Z^l = Z^{l-1} W_1^l + \hat{A} Z^{l-1} W_2^l \tag{13}$$

D. Multi-View Fusion

The graph structure can be resolved from multiple views in heterogeneous graphs. For example, in a movie network consisting of users, movies, directors, and actors, the meta-path user-movie-user (UMU) and user-movie-director-movie-user (UMDMU) represent fans of the same movie and users who adore the same director, respectively. The structure of the graph network varies when observed under different views, leading to the computation of distinct block matrices. These matrices then determine the corresponding weighted adjacency matrices, and the node embedding under each view is obtained by graph convolution propagation. After node-level aggregation, an additional layer of view-level aggregation is required to obtain the final node embedding.

Unlike heterogeneous graphs, homogeneous graphs only have a single view and cannot be fused with multiple views. In this case, a k-nearest neighbor graph is constructed for the nodes in all graphs. This graph, along with the original topological view, forms a feature view to make the proposed algorithm more inclusive. This approach involves computing a cosine similarity-based node similarity matrix, from which the k-nearest nodes are selected as neighbors for each node.

To determine the importance of each view, the multi-view fusion module first transforms the node embedding z_i of a particular view i through a layer of MLP exercises. Then, the importance of the view embedding w^i is determined as its similarity to the view-level attention vector, given as:

$$w^i = q^T \cdot \tanh\left(MLP(z_i)\right) \tag{14}$$

Following the determination of each view's importance, normalization is performed using the softmax function to derive the view weights, as outlined in (15):

$$a^i = \frac{\exp(w^i)}{\sum_{m \in M} \exp(w^m)} \tag{15}$$

where M is the total number of views. The view weights a^i can be interpreted as the degree of contribution of various views to a particular downstream task, and the larger

TABLE I
STATISTICS OF DATASETS

Dataset	Number of nodes	Number of relationships	Train	Test	Val	Class	Feature dimension
ACM	8916	12769	600	2125	300	3	1870
DBLP	27194	122393	200	3057	800	4	334
Cora	2708	10556	1192	497	796	7	1433
Pubmed	19717	88648	9463	3944	6310	6	500

the view weight, the more critical the view. In the final step, the learned view weights a^i are used as coefficients, and the node embedding representations Z_i under different views are weighted averaged to get the final node embedding representation $\hat{Z}$, as outlined in (16). By doing so, the final node embedding representation is more aligned with the views that hold larger weights.

$$\hat{Z} = \sum_{i=1}^{m} a^i \cdot Z_i \tag{16}$$

Similar to (4), a cross-entropy loss L_{GCN} is calculated between the final node embedding calculated by (16) and the actual label.

$$L_{GCN} = \sum_{v_i \in T_v} f\left(\hat{Z}_i, Y_i\right) \tag{17}$$

Finally, the embedding loss L_{GCN} is combined with the soft label loss L_{MLP} to obtain the final loss function L, as shown in (18).

$$L = \lambda L_{GCN} + \left(1-\lambda\right) L_{MLP} \tag{18}$$

Here λ is the balance parameter (default is 0.5). By minimizing L, we train all modules end-to-end.

IV. Experiments and Analysis

This section presents the baseline models for comparison and outlines the parameter settings of each model, alongside pertinent details about the four datasets used in this paper. In the following sections, we systematically conduct node classification, node clustering, model ablation, and visualization experiments. These tests aim to demonstrate the superiority and effectiveness of our proposed MVGCN across different downstream tasks. Notably, beyond its superior performance in heterogeneous graphs, MVGCN also results in better performance on two homogeneous graph datasets, which further illustrates the MVGCN's compatibility.

A. Experimental Settings

We test our MVGCN model against established benchmarks, including various classical graph neural network models. The baseline models used for comparison are shown as follows.

