Boosting Performance of Graph Convolutional Networks via Generating Pseudolabels and Feature Interaction

Qiqi Zhang, Zhongying Zhao, *Member, IEEE*, Chao Li, and Xin Huang

***Abstract*—The task of node classification based on graph convolutional networks (GCNs) is widely applied in many social media and e-commerce websites. In this context, rich node labels undoubtedly play a very important role. However, existing methods fail to generate high-quality pseudolabels that serve as a potential solution to alleviate label scarcity on graphs and reduce time-consuming annotation issues. Moreover, it is an interesting problem to boost the performance of GCNs with feature interaction and propagation. To tackle these limitations, we propose a new pseudolabels generation and feature interaction propagation based GCN, named PF-GCN. Specifically, the proposed PF-GCN first generates high-quality pseudolabels to enlarge the input set of node labels. Meanwhile, it fully exploits the information of feature interactions to enhance our PF-GCN model. We then present a straightforward yet highly effective cross-layer approach which is equipped with feature transformation to encode feature interaction at each layer. Finally, we propagate feature interactions through both topology space and feature space, and then employ an adaptive attention mechanism to learn rich node embeddings. Experimental results demonstrate that the proposed PF-GCN outperforms several state-of-the-art methods in the extensive evaluations of node classification. We have released the source codes of PF-GCN for public usage and evaluation at https://github.com/ZZY-GraphMiningLab/PF-GCN.**

***Index Terms*—Feature interaction, graph convolution networks (GCNs), node classification, pseudolabels.**

Received 20 January 2024; revised 7 October 2024 and 28 November 2024; accepted 5 December 2024. Date of publication 2 January 2025; date of current version 4 August 2025. This work was supported in part by the National Natural Science Foundation of China under Grant 62472263 and Grant 62072288; in part by Taishan Scholar Program of Shandong Province; in part by Shandong Youth Innovation Team; in part by the Natural Science Foundation of Shandong Province under Grant ZR2024MF034 and Grant ZR2022MF268; and in part by the Ministry of Education in China Foundation for Humanities and Social Sciences under Grant 24YJAZH058. *(Corresponding author: Zhongying Zhao.)*

Qiqi Zhang, Zhongying Zhao, and Chao Li are with the College of Computer Science and Engineering, Shandong University of Science and Technology, Qingdao 266590, China (e-mail: zqqstu0818@163.com; zzysuin@163.com; lichao@sdust.edu.cn).

Xin Huang is with Hong Kong Baptist University, Kowloon Tong 999077, Hong Kong (e-mail: xinhuang@comp.hkbu.edu.hk).

Digital Object Identifier 10.1109/TCSS.2024.3516712

I. Introduction

GRAPH data [1] is prevalent in various scenarios, ranging from social networks [2], [3], [4], [5], recommendation systems [6], traffic networks to the World Wide Web. As one of the powerful deep learning models, graph convolutional networks (GCNs) [7] have proven to be effective in learning graph/node embeddings. Thus, they are widely used to tackle diverse graph analyzing tasks, such as node classification [8], [9], personalized recommendations [6], [10] and node clustering [9], etc.

As we all know, the performance of GCNs on node classification [11], [12], [13], [14], [15] largely depends on the availability of high-quality and abundant labeled data for training [16], [17]. However, acquiring labeled data is a time-consuming process that demands domain knowledge [18]. As a result, various semisupervised learning (SSL) methods have been proposed and gained significant attention [7], [11], [13], [19], [20]. These methods leverage both explicit information from fully labeled data and potential information among a substantial amount of unlabeled data. However, they are facing the following two challenges.

1) *How to Improve the Quality of Pseudolabels in Graph Representation Learning?* Generating pseudolabels is a popular semisupervised method for addressing the issue of label scarcity, and it has demonstrated impressive performance in learning image representation [21], [22], [23], [24], [25], [26], [27], [28]. These methods aim to overcome the limited availability of labels by predicting pseudolabels and iteratively refining the model with both real and pseudolabels. Inspired by the success of learning pseudolabels in computer vision (CV), similar methods have been proposed for learning graph representation [29], [30], [31], [32]. For instance, Iscen et al. [30] introduce a transductive label propagation strategy that leverages the nearest neighbor graph to infer and optimize pseudolabels. But they overlook higher-order nodes similar to the target node. MFGCN [32] calculates the similarity between labeled and unlabeled samples, and then selects the most similar ones as pseudolabels for unlabeled nodes. Nevertheless, it ignores the relationships between the labeled classes and the unlabeled nodes. Therefore, there is a crucial need to generate pseudolabels by considering the relationships among global nodes and their associations with labeled classes in the graph.

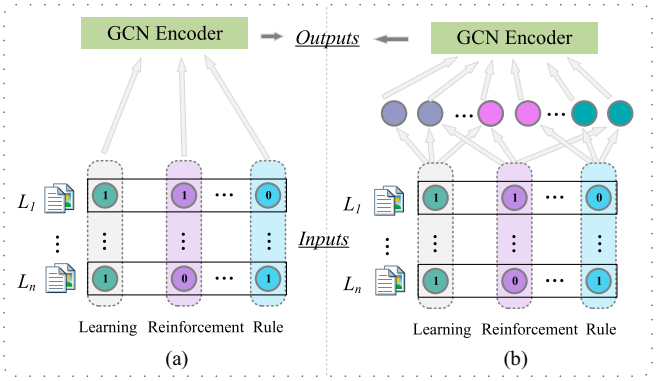

Fig. 1. Two different GCN frameworks. The traditional GCN (a) such as [7], [11], [13], [32], does not include feature interaction. Our method is similar to [33], [34] (b), which includes feature interaction in modeling. Solid-borderline boxes represent papers (L_1 and L_n), and dotted-borderline boxes represent attributes (learning, reinforcement, and rule).

2) *How to Design a Simple Yet Effective Strategy to Generate Feature Interaction?* In addition to labels, features also play an important role in learning node representations [13], [32], [35], [36], [37]. However, according to [34] and [33], the majority of existing GCNs scarcely capture feature interaction that can provide us with more comprehensive insights and is crucial for the success of many graph mining applications [38], [39], [40], [41]. Taking the citation network as an example, Fig. 1 contains two different GCN modules for propagating and aggregating node features. Fig. 1(a) illustrates the absence of feature interaction in the training process, where the feature is directly inputted into the traditional GCN model. Consequently, it becomes challenging to determine which class the literature L_1 and L_n belong to, despite both pieces of them containing the word "learning." In contrast, Fig. 1(b) demonstrates the process of feature interaction in our proposed method. For instance, the word "reinforcement" appears in the literature L_1, making it more likely to be in the field of "reinforcement learning" due to the feature interaction between "learning" and "reinforcement." Similarly, considering the feature interaction between "learning" and "rule," the literature L_n is most likely related to the topic of "rule learning." As a result, feature interaction should be taken into account while designing a representation learning model. For instance, cross-GCN [33] captures arbitrary-order feature interactions in graphs and calculates linearly increased parameters and time complexity. DFI-GCN [34] is capable of effectively handling arbitrary-order feature interactions by leveraging Newton's identities to address the issue of low efficiency in traditional feature interaction methods. While these methods exhibit the capability to capture feature interactions of arbitrary orders, they often encounter escalating computational costs as the feature dimension and interaction order expand. Consequently, how to design a simple yet effective strategy to generate feature interaction and comprehensively propagate it is a very interesting problem.

To address the above two problems, we present a pseudolabels generation and feature interaction propagation based GCN (PF-GCN). Specifically, we propose a method to generate pseudolabels based on the feature similarity between unlabeled and labeled nodes. This method serves as a plug-and-play module and can be applied to most existing GCN methods. We design a simple yet effective cross-layer mechanism to learn feature interactions. Additionally, PF-GCN learns node embeddings and employs an attention mechanism to adaptively fuse them in both the topology and feature spaces. Finally, we fuse the embeddings from both the topology and feature spaces and apply them to downstream tasks. The main contributions of this article are briefly presented as follows.

1) We propose a pseudolabel generation method to address the issue of label scarcity in graph data. This method calculates the similarity between the features of unlabeled nodes and both class-specific features and those of labeled nodes, thereby generating high-quality pseudolabels. It serves as a plug-and-play module, which can be easily integrated into most existing GCN models for convenient application.
2) We design a simple yet effective cross-layer to deal with the lack of feature interaction in existing GCN models. It enables to learn feature interaction information at each layer and integrates it into the node embedding. It captures the complex relationships between node features, thereby improving the expressiveness of node embeddings.
3) We incorporate a top-level attention mechanism in PF-GCN, which allows for the adaptive fusion of node embedding from the topology space and the feature space. With this fusion method, we comprehensively capture node information, thereby further enhancing the performance of node classification.
4) We compare PF-GCN with thirteen representative methods on six real-world datasets to evaluate its performance on the task of node classification. The experimental results demonstrate the superiority of PF-GCN, which achieves an average improvement of 5.28% and a maximum improvement of 8.92% against the state-of-the-art baselines.

The remainder of this article is organized as follows. Section II discusses some related work. Section III presents preliminary concepts and defines the problem. To clarify the differences of this work, we show the workflows in this section. Section IV describes the details of the proposed PF-GCN. Section V presents the performance of the PF-GCN on the task of node classification. Finally, Section VI concludes the article with some future directions.

II. Related Work

In this section, we introduce the related work of PF-GCN, which includes semisupervised GCNs, feature interaction, and pseudolabels generation.

