2025 5th International Conference on Artificial Intelligence and Industrial Technology Applications (AIITA)

Contrastive Learning Enhanced Semi-Supervised Graph Neural Networks

Dongdong Du 1st
China Academy of Industrial Internet
Beijing, 100015, China

Songzi Cao 2nd
China Academy of Industrial Internet
Beijing, 100015, China

Xu Ouyang 3rd
China Academy of Industrial Internet
Beijing, 100015, China

Jie Shi [*]
China Academy of Industrial Internet
Beijing, 100015, China
[*]Corresponding author: shijie@china-aii.com

Abstract—**Graphs are an effective data organization method capable of representing complex relationships and structures in the real world. Graph Neural Networks (GNN) extract feature representations of graphs by learning the relationships between nodes, enabling the modeling and analysis of graph data. GNN methods heavily rely on a substantial amount of labeled nodes to enhance model performance. Pseudo-labeling techniques can effectively leverage unlabeled nodes to mitigate this issue. However, existing pseudo-labeling methods only utilize unlabeled nodes with confidence scores above a certain threshold, while those below the threshold are not directly used for model training, resulting in lower data utilization. The method proposed in this paper adopts a contrastive learning strategy for nodes with confidence lower than the threshold, enhancing the node representation learning performance. For nodes with confidence higher than the threshold, a pseudo-labeling method is adopted to reduce the dependence of data on labeled samples. Finally, this paper conducted comparative experiments on three datasets. The experimental results show that compared with the baseline method, this method achieved the best experimental performance on all datasets.**

Keywords-Graph Neural Networks; Semi-Supervised Learning; Contrastive Learning; Pseudo-label Learning

I. INTRODUCTION

Graph is a powerful data organization way, which can represent many complex relationships in the real world and is widely used in various fields. In recent years, Graph Neural Networks (GNNs) [1] have shown excellent performance in processing graph data. GNNs can incorporate rich node attributes along with graph structural information [2][3][4][5]. Despite the tremendous achievements of these GNNs, they are still limited to the conventionally semi-supervised framework. If the labeled data are insufficient, the model performance is difficult to be improved effectively. Therefore, it is urgent to fully leverage quantities of unlabeled data to further improve the performance of graph-based semi-supervised learning.

Recently, there have been some semi-supervised learning methods that employ pseudo-labeling to fully leverage unlabeled data [6][7]. Pseudo-labeling is a simple but effective technology. By using a model trained with the labeled data, it generates pseudo-labels of unlabeled data and iteratively retrains the model with both given labels and pseudo-labels [8]. In the field of graph data, only a few models apply pseudo-labeling methods to graph neural networks, and these models merely utilize pseudo-labeling techniques in a rudimentary manner, overlooking the issues that arise when integrating pseudo-labeling with graph neural network models. These pseudo-labeling methods only use the unlabeled data with high prediction confidence to calculate the unsupervised loss. This will result in the information of low-confidence unlabeled nodes often being unable to be used for model training. But the nodes in the graph do not exist in isolation, and discarding low-confidence unlabeled nodes means that these data information cannot be used, which is not conducive to improving model performance.

Based on this issue, the method proposed in this paper employs a contrastive learning strategy for nodes with confidence levels below a certain threshold to enhance the performance of node representation learning. For nodes with confidence levels above the threshold, the method utilizes pseudo-labeling to reduce the reliance on labeled samples. In the experimental section, this paper conducted classification performance comparison experiments across three datasets, as well as a visualization experiment on the Cora dataset. The results show that, compared to baseline methods, the proposed method achieved the best performance across all datasets, with an average improvement of approximately 1.3% in node classification accuracy. This indicates that the proposed method significantly enhances data utilization, thereby boosting model performance.

II. THE PROPOSED MODEL

The method proposed in this paper aims to fully utilize unlabeled data for facilitating semi-supervised node classification. We design an adaptive confidence discrimination module to divide all unlabeled nodes into two subsets according to their confidence scores. Moreover, we present different constraint strategies for two subset nodes. Reliable pseudo-labels are used to iteratively expand the label set, and discriminative features are better learned by applying contrastive learning.