- GCN [4]: This model is a scalable semi-supervised learning method that performs convolutional operations on graph-structured data. Since the model is applied to homogeneous graphs, it was evaluated with each view and the best one was reported.
- GAT [24]: This model is a variant of GNNs designed for homogeneous graphs. It is a semi-supervised neural network that introduces an attention mechanism. Similarly, this experiment evaluated all views and selected the one with the best performance.
- HAN [22]: This model is a heterogeneous graph neural network based on hierarchical attention that includes node-level and semantic-level attention. It fully considers the importance of nodes and meta-paths.
- BM-GCN [20]: This model is also a variant of GNNs designed for homogeneous graphs. It introduces block modeling in the graph convolution operation and implements block-guided node classification aggregation.
- MVGCN: The model proposed in this study. It utilizes block matrix-guided graph convolutional networks for the node level aggregation to achieve classifying aggregation. At the semantic level, it implements multi-view fusion to improve the quality of node embedding.

For the experiments, we adhere to the default configurations for the GCN, GAT, HAN, and BM-GCN models, as these settings have been proven to yield optimal outcomes. To maintain consistency across models, the embedding dimension is uniformly set at 64. Regarding the MVGCN model develops in this research, we employ the Adam optimizer with key parameters configured as follows: a learning rate of 0.001, weight decay of 0.0005, a two-layer GCN architecture, and a dropout and balance parameter both set at 0.5. To optimize training efficiency, we implement an early stopping mechanism, ceasing training if no accuracy improvement is observed after 100 consecutive rounds.

B. Datasets

This study conducted experiments on four real datasets. Specifically, ACM [36] and DBLP [37] are heterogeneous graph networks, while Cora [38] and Pubmed [39] are two different homogeneous graph networks. All models were consistently trained using the training set nodes. Then the optimal model was selected using the validation set nodes, and the model performance was evaluated utilizing the test set nodes. Table I summarizes the number of nodes, number of relations, number of categories, number of training sets, number of validation sets, number of test sets, and feature dimensions for each dataset.

C. Node Classification

To validate the model's representational ability, Support Vector Machines (SVM) was used as the base classifier. Here, the SVM was chosen for its simplicity, efficiency, and broad applicability. Notably, the test set nodes were not involved in the optimization model in the graph neural network model, which means they were neutral in the model.

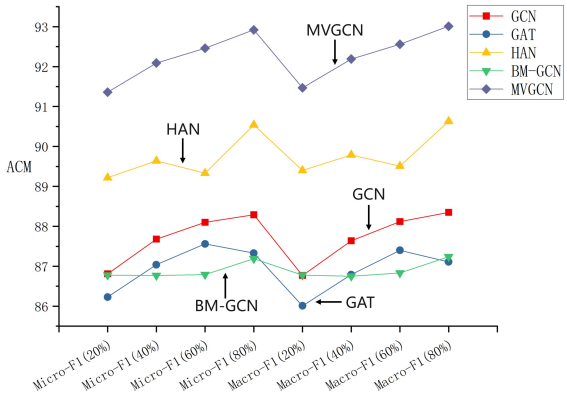


Fig. 4. Experiment results on the ACM datasets for the node classification task.

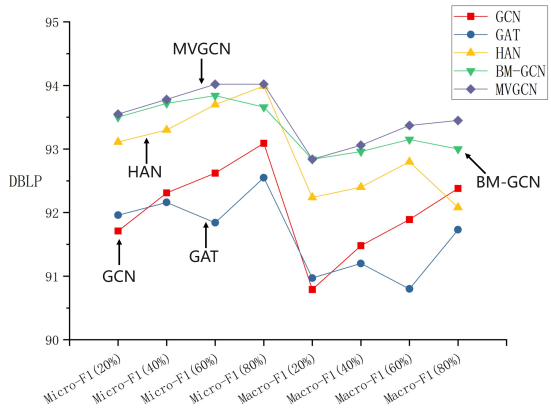


Fig. 5. Experiment results on the DBLP datasets for the node classification task.