A. Semi-supervised GCNs

Semisupervised GCNs have received significant attention from academia and industry [7], [11], [13], [16], [32], [33],

[42], [43], [44], and exhibit outstanding performance. For example, DeepWalk [45] is a representative graph embedding learning method, which learns the neighborhood relationships via a truncated random walk. Meanwhile, it can capture the information of low-dimensional embeddings. GCN [7] achieves a first-order approximation and superior performance in semisupervised node classification by efficiently aggregating neighbors' information using a standardized laplace matrix to learn node representations. Graph attention network (GAT) [11] performs a convolution operation by introducing the attention mechanism to specify different weights of neighboring nodes on the target node. MixHop [2] proposes to simultaneously aggregate feature information from first-order and higher-order neighbors in each layer. SCLR [46] is employed to learn the consensus representation of multispace by adding self-supervised loss to topology and feature spaces. AM-GCN [13] is another state-of-the-art method that fully integrates topology and node features of graphs. Specifically, it proposes an adaptive multichannel GCN as the core idea and learns the adaptive importance weights for merging embeddings from node features, topological structures, and their combinations. MFGCN [32] also designs an adaptive graph to fuse topology and feature graphs. Meanwhile, it proposes a fusion strategy on three graphs to learn a connected embedding. GPPO [47] integrates a GNN into the policy network, enabling a faster learning speed and better performance than the ordinary pure MLP-styled algorithm [48], [49]. We utilize multichannel GCN encoders to propagate representations. In different SCLR, AM-GCN, and MFGCN, we update features and add pseudolabels before training the model.

B. Feature Interaction

Feature interaction (*aka*. cross feature) can capture the relationship of features and deliver more accurate predictions. Traditional feature interaction methods are artificially designed, relying heavily on prior knowledge and unable to extend to new interaction patterns that have not yet appeared. This approach consumes a lot of resources and cannot achieve significant performance improvements [34]. Therefore, many studies are dedicated to enabling models to automatically and efficiently learn feature interaction patterns [34], [50], [51], [52], [53]. For example, factorization machine [50] (FM) incorporates dot products with deep neural networks to enhance performance. However, FM typically models low-order feature interactions. DeepFM [51] employs FM to extract feature interactions and feed them into deep neural networks to improve the model performance. DCN [52] exploits cross-network to capture high-level explicit feature interaction and achieve better performance. DCN-V2 [53] learns explicit and implicit feature interaction through cross layers. It inherits the simple cross-network structure of DCN but is more capable of learning explicit and bounded cross-over features. Cross-GCN [33] models feature crosses for graph-based learning problems with complexity linear to feature dimension and order size. In addition, cross features have been effective in improving the representation ability of GCNs. DFI-GCN [34] proposes to model arbitrary-order cross features with Newton's identities and perform pairwise interactions between adjacent nodes during convolution. They often encounter escalating computational costs as the feature dimension and interaction order expand. Currently, there are few GCNs that research feature interaction, which influences the representative ability of GCNs.

C. Pseudolabels Generation

Adding pseudolabels is an effective method to handle the problem of insufficient labeled data. It is predicted by Lee [54] calculating its class with the maximum probability, which is equal to entropy regularization through a network trained on the labeled nodes. Shi et al. [55] designed the confidence level for unlabeled samples to deal with uncertain samples and unreliable labels. Iscen et al. [30] trained the classifier via efficient transductive label propagation. It then constructs the nearest neighbor graph to optimize the model. Li et al. [29] proposed a self-training GCN that chooses top-K high-confidence unlabeled nodes to enlarge the label set for training. R2-D2 [28] is proposed to distinguish between predictions and pseudolabels. It then adopts the repetitive reproduction strategy to update pseudolabels. MFGCN [32] calculates the similarity between labeled and unlabeled samples. It then uses the labels of the most similar labeled nodes as the pseudolabels for unlabeled samples. InfoGNN [31] incorporates an MI-based informativeness measure for pseudolabel candidate selection to alleviate the negative impact of noisy pseudolabels.

III. Preliminary

In this section, we first define some key concepts. Then, we formulate the problems that this article aims to answer. Finally, we introduce the workflow of the traditional GCN paradigm and our proposed model. The notations mainly used in this article are summarized in Table I.

A. Problem Definition

Definition 1 (Attributed Graph [34]): Let $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{X})$ be an attributed graph, where $\mathcal{V}$ and $\mathcal{E}$ represent the sets of nodes and edges, respectively. $\mathcal{X} \in \mathbb{R}^{n \times d}$ represents the feature matrix of $\mathcal{G}$, where n is the number of nodes and d represents the feature dimension of each node. Moreover, $\mathcal{A} \in \mathbb{R}^{n \times n}$ represents the adjacency matrix of $\mathcal{G}$, where $\mathcal{A}_{ij} = 1$ indicates that there exists an edge between nodes i and j, while $\mathcal{A}_{ij} = 0$ otherwise.

Definition 2 (Partially Labeled Attributed Graph (PLAG) [16]): A PLAG can be represented as $\mathcal{G} = (\mathcal{V}_l, \mathcal{V}_u, \mathcal{E}, \mathcal{X}, \mathcal{Y}_l, \mathcal{Y}_u)$, where $\mathcal{V}_l$ and $\mathcal{V}_u$ represent the sets of labeled and unlabeled nodes, respectively, with $\mathcal{V}_l \cup \mathcal{V}_u = \mathcal{V}$. $\mathcal{Y}_l \in \mathbb{R}^{l \times c}$ denotes the one-hot labels for the first l labeled nodes, and the remaining nodes in $\mathcal{V}_u$ are unlabeled. c is the number of categories. The labeled set for nodes in $\mathcal{V}_l$ is denoted by $\mathcal{Y}_l$, whereas the predicted pseudolabels set for nodes in $\mathcal{V}_u$ is denoted by $\mathcal{Y}_u$.

Definition 3 (Class Feature (CX)): The class feature is defined as the average of features among nodes belonging to

TABLE I
NOTATIONS USED IN THIS ARTICLE

Notations	Descriptions
$\mathcal{G}$	The original graph as input.
$\mathcal{V}$	The set of nodes in graph $\mathcal{G}$.
$\mathcal{A}$	The adjacency matrix of $\mathcal{G}$ indicating the connection relationship between nodes.
$\mathcal{E}$	The set of edges in graph $\mathcal{G}$.
$\mathcal{X}$	The matrix including all nodes' feature in $\mathcal{G}$.
$\mathcal{X}_N$	The matrix including information of feature interaction.
x_i	The input features of the ith dimension.
n	The number of nodes in graph.
d	The feature dimension of each node.
$\mathcal{V}_l$, $\mathcal{V}_u$	The sets of labeled and unlabeled nodes in $\mathcal{V}$, $\mathcal{V}_l \cup \mathcal{V}_u = \mathcal{V}$.
$\mathcal{Y}_l$, $\mathcal{Y}_u$	The one-hot labels for the first l labeled nodes./ The predicted pseudolabels set for nodes in $\mathcal{V}_u$.
CX	The class feature of all labeled nodes.
c	The number of categories.
p	The certain label, $p \in \{1 \dots c\}$.
M	The number of labeled nodes in a certain class.
PL_{j1}, PL_{j2}	The candidate pseudolabels of the nodes.
PL_j	The predicted label of node j, which can be put in the set of pseudolabels.
Z_t, Z_f	The embeddings in topology and feature spaces.
α_t, α_f	The attention weight for the embedding in topology and feature spaces.
$X^{(N)}$	The feature interaction in the Nth cross-layer.
L_{real}, L_{pseudo}	The cross-entropy loss of two components.
$\tilde{\mathcal{Y}}$	The prediction result of node classification.
λ	The hyper-parameter to balance loss functions L_{real} and L_{pseudo}.

the same class within the labeled nodes. The k class feature is denoted as CX^k, and it is calculated as follows:

$$CX^k = \sum_{i=0}^{M} \frac{x_i^k}{M}, i = 1, \dots, M \tag{1}$$

where M represents the number of labeled nodes in a certain class, x_i^k represents the feature of the ith node in the kth class label, and $x_i \in \mathbb{R}^{1\times d}$.

Definition 4 (Feature Interaction [41]): The s-order feature interaction refers to a combination of s-dimensional input features, which can be represented as follows:

$$x^s_{t_1 t_2 \dots t_s} = \prod_{t \in \{t_1 t_2 \dots t_s\}} x_t \tag{2}$$

where x_t represents the tth dimension of the input features.

Problem: Semisupervised Node Classification. Given an attributed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{X}, \mathcal{Y}_l)$ where $\mathcal{V} = \mathcal{V}_l \cup \mathcal{V}_u$, the problem is to construct a learning model based on the existing labels $\mathcal{Y}_l$ of nodes $\mathcal{V}_l$, graph structure $\mathcal{E}$, adjacency matrix $\mathcal{A}$, and node features $\mathcal{X}$ to predict the class labels of unlabeled nodes $\mathcal{V}_u$.

B. The Workflow of Traditional GCN Paradigm and Our Proposed Model

The common traditional GCN paradigm is shown in Fig. 2(a) [7], [11]. To be specific, it first takes the original labeled nodes and node features as input. Then, it obtains the node embedding by the GCN encoder, which includes node aggregation and feature transformation modules. Node aggregation improves the representation of target nodes by aggregating the information from neighboring nodes. To more fully represent the target node, feature transformation projects the node embedding from the feature space to the latent space.

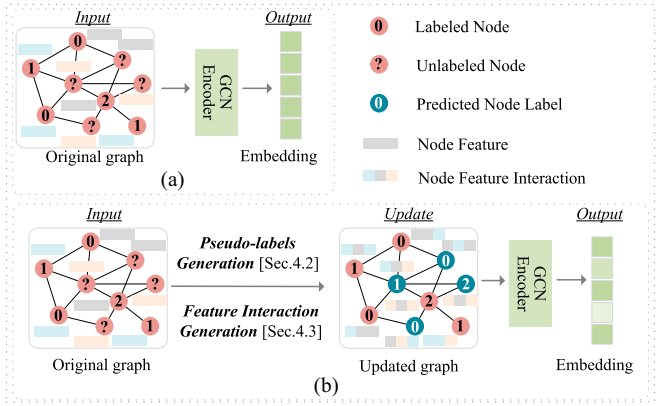

Fig. 2. Workflow of the traditional GCN paradigm and our proposed PF-GCN. (a) Traditional GCN. (b) Proposed PF-GCN.

The proposed PF-GCN framework adheres to the conventional GCN paradigm. In contrast to the traditional GCN, the PF-GCN model introduces pseudolabels and feature interaction as additional information in the multichannel GCN for model training. The workflow is illustrated in Fig. 2(b), where the original labeled nodes, pseudolabeled nodes, node features, and feature interaction information are inputted to the GCN model in both topology and feature spaces. More details about the proposed PF-GCN are described in Section IV.