A. *Graph Augmentation Encoder*

This model performs data augmentation on the input graph to generate enhanced views of the original graph, which serves as the foundation for subsequent pseudo-label learning. Given an undirected, unweighted graph $\mathrm{G} = (\mathrm{V}, \varepsilon)$, where V is the set of nodes $\{\upsilon_1, \upsilon_2, \dots\}$ and ε is the set of edges, the generalized graph diffusion defined by the diffusion matrix is as follows:

$$\mathrm{S} = \sum_{\mathrm{k}=0}^{\infty} \theta_{\mathrm{k}} \mathrm{T}^{\mathrm{k}}, \tag{1}$$

let $\sum_{k=0}^{\infty} \theta_k = 1$, where $\theta_k \in [0,1]$ represents the weight coefficients. The ratio of the information between the original view and the enhanced view can be adjusted by varying θ_k. $\mathrm{T} \in \mathbb{R}^{n\times n}$ denotes a generalized transition matrix, whose eigenvalues λ must satisfy $0 \le \lambda \le 1$. For a given adjacency matrix A, an identity matrix I, and a diagonal matrix D, the generalized graph diffusion is typically defined as $\mathrm{T}_k = AD^{-1}$ and $\theta_k = \alpha(1-\alpha)^k$,where $\alpha \in [0,1]$ represents the transmission probability. Equation 2 can be reformulated as:

$$\hat{S} = \alpha\left(I_n - (1-\alpha)D^{-\frac{1}{2}}AD^{-\frac{1}{2}}\right)^{-1}, \tag{2}$$

here, $\hat{S}$ denotes the augmented matrix obtained by applying graph data augmentation to the adjacency matrix A. In this study, both the augmented view corresponding to the augmented matrix and the original view corresponding to the adjacency matrix are fed into the graph data encoder to further extract features of the nodes from both views.

We regard adjacency matrix A and diffusion matrix $\hat{S}$ as two consistent views V_α and V_β, respectively. To learn the node representation of two views, GCN [2] is used as an encoder for parameter sharing. The propagation rules from $l-th$ layer to $(l+1)-th$ layer are given as follows:

$$H^{(l+1)} = \sigma\left(\hat{A}H^{(l)}W^{(l)}\right), \tag{3}$$

referring to Equation 3, the respective GCN layer transfer formulas of the two views can be expressed as follows:

$$H_\alpha^{(l+1)} = \sigma\left(\hat{A}H^{(l)}W^{(l)}\right), \tag{4}$$

$$H_\beta^{(l+1)} = \sigma\left(\hat{S}H^{(l)}W^{(l)}\right). \tag{5}$$

The final node feature representations $H_\alpha = (\overrightarrow{h_{\alpha 1}}, \overrightarrow{h_{\alpha 2}}, \dots \overrightarrow{h_{\alpha n}})^T \in \mathbb{R}^{n\times f}$ and $H_\beta = (\overrightarrow{h_{\beta 1}}, \overrightarrow{h_{\beta 2}}, \dots \overrightarrow{h_{\beta n}})^T \in \mathbb{R}^{n\times f}$ of the original view V_α and enhanced view V_β can be obtained by the graph data encoder. Where n is the number of nodes in the dataset, f is the dimension of node features, and $\overrightarrow{h_{\alpha 1}}$ and $\overrightarrow{h_{\beta 1}}$ are the encoded embedding vectors of the first node in the two views respectively. After the node is embedded through the fully connected layer, the $logits$: $g = \{g_1, g_2, \dots g_C\}$ belonging to each class is obtained, where C is the number of classes of nodes in the dataset. The $logits$ goes through the $softmax$ function of the classifier to get the probability distribution $p = \{p_1, p_2, \dots p_C\}$ of the nodes belonging to each class.

B. *Pseudo-label learning*

In the training process, we employ the cross-entropy loss of labeled nodes to train the encoder. The resulting supervised loss is:

$$L_{sup} = -\frac{1}{N_s}\sum_{i=1}^{N_s}\sum_{c=1}^{C} y_{i,c}\, log\left(p_{i,c}\right), \tag{6}$$

where C denotes the number of node classes, and $y_{i,c} \in \{0,1\}$ is an indicator function representing whether the label of node v_i is class c. If the label of node v_i is class c, then $y_{i,c} = 1$; otherwise, $y_{i,c} = 0$. The term $p_{i,c}$ represents the predicted probability that node v_i belongs to class c.