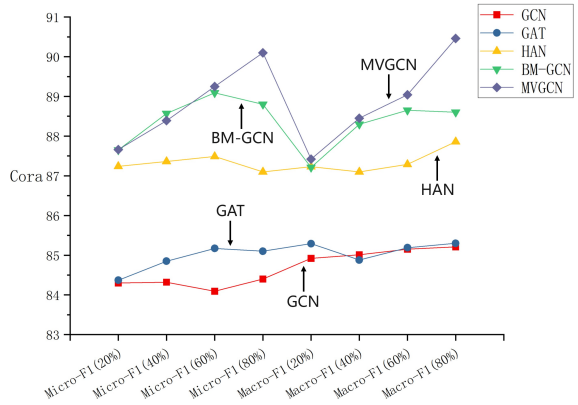


Fig. 6. Experiment results on the Cora datasets for the node classification task.

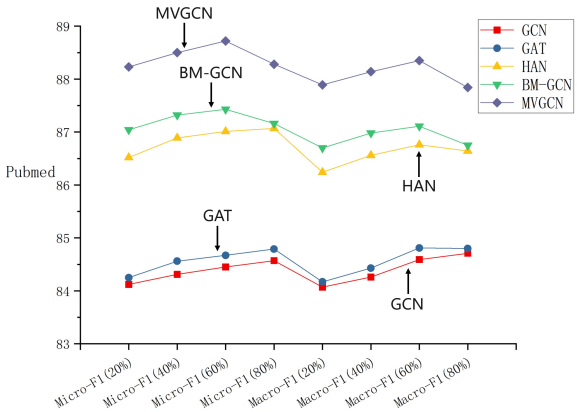


Fig. 7. Experiment results on the Pubmed datasets for the node classification task.

Therefore, incorporating these nodes as new entities into the SVM model validates the effectiveness of the graph neural network model in learning the node embedding. For a comprehensive evaluation, the test set nodes were re-split, and randomly selected the training set size to vary between 20% and 80% of the data. This process was repeated ten times, and the results were averaged to ensure reliability. We used macro-F1 and micro-F1 scores as our evaluation metrics.

TABLE II
NODE CLUSTERING TASK

Datasets	Metrics	GCN	GAT	HAN	BM-GCN	MVGCN
DBLP	NMI	75.01	71.43	79.26	75.30	**79.39**
	ARI	80.49	77.15	84.48	81.49	**85.18**
ACM	NMI	51.40	57.07	61.38	63.06	**71.16**
	ARI	53.01	60.42	64.46	66.15	**75.04**
Cora	NMI	56.97	70.05	74.87	72.90	**75.09**
	ARI	43.82	69.53	73.73	70.05	**75.43**
Pubmed	NMI	43.66	45.86	48.36	49.18	**50.23**
	ARI	47.18	48.24	50.67	50.09	**52.46**

Figs. 4 to 7 depict the performance comparison of various models on node classification across all datasets. Overall, the GCN performed the worst. This was because it treated all neighboring nodes equally, without differentiating their contributions from the central node. Such uniform aggregation of neighbor information resulted in significant noise. Conversely, GAT outperformed GCN on homogeneous graphs due to its attention mechanism. This feature enables selective weighting of neighbors, focusing on those most likely to improve node representation. However, its effectiveness was limited on heterogeneous graphs due to structural complexities. In addition to the node-level attention mechanisms of GAT, the HAN model included view-level attention mechanisms. It also features a two-layer architecture, enhancing compatibility with heterogeneous graphs. As a result, HAN outperforms both GCN and GAT. Similar to HAN, MVGCN adopts a two-layer architecture but goes further by adding a block matrix for improved classification, akin to BM-GCN. On the ACM dataset, the results of MVGCN are all more than 2% higher than the HAN model, which is the second-best result. However, unlike BM-GCN, which relies only on a block matrix and underperforms in multi-view scenarios like the ACM dataset, MVGCN enhances its architecture with multi-view fusion. This addition allows MVGCN to exceed the capabilities of BM-GCN, particularly in datasets with a large number of views. In addition, on the homogeneous graph PubMed dataset, the results of MVGCN essentially improve by more than 2% over the BM-GCN model, which is the second best in terms of results. In general, by combining a block matrix with multi-view fusion, MVGCN excels in both homogeneous and heterogeneous graphs, particularly exhibiting a significant lead over other GNN models on ACM and PubMed.