IV. THE PROPOSED METHOD

A. Framework

In this section, we propose the pseudolabels generation and feature interaction propagation based graph convolution networks (PF-GCN). The overall framework of the proposed PF-GCN is shown in Fig. 3. The PF-GCN mainly consists of three modules.

1) *Pseudolabels Generation*: The primary objective of this module is to produce accurate pseudolabels. To achieve this goal, the process begins by generating candidate pseudolabels. This is accomplished through the calculation of feature similarity between unlabeled nodes and both class-labeled and labeled nodes. Subsequently, the determination of pseudolabels is based on evaluating the consistency of predictions stemming from these two distinct results. Finally, the acquired pseudolabels are employed in the model training process.
2) *Feature Interaction Generation*: We devise a cross-layer to acquire feature interaction information. Subsequently, we update both the features and the adjacency matrix, integrating them into the topology and feature spaces,

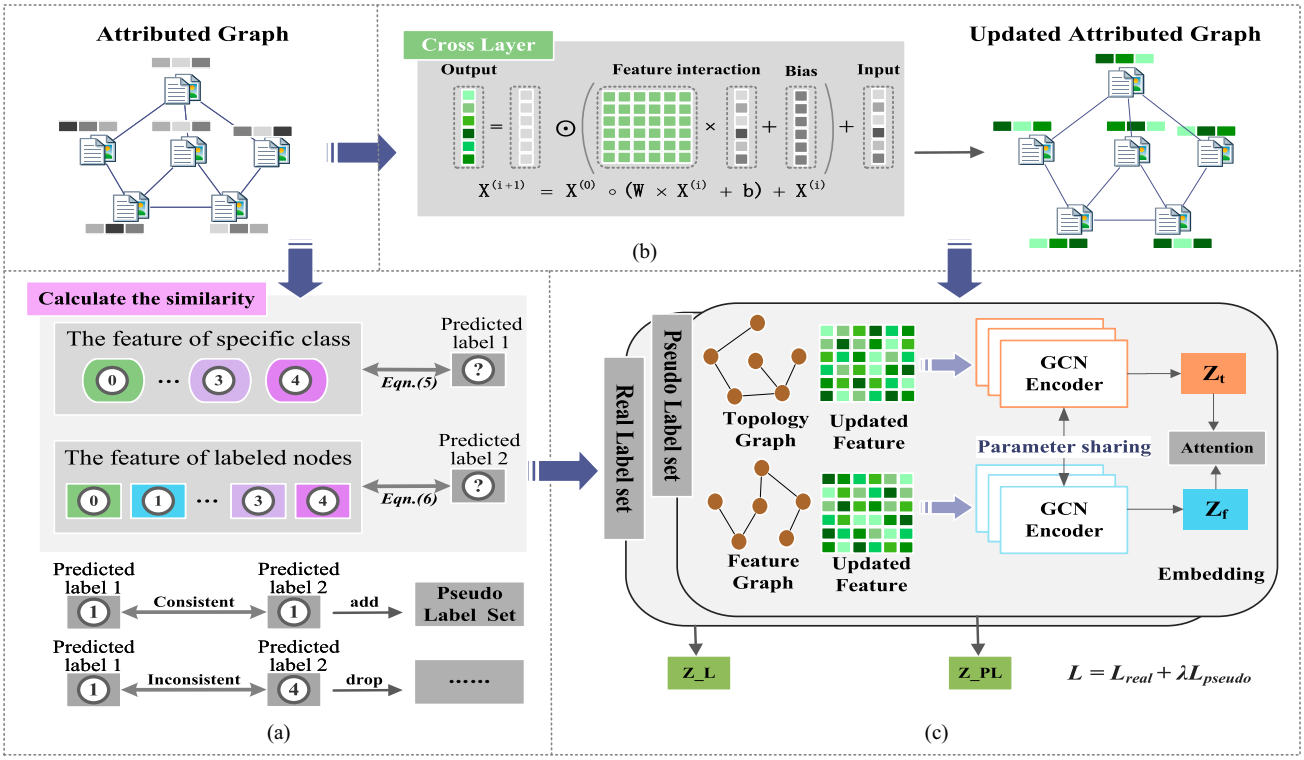

Fig. 3. Overall framework of the proposed PF-GCN model. (a) Pseudo label generation. (b) Feature interaction generation. (c) Propagation and optimization learning.

respectively. This process enables the generation of comprehensive node embeddings.

3) *Propagation and Optimization Learning*: Within this module, the convolutional layer in topology and feature spaces shares the same parameters. To enhance the learning process, an attention mechanism is utilized to adaptively capture embeddings from both real labels and pseudolabels. The embedding Z is generated via the modulation of hyper-parameter λ and the optimization of the loss function.

The specific details of the proposed PF-GCN are described in the following subsections.

B. Pseudolabels Generation

In this section, we propose a plug-and-play strategy for generating pseudolabels. The main process of generating pseudolabels is illustrated in Fig. 3(a), where different colors represent different labels. As depicted in Fig. 3(a), the candidates for pseudolabels are derived from two methods. The first method involves calculating the feature similarity between unlabeled nodes and each class. The class label with the highest similarity value is chosen as a candidate pseudolabel. The second method entails computing the feature similarity between unlabeled nodes and nodes that are already labeled. The label from the most similar labeled node is then selected as a candidate pseudolabel.

Specifically, the first step is to calculate the class feature. This is achieved by calculating the average of features among nodes belonging to the same class within the labeled nodes, are illustrated as follows:

$$CX^p = \sum_{i=0}^{M} \frac{x_i^p}{M}, i = 1, \ldots, M \tag{3}$$

where $p \in \{1, \ldots, c\}$ represents the certain label, c represents the number of categories, M represents the number of labeled nodes in a certain class, and i belongs to the set of labeled nodes $\mathcal{V}_l$ and $x_i \in \mathbb{R}^{1\times d}$.

With the class feature $\boldsymbol{CX}$ (4) at hand, the subsequent step is to calculate the feature similarity between unlabeled nodes and the class

$$\boldsymbol{CX} = CX^p, p = 0, 1, 2, \ldots, c \tag{4}$$

where X^p represents the feature of pth class.

Moreover, we calculate feature similarity with the cosine function, as shown as follows:

$$\begin{aligned} PL_{j1} &= \operatorname{argmax}\left(\cos\left(\boldsymbol{CX}, \boldsymbol{x_j}\right)\right) \\ \cos < x_i, x_j > &= \frac{x_i \cdot x_j}{|x_i||x_j|} \end{aligned} \tag{5}$$

where $j \in \mathcal{V}_u$. The label with the highest value in PL_{j1} is used as a candidate pseudolabels.

The second method for generating candidate pseudolabels is to compute the similarity of unlabeled nodes and labeled nodes. For each unlabeled node j in $\mathcal{V}_u$, we calculate $\cos(x_i, x_j)$ for every i in $\mathcal{V}_l$. A higher value of $\cos(x_i, x_j)$ indicates a

greater likelihood that these nodes belong to the same class. The calculation process is formulated as follows:

$$PL_{j2} = \text{argmax}(\cos < x_i, x_j >). \tag{6}$$

The final pseudolabel is determined by checking whether the two candidate labels are consistent or not, which is shown as follows:

$$PL_j = \begin{cases} 1 & PL_{j1} = PL_{j2} \\ 0 & PL_{j1} \neq PL_{j2} \end{cases} \tag{7}$$

where $PL_j = 1$ means the two candidate labels are consistent. We add the node and its label to the pseudolabels set.

C. Feature Interaction Generation

We take the original features as input of a cross-layer architecture, which is capable of producing information about feature interactions. Specifically, the training process of cross-layer is outlined as follows:

$$\mathbf{X}^{(N)} = \mathbf{X}^{(0)} \odot \text{MLP}(\mathbf{X}^{(i)}) + \mathbf{X}^{(i)} \tag{8}$$

where $X^{(0)} \in \mathbb{R}^d$ represents the base layer containing the original features and defaults to the initial value of the input layer. X^i, $X^{(N)} \in \mathbb{R}^d$ represent the input and output of the cross-layer, respectively. MLP$(\cdot)$, as an activation function, performs a nonlinear transformation.

Taking two cross-layers as an example, the process of generating feature interaction is shown as follows:

$$\mathbf{X}^{(1)} = \mathbf{X}^{(0)} \odot \left(\mathbf{W} \times \mathbf{X}^{(0)} + \mathbf{b}\right) + \mathbf{X}^{(0)} \tag{9}$$

$$\mathbf{X}^{(2)} = \mathbf{X}^{(0)} \odot \left(\mathbf{W} \times \mathbf{X}^{(1)} + \mathbf{b}\right) + \mathbf{X}^{(1)} \tag{10}$$

where $\mathbf{W} \in \mathbb{R}^{n \times d}$, $\mathbf{b} \in \mathbb{R}^d$ are the learnable weight matrix and bias vector, respectively.

Let's take the paper L_1 in Fig. 1 as an example. Three attributes are used to evaluate the feature interaction process: "Learning," "reinforcement," and "rule." This module first learns the supplementary representations between "learning" and "reinforcement," and then inputs the new interaction information and original features into the encoder, rather than just feeding attributes into it. This method is more effective in learning the embedding of the paper L_1.

D. Propagation and Optimization Learning

With feature interaction and pseudolabels at hand, we employ a GCN [7] as the encoder for aggregating node features. Recognizing the mutually reinforcing nature of the topology graph and the feature graph [13], we establish two distinct branches for propagation. This dual-branch approach aims to fully harness the complementary nature of topology and features in the node aggregation process. The process for propagating the representation and optimizing the model is illustrated in Fig. 3(c), where Z_F and Z_T represent the results of embedding learning achieved by the topology graph and the feature graph, respectively.

The topology graph is represented as $\mathcal{G} = (X_N, A)$, where X_N signifies the matrix capturing feature interactions, and $A \in \mathbb{R}^{n \times n}$ denotes the adjacency matrix of $\mathcal{G}$. The output Z_t^l can be expressed as as follows:

$$Z_t^l = \text{ReLU}\left(\tilde{D}_t^{-\frac{1}{2}} \tilde{A}_t \tilde{D}_t^{-\frac{1}{2}} Z_t^{l-1} W_t^l\right), Z_t^0 = X_N \tag{11}$$

where W_t^l represents the weight matrix of the lth convolution layer, and ReLU$(\cdot)$ is the activation function. $\tilde{A}_t = A + I_t$, I_t is the identity matrix. $\tilde{D}_t$ represents the degree matrix of $\tilde{A}_t$.