The threshold is a crucial criterion for determining whether an unlabeled node can be assigned a pseudo label. If the confidence score of an unlabeled node exceeds the threshold, the node is considered to have high reliability and can be used as a pseudo label for model training. Conversely, if the confidence score of an unlabeled node is below the threshold, the node is deemed to have low reliability and cannot be used as a pseudo label for model training. In the proposed method, we utilized a predefined threshold τ to partition the set of unlabeled nodes into two distinct subsets:

$$\begin{cases} v_k \in set\ \mathrm{I}\ ,\ if\ s_k \ge \tau \\ v_k \in set\ \mathrm{II}\ ,\ if\ s_k < \tau \end{cases}, \tag{7}$$

nodes in set I have higher confidence scores, making their predictions more reliable. Consequently, the pseudo-labeling method assigns pseudo-labels $\hat{y}_k$ based on these predictions. These predictions are represented as unit vectors and used as labels in the current training epoch, effectively expanding the label set. For nodes in set I, the unsupervised loss L_{plu} is defined as:

$$L_{plu} = -\frac{1}{N_h}\sum_{k=1}^{N_h}\sum_{c=1}^{C} \hat{y}_{k,c}\, log\left(p_{k,c}^{\beta}\right), \tag{8}$$

where C denotes the number of node categories, N_h represents the number of nodes in set I, and $\hat{y}_{k,c} \in \{0,1\}$ is an indicator function indicating whether the pseudo-label of node v_k is of category c, then $\hat{y}_{k,c} = 1$; otherwise, $\hat{y}_{k,c} = 0$. $p_k^\beta = \{p_{k,1}^\beta, \dots, p_{k,c}^\beta\}$ represents the probability distribution of node v_k after data augmentation for the augmented view V_β, where $p_{k,c}^\beta$ denotes the predicted probability of node v_k belonging to category c after data augmentation.

C. *Contrastive Learning*

For nodes in set II, which have lower confidence scores, their predictions are highly likely to be erroneous. Incorporating such erroneous predictions into model training may degrade model performance. Consequently, this section does not utilize these predictions; instead, it focuses on learning their feature representations. Specifically, this section employs contrastive learning to further learn the feature representations from the original view V_α and the augmented view V_β for both same-node and different-node instances, thereby enhancing the model's ability to discriminate between features of different nodes.

The concept of contrastive learning treats feature representations from the same node as positive sample pairs and aims to bring them closer together, while feature representations from different nodes are treated as negative sample pairs and pushed further apart [9]. In this paper, the method aims to bring closer together feature representations from the same node but different views, and to push apart feature representations from different nodes and different views. The cosine similarity between the feature representations of an unlabeled node v in the original view V_α and the augmented view V_β is expressed as follows:

$$sim\left(e_k^\alpha, e_k^\beta\right) = \frac{\left(e_k^\alpha\right)\left(e_k^\beta\right)^T}{\|e_k^\alpha\|\|e_k^\beta\|}, \tag{9}$$

here, the corresponding node representations in the original view V_α and the augmented view V_β are denoted as $\mathrm{e}_\mathrm{k}^\alpha$ and $\mathrm{e}_\mathrm{k}^\beta$, respectively. The unsupervised loss for nodes in the set II is computed based on the cosine similarity given by the above expression:

$$L_{con} = -\frac{1}{N_l}\sum_{k=1}^{N_l} log\left(\frac{e^{sim\left(e_k^\alpha, e_k^\beta\right)\cdot t^{-1}}}{e^{sim\left(e_k^\alpha, e_k^\beta\right)\cdot t^{-1}} + \sum_{m\neq k} e^{sim\left(e_k^\alpha, e_m^\beta\right)\cdot t^{-1}}}\right), \tag{10}$$

where $\mathrm{N_l}$ denotes the number of nodes in the set II, with $\mathrm{k, m} = \{1,2, \ldots \mathrm{N_l}\}$. The temperature coefficient $\mathrm{t} \in (0,1)$ is used to scale the size of similarity pairs. This method not only effectively leverages low-confidence unlabeled nodes but also enhances the model's feature recognition capability without introducing additional parameters.

By using a pre-defined threshold, unlabeled nodes are divided into two subsets, set I and set II, which are subjected to different constraints. This approach aims to mitigate pseudo-label bias and effectively utilize low-confidence unlabeled nodes. The overall objective function of the model is given by:

$$L = L_{sup} + \lambda_1 L_{plu} + \lambda_2 L_{con}, \tag{11}$$

where L_{sup} denotes the loss for labeled nodes, L_{plu} represents the debiasing loss for unlabeled nodes in the set I, and L_{con} denotes the contrastive loss for unlabeled nodes in the set II. The parameters λ_1 and λ_2 are used for balancing.