TABLE III
RESULTS OF MVGCN MODEL ABLATION EXPERIMENTS

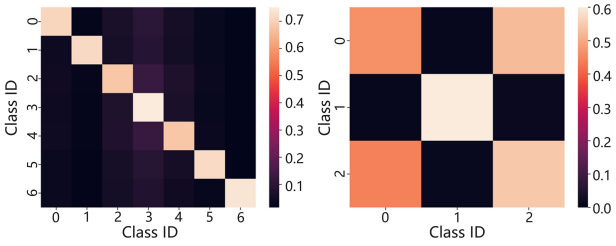
Datasets	Metrics	MVGCN-b	MVGCN-mv	MVGCN
DBLP	NMI	75.02	76.58	**79.39**
	ARI	80.85	82.34	**85.18**
	F1	93.82	93.86	**94.02**
ACM	NMI	69.55	69.73	**71.16**
	ARI	72.37	72.42	**75.04**
	F1	92.24	92.37	**93.01**
Cora	NMI	66.89	73.00	**75.09**
	ARI	55.68	71.35	**75.43**
	F1	88.10	88.90	**90.46**
Pubmed	NMI	49.40	48.34	**50.23**
	ARI	50.46	50.18	**52.46**
	F1	88.09	85.92	**88.28**

D. Node Clustering

Similar to node classification, K-means clustering algorithm was utilized for node clustering analysis. Here, we set the attributes of cluster centers as the number of categories in each dataset. Given K-means' sensitivity to initial center, iterations were repeated ten times to average the results. Then, NMI and ARI were used as evaluation metrics. Table II displays the outcomes of experiments conducted on all datasets. Similar to the node classification results, GCN and GAT have the worst outcomes because they only focus on the node level and do not consider that the graph structure is different in various views. On the other hand, HAN and BM-GCN introduce multi-view and block modules respectively, and have shown improved performance. On the ACM dataset, MVGCN outperforms the second-place BM-GCN model by over 8%, showing the good performance of MVGCN. In addition, MVGCN outperforms across all datasets, further illustrating the effectiveness of multi-view fusion and the necessity of classification aggregation.

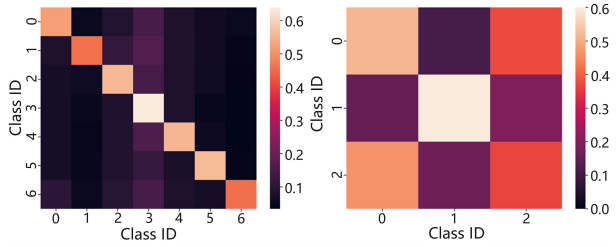
E. Ablation Experiments

This section presents further experiments with different variants of MVGCN to verify the effectiveness of the block similarity matrix and multiple views for the proposed model. The results were repeated ten times for averaging. Moreover, NMI, ARI, and F1 were used to determine the clustering and classification performance. Table III lists the outcomes across all datasets. In Table III, MVGCN-b indicates that the original adjacency matrix was utilized for the experiments rather than the block similarity matrix. Moreover, MVGCN-mv indicates the absence of the multi-view fusion module, where the best performing view in heterogeneous graphs was identified and used. The results in Table III demonstrate MVGCN's superior performance across all metrics and datasets, highlighting how the block similarity matrix and the multi-view fusion module help to improve the



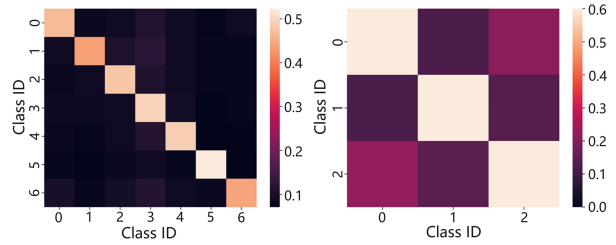
(a) $\widetilde{B}$ matrix on Cora dataset (b) $\widetilde{B}$ matrix on ACM dataset

Fig. 8. Block matrix calculated based on ground truth.