Similarly, to effectively capture node embeddings in the feature space, we construct the k-nearest neighbors (kNN) matrix which is denote as A_f. Thus, the configuration can be represented as $\mathcal{G} = (X_N, A_f)$. This approach is designed to emphasize relevant nodes in the feature space (12), further enhancing the quality of the node embeddings

$$Z_f^l = \text{ReLU}\left(\tilde{D}_f^{-\frac{1}{2}} \tilde{A}_f \tilde{D}_f^{-\frac{1}{2}} Z_f^{l-1} W_f^l\right). \tag{12}$$

Subsequently, the attention mechanism is used to combine the diverse embeddings that have acquired from both topology and feature spaces. The learned attention weights from these two perspectives are denoted as α_t and α_f. The attention values in the topology view and feature view for node i can be expressed as follows:

$$\omega_t^i = q^T \cdot \tanh\left(W \cdot \left(z_t^i\right)^T + b\right) \tag{13}$$

$$\omega_f^i = q^T \cdot \tanh\left(W \cdot \left(z_f^i\right)^T + b\right) \tag{14}$$

where q^T represents the shared attention vector, $\tanh(\cdot)$ is an activation function, and z_t^i and z_f^i are the embedding of the ith node.

The Softmax function [(15) and (16)] is used to normalize the attention values w_f^i and w_t^i and obtain the final weight

$$\alpha_t^i = \text{Softmax}\left(\omega_t^i\right) = \frac{\exp\left(\omega_t^i\right)}{\exp\left(\omega_t^i\right) + \exp\left(\omega_f^i\right)} \tag{15}$$

$$\alpha_f^i = \text{Softmax}\left(\omega_f^i\right) = \frac{\exp\left(\omega_f^i\right)}{\exp\left(\omega_f^i\right) + \exp\left(\omega_t^i\right)}. \tag{16}$$

To learn comprehensive and rich embedding Z, we utilize (17) to merge the results from the topology and feature graphs

$$Z = \alpha_t \cdot Z_t + \alpha_f \cdot Z_f \tag{17}$$

where $\alpha_t = \text{diag}(\alpha_t)$, $\alpha_f = \text{diag}(\alpha_f)$.

The embeddings learned under the actual labels are denoted as Z_L, while the embeddings learned under the pseudolabels are represented as Z_PL.

Before passing them to the softmax function, we apply a linear transformation to the get Z accroding to (18). The prediction results of classification are denoted by $Z' \in R^{n \times c}$

$$Z' = \text{Softmax}\left(W \cdot Z + b\right). \tag{18}$$

In the process of node classification, the computation of the cross-entropy loss is denoted as L and it is calculated as (19). As stated above, the loss consists of two components in which

Algorithm 1: The overall learning algorithm of PF-GCN.

Require: Adjacency matrix $\mathcal{A}$, feature matrix $\mathcal{X}$, labels Y_l, iterations τ, learning rate η, dropout ξ, hyper-parameters λ and p.
Ensure: Node embedding Z;
1: Initialize parameters τ, η, ξ, λ and p;
2: **for** $iter$=0,1,2,$\cdots$,τ-1 **do**
3: Generate the pseudo-labels to unlabeled nodes according to Eqn. (3)–(7);
4: Calculate and update feature interaction by Eqn. (8);
5: Generate the node embeddings of Z_t and Z_f in topology graph and feature graph by Eqn. (11) and Eqn. (12);
6: Calculate the adaptive attention coefficient by Eqn. (13) and Eqn. (15);
7: Generate node embedding by Eqn. (17);
8: Calculate the classification result $Z^{'}$ by Eqn. (18);
9: Calculate the overall loss according to Eqn. (20) and Eqn. (19);
10: Update hyper-parameters λ and p;
11: **end for**
12: Return Z;

L_{real} (20) is obtained from the labeled nodes, and L_{pseudo} (21), is obtained from the pseudolabels

$$L = L_{\text{real}} + \lambda L_{\text{pseudo}} \tag{19}$$

where λ represents a hyper-parameter to balance two loss functions L_{real} and L_{pseudo}

$$L_{\text{real}} = \sum_{i \in V_L} \ell\left(Z_i', \mathcal{Y}_i\right) \tag{20}$$

$$L_{\text{pseudo}} = \sum_{i \in V_{PL}} \ell\left(Z_i', \mathcal{Y}_i\right). \tag{21}$$

E. The Algorithm and Complexity Analysis

The algorithm of PF-GCN is presented in Algorithm 1. The main calculation consists of pseudolabels generation, feature interaction generation, propagation and optimization learning. We carefully analyze the complexity of the proposed PF-GCN. For the module of pseudolabels generation [from (3)–(7)], the time complexity is $\mathcal{O}(nd^2)$, where n is the number of nodes and d represents the dimension of nodes. While the time complexity of calculating feature interaction propagation is $\mathcal{O}(nd^2)$ (8). The computation cost of steps 6 to 8 mainly comes from the generation of node embeddings. Their complexity is $\mathcal{O}(fde)$, where f represents the number of GCN layers and e represents the number of edges $|\mathcal{E}|$ in the graph. Thus, the overall time complexity of PF-GCN is $\mathcal{O}(nd^2 + fde)$.

V. Experiments

In this section, we conduct a series of experiments to verify the performance of the proposed PF-GCN on the task of semisupervised node classification. We aim to answer the following four research questions.

RQ1: How does the proposed PF-GCN perform in the task of semisupervised node classification?

RQ2: How does the performance of GCN and GAT change when the pseudolabels module is utilized?

TABLE II
Dataset Statistics

Datasets	Nodes	Features	Links	Classes
Citeseer	3327	3703	4732	6
UAI2010	3067	4973	28 311	19
BlogCatalog	5196	8189	171 743	6
Flickr	7575	12 047	239 738	9
ACM	3025	1870	13 128	3
CoraFull	19 793	8710	65 311	70

RQ3: To what extent do the various modules contribute to the results?

RQ4: How does the classification performance vary with changes of hyperparameters?

A. Experimental Setup

1) Datasets: In the experiments, six widely used datasets, namely Citeseer, UAI2010 [56], BlogCatalog, Flickr, ACM, and CoraFull are selected for semisupervised node classification. Meanwhile, Citeseer and BlogCatalog datasets are used to to validate the module of pseudolabels. For each dataset, the number of nodes in the train, valid, and test sets is set the same as the previous works [13], [32].

2) Baselines: We carefully select thirteen representative methods, ensuring a diverse range of baselines both in terms of their underlying strategies and their chronological emergence. It is worth noting that these baselines utilize distinct approaches for learning node embeddings.

Specifically, these contain DeepWalk [45] that performs a truncated random walk, three traditional GNN-based methods: GCN [7], GAT [11], kNN-GCN [13], one multichannel and semisupervised learning method: AM-GCN [13], two self-supervised methods: SCRL [46] and GCA [57], one structure-preserving graph representation learning method: SP-GRL [58], and one recently published method for adding pseudolabels: MFGCN [32]. These baselines are briefly described as follows.

1) *DeepWalk* [45]: It conducts truncated random walks on a graph to capture multiple vertex sequences, which are treated as sentences.
2) *GCN* [7]: It constructs a convolutional layer by locally estimating the spectral graph convolutions to encode both local structures and node features.
3) *GAT* [11]: It is a semisupervised learning method that introduces the attention mechanism to specify different weights to aggregate neighbors features.
4) *kNN-GCN* [13]: It uses a sparse k-nearest neighbor graph to learn node embedding and calculates from the feature matrix as the input graph for GCN.
5) *MixHop* [2]: It performs a multiorder neighbor convolution by utilizing different aggregation methods for various neighbors.
6) *AM-GCN* [13]: It is an adaptive multichannel GCN that extracts embeddings from node features, topology, and their combinations.

7) *SCRL* [46]: It achieves a consistent representation of topology and feature spaces through self-supervised loss integration.
8) *GCA* [57]: It is designed to incorporates various priors for structure and semantic aspects of the graph by exploiting an adaptive data augmentation scheme.
9) *SPGRL* [58]: It preserves the global structure information and explores the local node-level relation.
10) *Cross-GCN* [33]: It models arbitrary-order cross features with linear complexity, relative to both feature dimension and order size.
11) *DFI-GCN* [34]: It utilizes Newton's identity to extract cross features of different orders and capture the pairwise interactions among nodes within the neighborhood.
12) *Info-GNN* [31]: It is an informative pseudolabeling framework that maximally represents the local neighborhoods through mutual information maximization to facilitate the learning of GNNs.
13) *MFGCN* [32]: It leverages the similarity between labeled and unlabeled samples and integrates a mechanism for generating pseudolabels.

3) Experimental Setting: We implement PF-GCN with PyTorch and accelerate the model training via NVIDIA RTX 3090 (24G) GPU. For the semisupervised classification task, we categorize the number of labeled nodes in each dataset into three levels: 20, 40, and 60 labeled nodes per class, denoted as L/C. To mitigate the impact of randomness on the results, we perform ten independent experimental runs for each dataset and report the average outcomes. We use accuracy (ACC) and macro F1-score (F1) to evaluate performance of models.

4) Parameter Setting: For the baselines, we closely follow the hyper-parameter settings of Deepwalk, kNN-GCN, and Mixhop in [13]. The parameters for SCLR and MFGCN are tested as suggested in [32]. For other baselines, we initialize the model using the parameters as suggested in the corresponding literature and then fine-tune them with grid search to attain the optimal performance. Specifically, in the MFGCN for generating pseudolabels, the parameter β is used to control the reliability of pseudolabels. It is set to 0.9 when L/C is 20, and to 0.99 when L/C is 40 or 60. Additionally, the parameter λ is set to 0.1 to balance the importance of the different components in the loss function. In generating feature interaction methods, the parameter k in DFI-GCN and cross-GCN represents the order of feature interaction. When k is set to 2, the model extracts second-order cross features, indicating the interaction information between any two features.