III. Experimental results and analysis

In this part, the proposed method is compared with mainstream methods across various datasets, and a detailed analysis of the experimental results is provided.

A. *Datasets*

To conduct a comprehensive evaluation, this chapter performs experiments using three commonly utilized graph datasets.

Cora and Pubmed [10]: Classic citation network datasets. In these datasets, nodes represent machine learning papers and biomedical papers, respectively. Node attributes are represented by keywords, and edges are constructed based on citation relationships between these nodes.

Amazon Computer [11]: A subset of the Amazon co-purchase graph. Nodes represent products included in the dataset, with node attributes encoded by a bag-of-words representation of product reviews. Edges are constructed based on the co-purchase relationships between products.

Each dataset is divided into training, validation, and test sets. For the experiments in this study, 20 labeled nodes are selected from each category for the training set, while 500 and 1,000 nodes are chosen for the validation and test sets, respectively. The partitioning strategy varies across different datasets. For the Cora and Pubmed datasets, this study follows the existing dataset partitioning strategies [10]. For the Amazon Computer dataset, the partitioning is based on labels: 20 nodes are randomly selected from each category for the training set, while the validation and test sets are randomly sampled from the remaining nodes.

B. *Parameter Settings and Evaluation Indicators*

The backbone network of the proposed method is a two-layer GCN. The hidden layer size is set to 16, and the Adam optimizer with an initial learning rate ranging from 0.01 to 0.1 is used, along with a weight decay of $5e^{-4}$ to prevent over-fitting. Dropout is set to 0.5. In the comparative experiments, this paper evaluated the proposed method against three advanced techniques using three datasets, with the experimental process divided into two phases. In the first phase, the graph neural network encoder is trained using labeled nodes to obtain a trained encoder. In the second phase, the trained encoder is used for pseudo-label prediction and feature learning. For the Cora and Pubmed datasets, the epoch value for the first phase is set to 150 and for the second phase to 500. For the Amazon Computer dataset, the epoch value is set to 1000 for the first phase and 2000 for the second phase.

After the first phase, for the Cora and PubMed datasets, this study compares node confidence scores with predefined thresholds every 50 epochs. For the Amazon Computer dataset, this comparison is made every 100 epochs. Following this comparison, unlabeled nodes are divided into set I and set II, with different constraint strategies applied to the unlabeled nodes in each set.

This study uses two common evaluation metrics—accuracy (ACC) and F1 score (F1)—to assess the performance of the proposed method and comparison methods in node classification tasks. Accuracy measures the proportion of correctly predicted nodes out of the total predicted nodes. The F1 score considers both precision and recall, representing their harmonic mean.

TABLE I. Node classification accuracy for different methods (%).

Datasets	Cora	Pubmed	Amazon Computer
GCN	81.5	79.0	73.0±0.6
GAT	83.0±0.7	79.0 ± 0.3	78.0 ± 0.4
TAGCN	83.3±0.7	81.1± 0.4	77.5 ± 0.5
GraFN	83.1±0.5	80.3 ± 0.3	78.2 ± 0.5
Ortho-GConv	83.3±0.4	79.5±0.5	78.0±1.5
Ours	**84.6±0.1**	**82.1 ± 0.2**	**79.7 ± 0.2**

C. *Comparison Methods*

The proposed method is compared with the following four state-of-the-art methods, with their characteristics described as follows.

GCN [2]: Adopts a layered propagation rule to encode graph structure and node features based on first-order neighbors, effectively capturing node relationships and improving representation learning of graph data.

GAT [12]: Introduces an attention mechanism to assign different weights to neighboring nodes. By focusing on the contribution of relevant nodes to the current node, it enhances the ability to learn node representations.

TAGCN [13]: Defines vertex domain convolution, unifying spectral and vertex domain graph convolutional networks. By combining graph convolution with adaptive mechanisms, it effectively learns node representations and is suitable for graph data with varying structures.

GraFN [14]: Utilizes consistent regularization to ensure that nodes belonging to the same class are grouped together, and then employs pseudo-labeling techniques to enhance model performance.