(a) B matrix on Cora dataset (b) B matrix on ACM dataset

Fig. 9. Block matrix calculated based on the method in this paper.



(a) Q matrix on Cora dataset (b) Q matrix on ACM dataset

Fig. 10. Block similarity matrix calculated based on the method in this paper.

embedding ability of the model nodes. Especially on the Cora dataset, the MVGCN demonstrated notably superior performance over the MVGCN-b model, with respective improvements of 8.2%, 19.75%, and 2.36% in the NMI, ARI, and F1 metrics. The slightly lower findings of MVGCN-b compared to MVGCN-mv indicate that it can be more critical to implement classifying aggregation to remove excessive noise in node-level aggregation than to introduce additional views in most cases.

F. Block Similarity Matrix Experiment

This study proposes MVGCN to achieve block-guided classification aggregation, so that nodes belonging to the same or similar classes have more information exchange. To facilitate this, MVGCN introduces a block matrix to model the relationship between classes. The elements in the block matrix B represent the possibility of a connection between various types of nodes. However, in heterogeneous graphs, where connections more frequently occur between different classes, the block matrix alone doesn't suffice for guiding GCN in classification aggregation. To address this, MVGCN further calculates the block similarity matrix Q through the block matrix, so that the model presented in this paper can simultaneously operate on graph

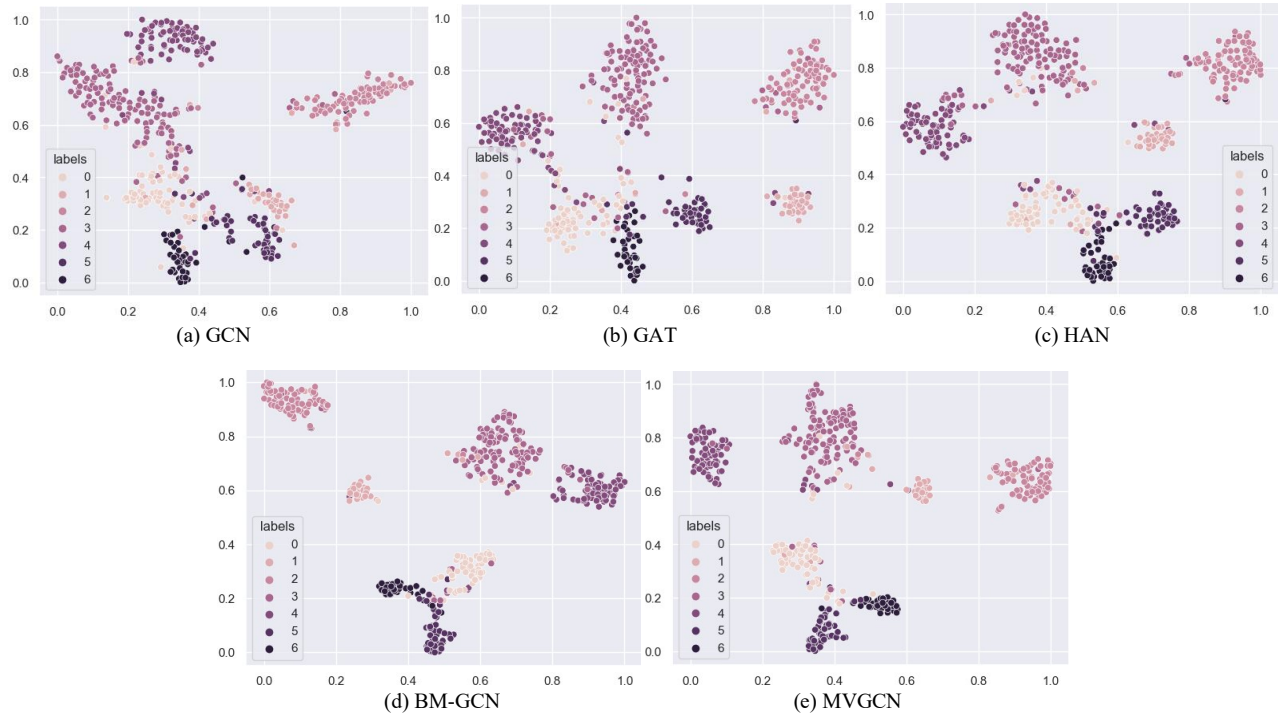


Fig. 11. Visualization results of different GNN models using t-SNE.

networks with homology and heterology. This study uses the homology graph Cora and the heterology graph ACM to illustrate the operation of the block similarity matrix.