As for the proposed PF-GCN, we set the initial learning rate to 0.01, employing the Adam optimizer, and set the dropout rate to 0.5. We perform a grid search to identify the optimal learning rate and weight decay rate, with the search ranges set to $[e^{-5}, 5e^{-4}]$ and $[e^{-5}, 5e^{-3}]$, respectively. In addition, we conduct a comprehensive hyper-parameter search for the values of λ and p, spanning from 0.1 to 1, and utilize cross-validation to select the best values. To further optimize the model, we employ Taguchi's [59], [60], [61] experimental design method to systematically explore and determine the optimal combination of hyper-parameters. The specific values of these parameters are determined by hyper-parameter analysis, as specified in Section V-E.

B. Performance Comparison and Analyses (RQ1)

In this subsection, we compare and evaluate PF-GCN with thirteen representative methods on six benchmark datasets (UAI2010, Citeseer, BlogCatalog, Flickr, ACM, and CoraFull). Table III shows the experimental results for each method. In the following, we elaborate on the experimental results and make a detailed analysis and discussion.

1) GAT usually outperforms the traditional methods (DeepWalk, GCN, and Mixhop). One plausible explanation is that the attributes of nodes are high-dimensional sparse vectors. GAT utilizes an attention mechanism with a substantial number of parameters, contributing to enhanced accuracy in various datasets. The GCN-based methods (kNN-GCN, AM-GCN, SCRL, MFGCN, and PF-GCN) outperform GAT. Specifically, GCN-based methods exhibit improvements of 6.36% and 18.04% over traditional methods on UAI2010 and Flickr datasets. Similarly, GCN-based methods improve ACC and F1 by 2.57% and 26.97% on Citeseer and BlogCatalog datasets, respectively. AM-GCN, SCRL, and MFGCN are designed to learn node embedding by integrating the information of topology graph, feature graph, and their combinations. The main reason is that GCN-based methods can capture deeper relationships of nodes and design a multispace to fuse node embedding.
2) The proposed PF-GCN achieves the best performance compared with baselines. PF-GCN improves ACC and F1 by 0.9% and 9.15% on the UAI2010 dataset, 0.13% and 0.91% on the Citeseer dataset, compared with previous methods. On the BlogCatalog dataset, PF-GCN improves ACC by 1.6% and F1 by 1.09% with L/C = 40. Likewise, PF-GCN improves ACC by 0.8% and F1 by 1.61% on the Flickr dataset, compared with SPGRL and MFGCN. This observation demonstrates the effectiveness of pseudolabels and feature interaction. On the BlogCatalog dataset, MFGCN performs better than our proposed PF-GCN. The main reason is that it designs an adaptive graph to fuse the topology graph and feature graph. This adaptive strategy can automatically capture information between two graphs. PF-GCN does not achieve the optimal performance on ACM. The reason is that ACM is a subset extracted from a heterogeneous graph. The proposed PF-GCN is not good at handling heterogeneous information. PF-GCN improves ACC by 3.7% and F1 by 2.73% on CoraFull compared with the competitors. As a result, comparing it with the GCN-based model highlights the critical importance of feature interaction and underscores the significance of pseudolabels as additional information.
3) GCA and SPGRL outperform SCRL, attributed to their proficiency in fully leveraging local node-level relations

TABLE III
EXPERIMENTAL RESULTS ON THE TASK OF NODE CLASSIFICATION

Dataset	**UAI2010**						**Citeseer**					
L/C	20		40		60		20		40		60	
LR	12.39%		24.78%		37.17%		3.61%		7.21%		10.82%	
Metrics	ACC	F1	ACC	F1	ACC	F1	ACC	F1	ACC	F1	ACC	F1
DeepWalk (2014)	42.22	31.45	51.63	46.16	55.19	44.43	43.36	38.01	45.65	42.07	48.92	47.99
GCN (2017)	49.78	32.16	52.08	34.40	55.73	35.13	70.26	67.55	73.14	69.16	74.63	70.15
GAT (2018)	56.47	39.00	63.00	45.63	69.13	49.22	72.63	68.19	74.01	69.76	74.45	71.03
MixHop (2019)	61.90	49.84	65.55	54.03	68.24	56.15	71.31	66.04	71.23	66.99	71.68	68.71
kNN-GCN (2020)	66.36	52.03	68.24	54.15	71.70	54.11	61.16	58.60	61.93	59.67	62.57	60.12
AM-GCN (2020)	70.55	56.03	73.14	64.88	74.40	65.99	73.10	66.96	74.70	68.72	75.56	70.92
SCRL (2021)	72.90	57.80	74.58	67.40	74.90	67.54	73.62	69.78	75.08	70.68	75.96	72.84
GCA (2021)	72.55	56.97	73.27	54.55	73.60	56.00	71.39	68.46	72.96	68.02	73.92	69.10
SPGRL (2022)	76.30	61.49	78.20	68.73	79.80	71.38	75.19	70.98	**76.51**	**72.30**	76.21	73.14
Cross-GCN (2023)	73.19	61.72	76.27	64.36	77.52	67.45	70.05	68.45	73.20	69.82	75.30	70.87
DFI-GCN (2023)	72.69	60.91	76.18	64.17	76.59	66.92	73.51	70.03	75.28	71.13	76.14	72.27
Info-GNN (2023)	75.84	61.23	77.21	63.37	76.63	68.71	74.93	70.89	75.87	71.49	76.23	73.15
MFGCN (2023)	73.00	59.84	75.60	64.69	77.50	69.86	75.20	70.18	75.40	70.89	76.00	71.47
PF-GCN	**77.20**	**70.64**	**79.50**	**73.68**	**80.90**	**76.94**	**75.28**	**71.89**	76.23	71.67	**76.34**	**73.30**
Dataset	**BlogCatalog**						**Flickr**					
L/C	20		40		60		20		40		60	
LR	2.31%		4.62%		6.93%		2.38%		4.75%		7.13%	
Metrics	ACC	F1	ACC	F1	ACC	F1	ACC	F1	ACC	F1	ACC	F1
DeepWalk (2014)	37.53	34.37	50.87	46.70	55.23	53.01	24.30	21.42	28.86	27.12	30.33	27.12
GCN (2017)	69.91	68.83	71.40	70.84	73.22	72.10	41.92	39.98	45.00	43.06	48.13	46.77
GAT (2018)	65.01	63.43	67.66	66.84	69.99	69.22	38.52	37.01	38.63	36.82	38.66	37.14
MixHop (2019)	65.13	64.03	71.03	70.11	76.06	75.60	40.11	40.79	55.86	57.63	65.03	66.35
kNN-GCN (2020)	75.30	71.55	80.64	80.00	82.03	81.06	70.48	70.96	76.01	76.10	78.74	78.51
AM-GCN (2020)	81.98	81.36	84.32	84.32	87.30	86.94	75.26	74.63	80.06	79.36	82.10	81.81
SCRL (2021)	90.22	89.89	90.26	89.90	91.58	90.76	79.52	78.89	84.23	84.03	84.54	84.51
GCA (2021)	80.51	81.28	84.89	84.04	86.34	86.19	63.44	63.26	63.90	64.60	64.43	64.64
SPGRL (2022)	90.70	90.12	92.10	91.34	92.30	92.13	82.20	81.24	**86.20**	**85.93**	87.17	85.97
Cross-GCN (2023)	90.13	89.91	91.44	90.87	91.86	91.07	80.67	80.13	83.45	82.94	84.04	84.01
DFI-GCN (2023)	90.47	90.07	91.02	90.45	92.38	92.15	81.87	81.27	82.12	81.87	83.98	84.14
Info-GNN (2023)	89.94	89.79	90.37	90.18	92.35	92.27	80.57	80.50	82.11	81.35	83.41	83.26
MFGCN (2023)	**92.10**	**91.73**	92.40	92.72	93.00	93.08	81.20	81.37	84.90	85.05	87.20	**87.27**
PF-GCN	91.60	91.32	**94.00**	**93.81**	**93.84**	**93.64**	**83.00**	**82.98**	85.60	85.40	**87.30**	87.21
Dataset	**ACM**						**CoraFull**					
L/C	20		40		60		20		40		60	
LR	1.98%		3.97%		5.95%		2.31%		4.62%		6.93%	
Metrics	ACC	F1	ACC	F1	ACC	F1	ACC	F1	ACC	F1	ACC	F1
GCN (2017)	87.80	87.82	89.06	89.00	90.54	90.49	56.68	52.48	60.60	55.57	62.00	56.24
GAT (2018)	87.36	87.44	88.60	88.55	90.40	90.39	58.44	54.44	62.98	58.30	64.38	59.61
AM-GCN (2020)	90.40	90.43	90.76	90.66	91.42	91.36	58.90	54.74	63.62	59.19	65.36	61.32
Cross-GCN (2023)	92.10	91.59	91.97	91.84	92.17	92.18	60.58	57.62	63.55	59.10	65.93	61.55
DFI-GCN (2023)	92.27	92.48	92.49	92.11	92.91	92.74	61.34	57.17	64.61	58.64	66.35	61.97
Info-GNN (2023)	92.40	90.42	92.62	92.58	92.80	92.78	61.54	57.46	64.12	58.70	66.12	61.90
MFGCN (2023)	**92.70**	**92.64**	**93.00**	**92.99**	**93.11**	**93.00**	62.10	**57.66**	64.40	59.82	66.50	62.45
PF-GCN	92.49	92.30	92.20	92.18	93.10	92.88	**64.00**	57.37	**65.90**	**61.53**	**66.80**	**63.64**

Note: The best results are in bold and the second-best results are in underline. L/C refers to the number of labeled nodes in per class. LR refers to the proportion of labeled nodes.

through the feature graph. Furthermore, the feature view preserves high-order relations. AM-GCN and MFGCN demonstrate remarkable abilities to learn node embeddings by leveraging both graph topology and node features, with the flexibility to handle various combinations. However, a notable limitation of these methods lies in their neglect of the influence of feature interaction. In contrast, PF-GCN introduces an innovative

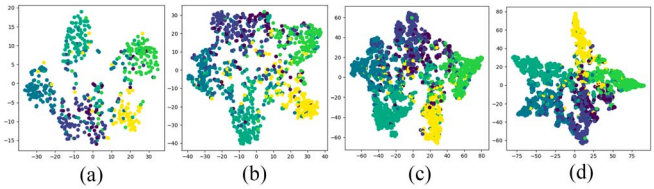

Fig. 4. Visualization of the embeddings on Citeseer dataset. (a) GCN. (b) GAT. (c) AM-GCN. (d) PF-GCN.

cross-layer that aims to capture and propagate feature interactions across a multichannel network. Compared with SCRL, PF-GCN exhibits competitive performance, primarily owing to its distinctive ability to assess feature similarity among unlabeled and labeled nodes. This comprehensive consideration of diverse factors empowers PF-GCN to achieve a superior level of performance.