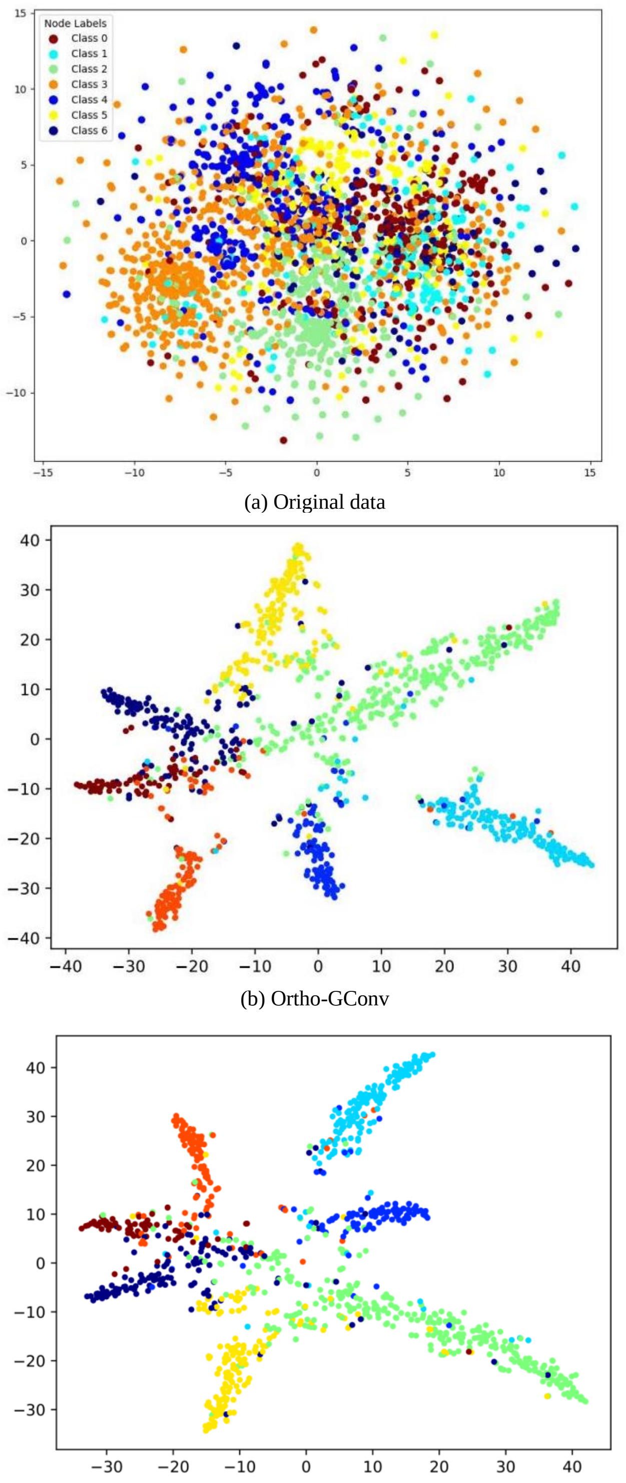
Ortho-GConv [15]: By applying orthogonal graph convolution to the backbone network of the model, this method minimizes correlations in the node feature space, thereby stabilizing model training and enhancing generalization performance.

D. *Results Analysis*

In Table 1, the proposed method achieves the best classification performance across three datasets, with an average improvement in model performance of approximately 1.3%. These results demonstrate the superiority of the proposed method when labeled nodes are scarce. Furthermore, the unsupervised feature learning module introduced in this paper effectively leverages all unlabeled nodes' information, aside from pseudo-labels, enhancing the utilization of unlabeled nodes. It learns features from both the original and augmented views of each node, thereby significantly improving the model's ability to discern node features.

To further demonstrate the superiority of the proposed method, this section presents node clustering and class similarity visualization experiments on 1,000 nodes from the test set of the Cora dataset. The comparative method used is the graph neural network approach Ortho-GConv.

The node clustering experiment utilizes the t-SNE [16] method to project node embeddings from the test set into a two-dimensional space. In Figure 1, circles represent nodes in the graph, with different colors indicating different classes. Figure 1 (a) illustrates the original distribution of nodes in the Cora dataset. The visualization of node clustering results reveals that, although the Ortho-GConv method can roughly project nodes of all classes into distinct clusters as shown in Figure 1 (b), it exhibits some mixing of nodes from different classes and has relatively blurred boundaries between classes, resulting in suboptimal visualization for the comparative method. In contrast, Figure 1 (c) shows that the proposed method more effectively partitions nodes of the same class into compact clusters with clearer boundaries between different class clusters. Consequently, the proposed method achieves the best performance in the node clustering experiment.