Fig. 8 depicts the block matrix $\widetilde{B}$ of Cora and ACM calculated based on the ground truth labels. It indicates that the two matrices have different distribution patterns. In the homogeneous graph Cora, the connection possibility of nodes within the same class is relatively high, whereas in the heterogeneous graph ACM, the connection possibility of nodes between different classes is relatively high. Fig. 9 shows the block matrix B learned on the Cora and ACM datasets using the proposed method. Since the label is unknown in advance, MVGCN determines the soft label using (3) and then obtains the block matrix B . As illustrated in Figs. 8 and 9, regardless of the Cora or ACM dataset, the block matrix B calculated by MVGCN is always close to $\widetilde{B}$, demonstrating that the proposed method has a strong learning ability. In this case, the learned block matrix B can only be employed to aggregate more information on homology graphs and cannot be applied to heterology graphs. In the block matrix of ACM, larger off-diagonal elements still cause nodes to receive an excessive amount of noise during graph convolution, resulting in performance degradation. Therefore, this study uses the (6) to create a new block similarity matrix Q based on B , which can accurately measure the relationship between two classes from a new perspective. Two classes should be more similar if their graph connections are distributed similarly. As shown in Fig. 10, the diagonal elements of the block similarity matrix Q always have a larger value than that of off-diagonal elements. It successfully achieves classification aggregation on both homology and heterology graphs while preserving the original distribution of the block matrix, ensuring the stability of MVGCN performance across various datasets.

G. *Visualization*

A visualization task is performed to compare the representational abilities of different GNN models by depicting node embedding distributions in a two-dimensional space. Using the t-SNE method on the Cora dataset, Fig. 11 presents node embeddings method for each model. In more detail, panels (a) and (b) of Fig. 11 present that GCN and GAT have the worst performances as nodes of various classes are mixed together in the figure's lower left corner. In contrast, HAN and BM-GCN show improved results, indicating that both multi-view and block matrices contribute to the ability of node representation. Compared to the HAN model, MVGCN has tighter intra-class distances. Additionally, compared to BM-GCN, MVGCN has large inter-class distances. These highlight the superior efficacy of MVGCN's multi-view fusion and block similarity matrices in achieving more accurate categorical aggregation and leveraging information from multiple views for improved outcomes.

V. Conclusion

In this paper, we propose a novel method called MVGCN, aiming at refining the aggregation process in GCNs models by distinguishing among complex neighbor categories. Specifically, the MVGCN model enhances node-level aggregation by introducing a block matrix-guided graph convolutional network. Additionally, MVGCN creates a novel weighted adjacency matrix from block similarity and soft label matrices. This new matrix refines the network's topology, enhancing the model's ability to aggregate and classify nodes accurately. Moreover, MVGCN adapts to different graph topologies by calculating unique weight matrices for each view, leading to specialized node embeddings through graph convolution. These embeddings

are integrated using a view-level attention mechanism, ensuring a cohesive aggregation of multiple perspectives. Empirical results underscore the robustness of the algorithms introduced in this study. In tasks such as node classification, node clustering, ablation analysis, and visualization experiments, the MVGCN model demonstrates superior performance. The model's learned representations are versatile, proving beneficial for a range of downstream tasks in both heterogeneous and homogeneous graph contexts.

REFERENCES

[1] B. Perozzi, R. Al-Rfou, and S. Skiena, "Deepwalk: Online learning of social representations," in *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2014, pp. 701–710.

[2] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, "Line: Large-scale information network embedding," in *Proceedings of the 24th International Conference on World Wide Web*, 2015, pp. 1067–1077.