4) Cross-GCN and DFI-GCN are both methods based on the GCN model that enhance model performance by capturing feature interactions. As shown in Table III, Cross-GCN and DFI-GCN exhibit superior performance compared with traditional GCN methods (GCN, GAT, kNN-GCN, and AM-GCN). PF-GCN outperforms these two methods in terms of performance. The main reason is that the learnable weights effectively mine higher-order feature interaction information. Info-GNN and MFGCN improve model performance via the design of pseudolabels. Experimental results indicate that their performance generally surpasses that of most semisupervised learning methods. This is attributed to the additional information provided by the generated pseudolabels. In particular, PF-GCN achieves outstanding performance on the UAI2010, Citeseer, and BlogCatalog datasets. This is mainly because the generated pseudolabels consider the class feature information, which ensures higher generation quality.

To provide a more intuitive demonstration of the performance, we utilize t-distributed stochastic neighbor embedding (t-NSE) [62] for visualizing the node classification outcomes on the Citeseer dataset. The visualization results are presented in Fig. 4, in which each dot corresponds to a node with the color indicating the respective class label. To offer a comprehensive model performance comparison, we select four different types of models for this analysis: GCN, GAT, AM-GCN, and PF-GCN. This selection encompasses a diverse range of models, allowing for a meaningful comparison of their respective node classification capabilities. The analysis of the visualization results yields the following conclusions.

The visualization result for GCN [Fig. 4(a)] is the least satisfactory, as it exhibits a significant mixing of different node types, lacking clear distinctions. GAT [Fig. 4(b)] showcases somewhat blurred boundaries among different node types, leading to an unclear separation of classes. Although the visualization outcome of AM-GCN [Fig. 4(c)] displays some improvement, certain nodes exhibit suboptimal clustering, indicating room for enhancement. In striking contrast, our proposed model [Fig. 4(d)] clearly outperforms the other methods, reaffirming the effectiveness of PF-GCN in achieving superior node classification results.

C. Pseudolabels Experiment (RQ2)

To demonstrate the efficacy of the pseudolabels module within PF-GCN, we integrate this module into two well-established models: GCN and GAT. We train these models and evaluate their performance on the Citeseer and BlogCatalog datasets. To investigate the effects of pseudolabels with different percentages, we incorporate pseudolabels ranging from 10% to 50% of all nodes, with values set at 10%, 20%, 30%, 40%, and 50%. Table IV shows the experimental results of node classification. According to these results, we can draw the following conclusions:

1) The increase in the number of nodes assigned pseudolabels consistently leads to total accuracy improvements, surpassing the baseline results. Generally speaking, as the proportion of pseudolabels increases, the accuracy of both GCN and GAT exhibits a positive trend. On the Citeseer dataset, both GCN and GAT exhibit a continuous upward trend. On the BlogCatalog dataset, GCN and GAT initially experience a rise in accuracy and F1-score, followed by a more stabilized trend. It is obvious that adding different proportions of pseudolabels performs better than the original traditional model.
2) GCN achieves an improvement of 12.12% and 11.79% on the Citeseer dataset, 3.44% and 3.01% on the BlogCatalog dataset in ACC and F1, respectively. Similarly, GAT achieves an improvement of 7.89% and 7.72% on the Citeseer dataset, 3.79% and 3.56% on the BlogCatalog dataset in ACC and F1, respectively. The main reason is that adding pseudolabels can provide the model with additional information about the nodes, which is beneficial to node classification task.
3) The results prove the effectiveness of generating pseudolabels as a plug-and-play module. Furthermore, we assess the accuracy of the generated pseudolabels, revealing impressive results of 82.57% on the Citeseer dataset and 95.57% on the UAI2010 dataset, respectively. It achieves an improvement of 23.67% and 83.44%, respectively. This observation represents a significant improvement compared with the previous method.

D. Ablation Study (RQ3)

We conduct an ablation study to evaluate the contribution of different modules to the PF-GCN. To this end, we compare the proposed PF-GCN with its three following variants on BlogCatalog, UAI2010, Citeseer, and Flickr datasets

1) *PF-GCN_pl*: removes the pseudolabels module.
2) *PF-GCN_cf*: removes the feature interaction module.
3) *PF-GCN_pl_cf*: removes the pseudolabels and the feature interaction modules.

The ablation results are shown in Fig. 5. We can find that the proposed PF-GCN achieves the best performance compared with the other three variants. This suggests that all the

TABLE IV
RESULTS OF ADDING DIFFERENT PROPORTIONS PSEUDOLABELS IN GCN MODEL ON NODE CLASSIFICATION TASK, HIGHLIGHTING THE BEST VALUE IN BOLD

Model	Dataset	Metric	-	+10%	+20%	+30%	+40%	+50%
GCN	Citeseer	ACC	63.31 ± 1.9	66.98 ± 1.9	70.61 ± 1.5	72.64 ± 1.1	74.56 ± 3.0	$\mathbf{75.45 \pm 1.5}$
		F1	61.09 ± 1.6	64.18 ± 1.5	67.55 ± 2.2	69.69 ± 1.0	71.72 ± 3.2	$\mathbf{72.88 \pm 1.7}$
	BlogCatalog	ACC	67.72 ± 6.9	69.43 ± 3.2	70.47 ± 6.4	67.32 ± 9.7	68.46 ± 5.9	$\mathbf{71.16 \pm 3.9}$
		F1	66.76 ± 6.8	67.30 ± 4.2	68.90 ± 7.1	69.10 ± 5.8	69.62 ± 5.8	$\mathbf{69.77 \pm 3.8}$
GAT	Citeseer	ACC	69.29 ± 2.7	71.84 ± 1.7	73.98 ± 0.8	74.50 ± 1.5	76.03 ± 1.4	$\mathbf{77.18 \pm 0.9}$
		F1	66.35 ± 2.7	68.92 ± 1.8	70.66 ± 0.7	71.60 ± 1.6	72.92 ± 1.4	$\mathbf{74.07 \pm 1.3}$
	BlogCatalog	ACC	69.00 ± 1.6	71.20 ± 3.8	71.82 ± 5.4	71.97 ± 2.9	$\mathbf{73.10 \pm 3.2}$	72.79 ± 3.2
		F1	68.03 ± 1.9	70.00 ± 3.8	70.59 ± 5.3	70.71 ± 1.9	$\mathbf{72.01 \pm 3.6}$	71.59 ± 2.6

Note: '-' indicates not adding pseudolabels. '+xx%' represents the proportion of added pseudolabeled nodes to the total number of nodes.

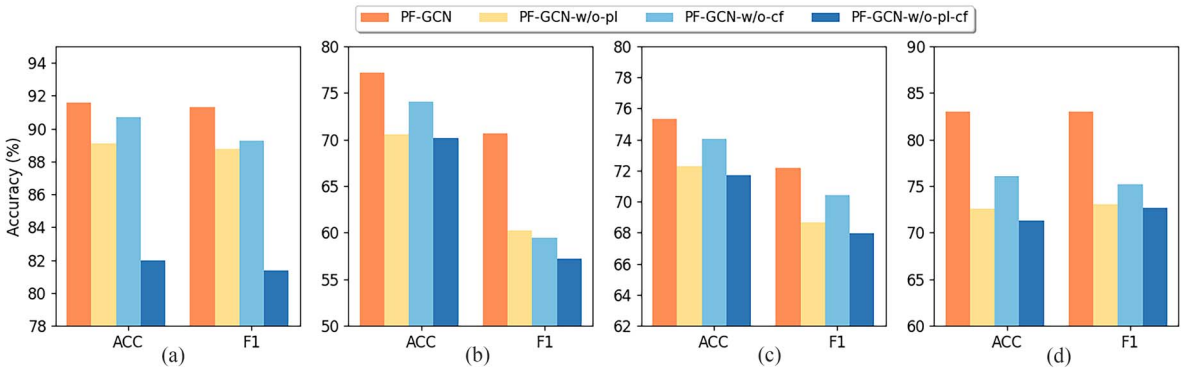

Fig. 5. Ablation experimental results of PF-GCN and its variants. (a) BlogCatalog. (b) UAI2010. (c) Citeseer. (d) Flickr.

components of PF-GCN are essential for boosting performance. *PF-GCN_pl* outperforms *PF-GCN_pl_cf*, which demonstrates that the pseudolabels module plays a significant role in the task of node classification. In addition, the efficiency of the pseudolabels module is also demonstrated in Table IV. *PF-GCN_cf* achieves the second-best performance on the BlogCatalog, Citeseer, and Flickr datasets in terms of ACC, and F1 and on the UAI2010 dataset in terms of ACC. This proves that feature interaction can be effectively represented in topology space and feature space. *PF-GCN_pl_cf* obtains the lowest effect on the two datasets, indicating the critical importance of the previously mentioned modules in achieving superior performance. The results demonstrate that the two modules of the proposed PF-GCN are beneficial for improving the effectiveness of classification.

E. Hyper-Parameter Sensitivity Analysis (RQ4)

Hyper-parameters have a significant impact on our proposed PF-GCN. We test the sensitivity of three parameters (p, λ, and l) on BlogCatalog, Citeseer, Flickr, and UAI2010 datasets. All experiments are conducted on the node classification task and then evaluated by ACC and F1. The experimental results are shown in Figs. 6, 7, and 8.