[3] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016, pp. 855–864.

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

[5] D. Bo, X. Wang, C. Shi, M. Zhu, E. Lu, and P. Cui, "Structural deep clustering network," in *Proceedings of the Web Conference 2020, 2020, pp. 1400–1410.*

[6] C. Huang, Y. Zhong, "A network representation learning method fusing multi-dimensional classification information of nodes," *IAENG International Journal of Computer Science*, vol. 50, no.1, pp94-105, 2023.

[7] J. Koskinen and T. A. Snijders, "Multilevel longitudinal analysis of social networks," *Journal of the Royal Statistical Society Series A: Statistics in Society*, vol. 186, no. 3, pp. 376–400, 2023.

[8] D. K. S. Singh, N. Nithya, L. Rahunathan, P. Sanghavi, R. S. Vaghela, P. Manoharan, M. Hamdi, and G. B. Tunze, "Social network analysis for precise friend suggestion for twitter by associating multiple networks using ml," *International Journal of Information Technology and Web Engineering (IJITWE)*, vol. 17, no. 1, pp. 1–11, 2022.

[9] A. Thangamuthu, G. Kumar, S. Bishnoi, R. Bhattoo, N. Krishnan, and S. Ranu, "Unravelling the performance of physics-informed graph neural networks for dynamical systems," *Advances in Neural Information Processing Systems*, vol. 35, pp. 3691–3702, 2022.

[10] A. A. Musa, A. Hussaini, W. Liao, F. Liang, and W. Yu, "Deep neural networks for spatial-temporal cyber-physical systems: A survey," *Future Internet*, vol. 15, no. 6, p. 199, 2023.

[11] W. Weng, J. Fan, H. Wu, Y. Hu, H. Tian, F. Zhu, and J. Wu, "A decomposition dynamic graph convolutional recurrent network for traffic forecasting," *Pattern Recognition*, vol. 142, p. 109670, 2023.

[12] J. James, "Graph construction for traffic prediction: A data-driven approach," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 9, pp. 15015–15027, 2022.

[13] L. Kaibiao, J. Chen, C. Ruicong, Y. Fan, Z. Yang, L. Min, and L. Ping, "Adaptive neighbor graph aggregated graph attention network for heterogeneous graph embedding," *ACM Transactions on Knowledge Discovery from Data*, vol. 18, no. 1, pp. 1–21, 2023.

[14] Y. Chang, C. Chen, W. Hu, Z. Zheng, X. Zhou, and S. Chen, "Megnn: Meta-path extracted graph neural network for heterogeneous graph representation learning," *Knowledge-Based Systems*, vol. 235, p. 107611, 2022.

[15] B. Hui, L. Zhang, X. Zhou, X. Wen, and Y. Nian, "Personalized recommendation system based on knowledge embedding and historical behavior," *Applied Intelligence*, pp. 1–13, 2022.

[16] S. Wu, F. Sun, W. Zhang, X. Xie, and B. Cui, "Graph neural networks in recommender systems: A survey," *ACM Computing Surveys*, vol. 55, no. 5, pp. 1–37, 2022.

[17] W Pan, K Yang, "Enhanced multi-head self-attention graph neural networks for session-based recommendation," *Engineering Letters*, vol. 30, no.1, pp37-44, 2022.

[18] Z. Li, Y. Zhao, Y. Zhang, and Z. Zhang, "Multi-relational graph attention networks for knowledge graph completion," *Knowledge-Based Systems*, vol. 251, p.109262, 2022.

[19] B. Koloski, T.S. Perdih, M. Robnik-Šikonja, S. Pollak, B. Škrlj, "Knowledge graph informed fake news classification via heterogeneous representation ensembles," *Neurocomputing*, vol. 496, pp. 208–226, 2022.

[20] D. He, C. Liang, H. Liu, M. Wen, P. Jiao, and Z. Feng, "Block modeling-guided graph convolutional neural networks," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, no. 4, 2022, pp. 4022–4029.