1) Parameter p: It refers to the proportion of pseudolabeled nodes that are added to the labeled dataset. The context size is tested from 10%, 30%, 50%, 70%, to 90% of all nodes

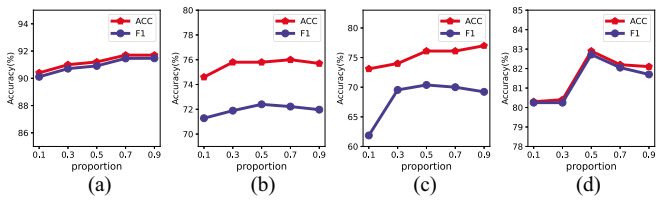

Fig. 6. Parameter sensitivity analysis of proportion p. (a) BlogCatalog. (b) Citeseer. (c) UAI2010. (d) Flickr.

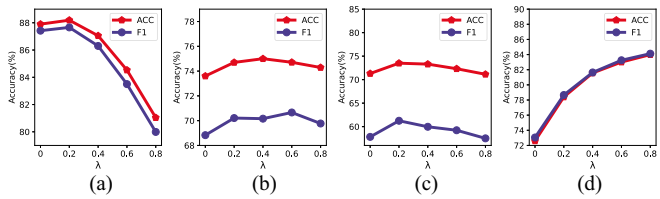

Fig. 7. Parameter sensitivity analysis of λ. (a) BlogCatalog. (b) Citeseer. (c) UAI2010. (d) Flickr.

(Fig. 6). On the BlogCatalog dataset, as the parameter p increases, the performance steadily improves and then stabilizes. On the Citeseer, UAI2010, and Flickr datasets, the classification performance first increases, followed by a subsequent decline. These results show that adding pseudolabels helps the PF-GCN capture a more comprehensive embedding by providing

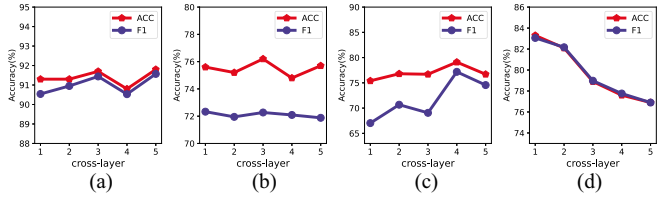

Fig. 8. Parameter sensitivity analysis of cross-layer l. (a) BlogCatalog. (b) Citeseer. (c) UAI2010. (d) Flickr.

an extra supervised signal. However, an excessive number of pseudolabels will introduce noise, which is detrimental to the performance.

2) Parameter λ: This hyper-parameter controls the balance between the real and the pseudolabels set. In determining the optimal value for λ, we conduct a comprehensive evaluation by varying λ across the range from 0 to 1 with intervals of 0.2. As depicted in Fig. 7, the accuracy initially rises, followed by a subsequent decline on BlogCatalog, Citeseer, and UAI2010 datasets. This phenomenon can be attributed to the increased inclusion of pseudolabels, introducing higher levels of noise that ultimately interfere with the performance. With the increase of the parameter λ, there is a consistent improvement in the classification performance observed on the Flickr dataset. Thus, careful consideration of the value of λ is crucial. The optimal values of λ are set at 0.2, 0.4, and 0.2 for the respective scenarios.

3) Parameter l: It refers to the number of cross-layers utilized to generate feature interaction. We conduct a comprehensive experiment to evaluate the influence by varying l from 1 to 5 with intervals of 1. The results are shown in Fig. 8. It is observed that the optimal values of l are set at 0.3, 0.3, 0.4, and 0.1 on BlogCatalog, Citeseer, UAI2010, and Flickr datasets. The accuracy initially rises, followed by a subsequent decline on the BlogCatalog, Citeseer, and UAI2010 datasets. With the increase of the parameter l, there is a consistent decrease in the classification performance observed on the Flickr dataset. Thus, it is important to carefully design the value of l. For density and high feature dimensions datasets like Flickr, cross-layer = 1 is sufficient to mine feature interaction. For sparse and lower feature dimensions datasets, such as Citeseer and UAI2010, multiple layers of feature interaction yield better performance.

VI. Conclusion and Future Works

In this article, we propose PF-GCN, a semisupervised attributed network representation learning method. The key focus of PF-GCN lies in its ability to predict pseudolabels for unlabeled nodes, effectively addressing the problem of sparse labels in real-world scenarios. Moreover, PF-GCN enables the propagation of feature interaction across both the topology and feature spaces, enhancing the learning for node embeddings. Experimental results demonstrate the superiority of the proposed PF-GCN against several competitors. Experimental results demonstrate the importance of incorporating pseudolabels and feature interaction into models. One further exploration of this study is to explore our proposed model to have a deeper structure and higher-order features. In the future, we plan to extend the pseudolabels module to heterogeneous and dynamic graph representation learning. Moreover, we intend to explore different methods of generating higher-order feature interaction.

References

[1] M. Li et al., "Guest editorial: Deep neural networks for graphs: Theory, models, algorithms, and applications," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 35, no. 4, pp. 4367–4372, Apr. 2024.

[2] S. Abu-El-Haija et al., "MixHop: Higher-order graph convolutional architectures via sparsified neighborhood mixing," in *Proc. Int. Conf. Mach. Learn.*, PMLR, 2019, pp. 21–29.

[3] J. Cao, X. Lin, S. Guo, L. Liu, T. Liu, and B. Wang, "Bipartite graph embedding via mutual information maximization," in *Proc. 14th ACM Int. Conf. Web Search Data Mining*, 2021, pp. 635–643.

[4] M. Li, X. Zhuang, L. Bai, and W. Ding, "Multimodal graph learning based on 3d haar semi-tight framelet for student engagement prediction," *Inf. Fusion*, vol. 105, 2024, Art. no. 102224.

[5] K. Wang, J. An, M. Zhou, Z. Shi, X. Shi, and Q. Kang, "Minority-weighted graph neural network for imbalanced node classification in social networks of internet of people," *IEEE Internet Things J.*, vol. 10, no. 1, pp. 330–340, Jan. 2023.

[6] L. Zou et al., "Neural interactive collaborative filtering," in *Proc. 43rd Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2020, pp. 749–758.

[7] M. Welling and T. N. Kipf, "Semi-supervised classification with graph convolutional networks," in *Proc. 5th Int. Conf. Learn. Representations (ICLR)*, 2017, pp. 1–14.

[8] S. Yun, M. Jeong, R. Kim, J. Kang, and H. J. Kim, "Graph transformer networks," in *Proc. Neural Inf. Process. Syst.*, vol. 32, Dec. 2019, pp. 11960–11970.

[9] X. Wang, N. Liu, H. Han, and C. Shi, "Self-supervised heterogeneous graph neural network with co-contrastive learning," in *Proc. 27th ACM SIGKDD Conf. Knowl. Discovery*, New York, NY, USA: ACM, 2021, pp. 1726–1736.

[10] S. Fan et al., "Metapath-guided heterogeneous graph neural network for intent recommendation," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discovery*, 2019, pp. 2478–2486.

[11] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," in *Proc. 6th Int. Conf. Learn. Representations*, 2018, pp. 1–12.

[12] J. You, R. Ying, and J. Leskovec, "Position-aware graph neural networks," in *Proc. Int. Conf. Mach. Learn.*, PMLR, 2019, pp. 7134–7143.

[13] X. Wang, M. Zhu, D. Bo, P. Cui, C. Shi, and J. Pei, "AM-GCN: Adaptive multi-channel graph convolutional networks," in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discovery & Data Mining*, 2020, pp. 1243–1253.

[14] C. Huang, M. Li, F. Cao, H. Fujita, Z. Li, and X. Wu, "Are graph convolutional networks with random weights feasible?" *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 3, pp. 2751–2768, Mar. 2023.

[15] X. Liu, M. Yan, L. Deng, G. Li, X. Ye, and D. Fan, "Sampling methods for efficient training of graph convolutional networks: A survey," *IEEE/CAA J. Automatica Sinica*, vol. 9, no. 2, pp. 205–234, Feb. 2022.

[16] M. Gong, H. Zhou, A. K. Qin, W. Liu, and Z. Zhao, "Self-paced co-training of graph neural networks for semi-supervised node classification," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 34, no. 11, pp. 9234–9247, Nov. 2023.

[17] J. Chen, G. Huang, H. Zheng, D. Zhang, and X. Lin, "Graphfool: Targeted label adversarial attack on graph embedding," *IEEE Trans. Comput. Social Syst.*, vol. 10, no. 5, pp. 2523–2535, Oct. 2023.

[18] Q. Zhang, Z. Zhao, H. Zhou, X. Li, and C. Li, "Self-supervised contrastive learning on heterogeneous graphs with mutual constraints of structure and feature," *Inf. Sci.*, vol. 640, pp. 119026–119040, Sep. 2023.

[19] X. Zhu and A. B. Goldberg, "Introduction to semi-supervised learning," *Synthesis Lectures on Artificial Intelligence and Machine Learning*. Morgan & Claypool Publishers, 2009. [Online]. Available: https://doi.org/10.2200/S00196ED1V01Y200906AIM006