[21] J. Zhu, Y. Yan, L. Zhao, M. Heimann, L. Akoglu, and D. Koutra, "Generalizing graph neural networks beyond homophily," *arXiv preprint arXiv:2006.11468*, 2020.

[22] X. Wang, H. Ji, C. Shi, B. Wang, Y. Ye, P. Cui, and P. S. Yu, "Heterogeneous graph attention network," in *The World Wide Web Conference*, 2019, pp. 2022–2032.

[23] E. Chien, J. Peng, P. Li, and O. Milenkovic, "Adaptive universal generalized pagerank graph neural network," *arXiv preprint arXiv:2006.07988*, 2020.

[24] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," in International Conference on Learning Representations, 2018, pp. 1–12.

[25] D. Wang, J. Lin, P. Cui, Q. Jia, Z. Wang, Y. Fang, Q. Yu, J. Zhou, S. Yang, and Y. Qi, "A semi-supervised graph attentive network for financial fraud detection," in *2019 IEEE International Conference on Data Mining (ICDM)*. IEEE, 2019, pp. 598–607.

[26] H. Ma, Y. Bian, Y. Rong, W. Huang, T. Xu, W. Xie, G. Ye, and J. Huang, "Multi-view graph neural networks for molecular property prediction," *arXiv preprint arXiv:2005.13607*, 2020.

[27] J. Cheng, Q. Wang, Z. Tao, D. Xie, and Q. Gao, "Multi-view attribute graph convolution networks for clustering," in *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, 2021, pp. 2973–2979.

[28] X. Wang, D. Bo, C. Shi, S. Fan, Y. Ye, and S. Y. Philip, "A survey on heterogeneous graph embedding: methods, techniques, applications and sources," *IEEE Transactions on Big Data*, vol. 9, no. 2, pp. 415–436,2022.

[29] 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.

[30] H.-C. Yi, Z.-H. You, D.-S. Huang, and C. K. Kwoh, "Graph representation learning in bioinformatics: trends, methods and applications," *Briefings in Bioinformatics*, vol. 23, no. 1, p. bbab340, 2022.

[31] YH Zhang, JS Wang, and ZH Zhang, "Retinal vessel segmentation algorithm based on U-NET convolutional neural network," *Engineering Letters*, vol. 31, no.4, pp1837-1846, 2023.

[32] D. Chen, Y. Lin, G. Zhao, X. Ren, P. Li, J. Zhou, and X. Sun, "Topology-imbalance learning for semi-supervised node classification," *Advances in Neural Information Processing Systems,* vol. 34, pp. 29885–29897, 2021.

[33] M. Zhang, X. Wang, M. Zhu, C. Shi, Z. Zhang, and J. Zhou, "Robust heterogeneous graph neural networks against adversarial attacks," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, no. 4, 2022, pp. 4363–4370.

[34] Y. Dou, Z. Liu, L. Sun, Y. Deng, H. Peng, and P. S. Yu, "Enhancing graph neural network-based fraud detectors against camouflaged fraudsters," in *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, 2020, pp. 315–324.

[35] Z. Liu, Y. Dou, P. S. Yu, Y. Deng, and H. Peng, "Alleviating the inconsistency problem of applying graph neural network to fraud detection," in *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2020, pp. 1569–1572.

[36] X. Kong, P. S. Yu, Y. Ding, and D. J. Wild, "Meta path-based collective classification in heterogeneous information networks," in *Proceedings of the 21st ACM International Conference on Information and Knowledge Management*, 2012, pp. 1567–1571.

[37] M. Ji, Y. Sun, M. Danilevsky, J. Han, and J. Gao, "Graph regularized transductive classification on heterogeneous information networks," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2010, pp. 570–586.

[38] P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Galligher, and T. Eliassi-Rad, "Collective classification in network data," *AI Magazine*, vol. 29, no. 3, pp. 93–106, 2008.

[39] G. Namata, B. London, L. Getoor, B. Huang, and U. Edu, "Query-driven active surveying for collective classification," in *10th International Workshop on Mining and Learning with Graphs*, vol. 8, 2012, p. 1.