[20] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, "Spectral networks and locally connected networks on graphs," 2013, *arXiv:1312.6203*.
[21] P. Zhu, J. Li, B. Cao, and Q. Hu, "Multi-task credible pseudo-label learning for semi-supervised crowd counting," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 35, no. 8, pp. 10394–10406, Aug. 2024.
[22] J. Kim et al., "Semi-supervised learning of semantic correspondence with pseudo-labels," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2022, pp. 19667–19677.
[23] R. Wang, L. Qi, Y. Shi, and Y. Gao, "Better pseudo-label: Joint domain-aware label and dual-classifier for semi-supervised domain generalization," *Pattern Recognit.*, 2023, Art. no. 108987.
[24] X. Jin et al., "Pseudo-labeling and meta reweighting learning for image aesthetic quality assessment," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 12, pp. 25226–25235, Dec. 2022.
[25] M. Seydgar, S. Rahnamayan, P. Ghamisi, and A. A. Bidgoli, "Semisupervised hyperspectral image classification using a probabilistic pseudo-label generation framework," *IEEE Trans. Geosci. Remote Sens.*, vol. 60, pp. 1–18, 2022.
[26] Z. Lu, R. Lin, Q. He, and H. Hu, "Mask-aware pseudo label denoising for unsupervised vehicle re-identification," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 4, pp. 4333–4347, Apr. 2023.
[27] M. Xu, H. Guo, Y. Jia, Z. Dai, and J. Wang, "Pseudo label rectification with joint camera shift adaptation and outlier progressive recycling for unsupervised person re-identification," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 3, pp. 3395–3406, Mar. 2023.
[28] G.-H. Wang and J. Wu, "Repetitive reprediction deep decipher for semi-supervised learning," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 6170–6177.
[29] Q. Li, Z. Han, and X.-M. Wu, "Deeper insights into graph convolutional networks for semi-supervised learning," in *Proc. AAAI Conf. Artif. Intell.*, 2018, pp. 3538–3545.
[30] A. Iscen, G. Tolias, Y. Avrithis, and O. Chum, "Label propagation for deep semi-supervised learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 5070–5079.
[31] Y. Li, J. Yin, and L. Chen, "Informative pseudo-labeling for graph neural networks with few labels," *Data Mining Knowl. Discovery*, vol. 37, no. 1, pp. 228–254, 2023.
[32] Y. Yang, Y. Sun, F. Ju, S. Wang, J. Gao, and B. Yin, "Multi-graph fusion graph convolutional networks with pseudo-label supervision," *Neural Netw.*, vol. 158, pp. 305–317, Jan. 2023.
[33] F. Feng, X. He, H. Zhang, and T.-S. Chua, "Cross-GCN: Enhancing graph convolutional network with k-order feature interactions," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 1, pp. 225–236, Jan. 2023.
[34] Z. Zhao, Z. Yang, C. Li, Q. Zeng, W. Guan, and M. Zhou, "Dual feature interaction-based graph convolutional network," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 9, pp. 9019–9030, Sep. 2023.
[35] D. Jin, C. Huo, C. Liang, and L. Yang, "Heterogeneous graph neural network via attribute completion," in *Proc. Web Conf.*, 2021, pp. 391–400.
[36] S. Li, G. Zhong, Y. Jin, X. Wu, P. Zhu, and Z. Wang, "A deceptive reviews detection method based on multidimensional feature construction and ensemble feature selection," *IEEE Trans. Comput. Social Syst.*, vol. 10, no. 1, pp. 153–165, Feb. 2023.
[37] Y. Tian, G. Liu, J. Wang, and M. Zhou, "ASA-GNN: Adaptive sampling and aggregation-based graph neural network for transaction fraud detection," *IEEE Trans. Comput. Social Syst.*, vol. 11, no. 3, pp. 3536–3549, Jun. 2024.
[38] E. Arazo, D. Ortego, P. Albert, N. E. O'Connor, and K. McGuinness, "Pseudo-labeling and confirmation bias in deep semi-supervised learning," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Piscataway, NJ, USA: IEEE Press, 2020, pp. 1–8.
[39] G. Yue, R. Xiao, Z. Zhao, and C. Li, "AF-GCN: Attribute-fusing graph convolution network for recommendation," *IEEE Trans. Big Data*, vol. 9, no. 2, pp. 597–607, Apr. 2023.
[40] Z. Zhao, Z. Yang, C. Li, Q. Zeng, W. Guan, and M. Zhou, "Dual feature interaction-based graph convolutional network," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 9, pp. 9019–9030, Sep. 2023.
[41] J. Lian, X. Zhou, F. Zhang, Z. Chen, X. Xie, and G. Sun, "XDeepFM: Combining explicit and implicit feature interactions for recommender systems," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2018, pp. 1754–1763.
[42] X. Liu, Z. Zhao, Y. Zhang, C. Liu, and F. Yang, "Social network rumor detection method combining dual-attention mechanism with graph convolutional network," *IEEE Trans. Comput. Social Syst.*, vol. 10, no. 5, pp. 2350–2361, Oct. 2023.
[43] K. Huang, Y. G. Wang, M. Li, and P. Lio, "How universal polynomial bases enhance spectral graph neural networks: Heterophily, over-smoothing, and over-squashing," in *Proc. 41st Int. Conf. Mach. Learn.*, vol. 235, 2024, pp. 20310–20330.
[44] L. Wang, X. Liu, X. Ma, J. Wu, J. Cheng, and M. Zhou, "A progressive quadric graph convolutional network for 3d human mesh recovery," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 33, no. 1, pp. 104–117, Jan. 2023.
[45] B. Perozzi, R. Al-Rfou, and S. Skiena, "DeepWalk: Online learning of social representations," in *Proc. 20th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2014, pp. 701–710.
[46] C. Liu, L. Wen, Z. Kang, G. Luo, and L. Tian, "Self-supervised consensus representation learning for attributed graph," in *Proc. 29th ACM Int. Conf. Multimedia*, 2021, pp. 2654–2662.
[47] H. Gao, J. Xiao, Y. Yin, T. Liu, and J. Shi, "A mutually supervised graph attention network for few-shot segmentation: The perspective of fully utilizing limited samples," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 35, no. 4, pp. 4826–4838, Apr. 2024.
[48] W. Ding, T. Zhang, H. Gao, Q. Yu, J. Wang, and Z. Zhao, "Multi-graph spatio-temporal convolution for traffic flow prediction focusing on edge derived imbalanced data from highway electronics," *IEEE Trans. Consum. Electron.*, early access, Jun. 18, 2024.
[49] Z. Xiao, P. Li, C. Liu, H. Gao, and X. Wang, "MACNS: A generic graph neural network integrated deep reinforcement learning based multi-agent collaborative navigation system for dynamic trajectory planning," *Inf. Fusion*, vol. 105, 2024, Art. no. 102250.
[50] S. Rendle, "Factorization machines," in *Proc. IEEE Int. Conf. Data Mining*, 2010, pp. 995–1000.
[51] H. Guo, R. Tang, Y. Ye, Z. Li, and X. He, "DeepFM: A factorization-machine based neural network for ctr prediction," 2017, *arXiv:1703.04247*.
[52] R. Wang, B. Fu, G. Fu, and M. Wang, "Deep & cross network for ad click predictions," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (ADKDD)*, 2017, pp. 1–7.
[53] R. Wang, R. Shivanna, D. Cheng, S. Jain, D. Lin, L. Hong, and E. Chi, "DCN V2: Improved deep & cross network and practical lessons for web-scale learning to rank systems," in *Proc. Web Conf.*, 2021, pp. 1785–1797.
[54] D.-H. Lee, "Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks," in *Proc. Workshop Challenges Representation Learn. (ICML)*, 2013, p. 896.
[55] W. Shi, Y. Gong, C. Ding, Z. Ma, X. Tao, and N. Zheng, "Transductive semi-supervised deep learning using min-max features," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 311–327.
[56] W. Wang, X. Liu, P. Jiao, X. Chen, and D. Jin, "A unified weakly supervised framework for community detection and semantic matching," in *Proc. Adv. Knowl. Discovery Data Mining: 22nd Pacific-Asia Conf.*, New York, NY, USA: Springer-Verlag, 2018, pp. 218–230.
[57] Y. Zhu, Y. Xu, F. Yu, Q. Liu, S. Wu, and L. Wang, "Graph contrastive learning with adaptive augmentation," in *Proc. Web Conf.*, 2021, pp. 2069–2080.
[58] R. Fang, L. Wen, Z. Kang, and J. Liu, "Structure-preserving graph representation learning," in *Proc. IEEE Int. Conf. Data Mining (ICDM)*, 2022, pp. 927–932.
[59] S. Gao, M. Zhou, Y. Wang, J. Cheng, H. Yachi, and J. Wang, "Dendritic neuron model with effective learning algorithms for classification, approximation, and prediction," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 2, pp. 601–614, Feb. 2019.
[60] X. Luo, Y. Yuan, S. Chen, N. Zeng, and Z. Wang, "Position-transitional particle swarm optimization-incorporated latent factor analysis," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 8, pp. 3958–3970, Aug. 2022.
[61] P. Zhang, S. Shu, and M. Zhou, "An online fault detection model and strategies based on SVM-grid in clouds," *IEEE/CAA J. Automatica Sinica*, vol. 5, no. 2, pp. 445–456, Mar. 2018.
[62] L. van der Maaten and G. Hinton, "Visualizing data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, no. 11, pp. 2579–2605, 2008.

Qiqi Zhang received the M.S. degree in computer science in 2024 from Shandong University of Science and Technology (SDUST), Qingdao, China, where she is currently working toward the Ph.D. degree with the College of Computer Science and Engineering.

Her research interests include graph neural networks and graph mining.

Zhongying Zhao (Member, IEEE) received the Ph.D. degree in computer science from the Institute of Computing Technology, Chinese Academy of Sciences, Shenzhen, China, in 2012.

Currently, she is a Professor with the College of Computer Science and Engineering, Shandong University of Science and Technology, Qingdao, China. From 2012 to 2014, she was an Assistant Professor with Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences. Her research interests include graph learning, graph neural networks, and data mining. She has published more than 80 papers in international journals and conferences such as IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, IEEE TRANSACTIONS ON BIG DATA, IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, IEEE TRANSACTIONS ON NETWORK SCIENCE AND ENGINEERING, *ACM TOMM*, SIGKDD, and AAAI.

Dr. Zhao is a Distinguished Member of CCF. She was awarded as Taishan Scholar of Shandong Province, and she is the Leader of Shandong Youth Innovative Team.

Chao Li received the Ph.D. degree in computer science from Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, Shenzhen, China, in 2014.

From 2014 to 2015, he was a Visiting Scholar with the Department of Computer Science and Engineering, The Hong Kong University of Science and Technology, Hong Kong. Currently, he is a Professor with the College of Computer Science and Engineering, Shandong University of Science and Technology, Qingdao, China. He has published more than 40 papers in international journals and conferences. His research interest includes big data analysis, social network analysis and graph mining.

Xin Huang received the Ph.D. degree from the Chinese University of Hong Kong, Hong Kong, in 2014.

Currently, he is an Associate Professor with Hong Kong Baptist University, Hong Kong. His research interests include graph data management and mining. He has published in top conferences and journals including SIGMOD, PVLDB, WWW, ICDE, CIKM, AAAI, IJCAI, IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, *TKDD*, and *VLDBJ*.