(a) Original data

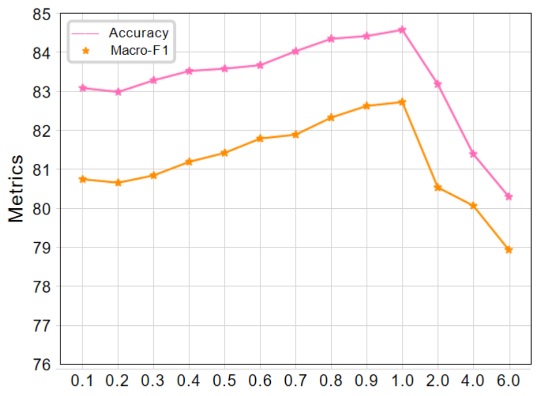
(b) Ortho-GConv

(c) Our method

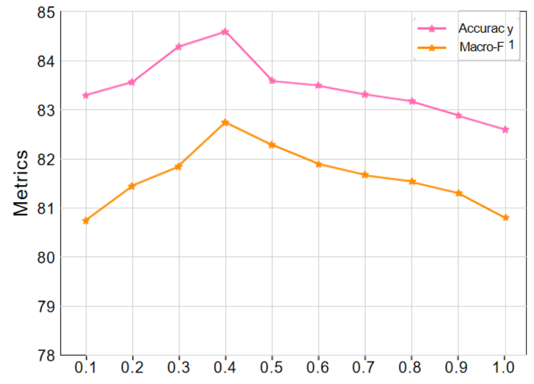
Figure 1. Node clustering visualization on the Cora dataset.

E. *Hyperparameter Optimization*

In order to study the impact of hyperparameters $\boldsymbol{\lambda_1}$ and $\boldsymbol{\lambda_2}$ on model performance, this section conducts hyperparameter sensitivity analysis experiments on the Cora dataset, with the evaluation indicators being accuracy and F1 score. As shown in Figure 2 (a), it can be found that when $\boldsymbol{\lambda_1 \leq 1}$, the performance of the model gradually improves as $\boldsymbol{\lambda_1}$ increases. From Figure 2 (b), we can see that as $\boldsymbol{\lambda_2}$ increases, the model performance first improves and then slowly decreases. To maximize the model performance, this paper sets $\boldsymbol{\lambda_1} \in [\mathbf{0.8}, \mathbf{1.0}]$ and $\boldsymbol{\lambda_2} \in [\mathbf{0.3}, \mathbf{0.5}]$.



(a) $\boldsymbol{\lambda_1}$ -Cora



(b) $\boldsymbol{\lambda_2}$ -Cora

Figure 2. Effects of hyperparameter λ_1 and λ_2 .

F. *Different Confidence Thresholds*

We experimented with different predefined thresholds to partition the unlabeled nodes on the Cora dataset. As shown in Table 2, when the threshold increased from the commonly used 0.80 to 0.98 (resulting in a larger number of unlabeled nodes participating in contrastive learning), the accuracy of node classification in the proposed model progressively improved. These results demonstrate the effective utilization of low-confidence unlabeled nodes by our model.

IV. Conclusion

This paper first introduces the extensive applications and research significance of graph data and then reviews the current state of research on graph neural networks for handling graph data. Existing graph neural network methods have only made rudimentary use of pseudo-labeling techniques, resulting in low node utilization. Therefore, the proposed method aims to fully exploit unlabeled nodes in semi-supervised learning to enhance graph data utilization. We apply contrastive learning to nodes below a certain threshold, effectively learning feature representations for low-confidence unlabeled nodes and improving the model's discriminative capability. For nodes with confidence above the threshold, we employ a pseudo-labeling approach to reduce reliance on labeled samples. Our experiments demonstrate that the proposed method effectively addresses issues associated with pseudo-labeling in graph neural networks, thereby improving model performance.

TABLE II. Effects of predefined threshold (%).

Threshold	0.98	0.95	0.90	0.80
ACC	**84.5**	84.0	81.8	78.5

References

[1] Kipf T N, Welling M. Semi-supervised classification with graph convolutional networks[C]. Proceedings of the International Conference on Learning Representations,2017.

[2] Lu W, Guan Z, Zhao W. Pseudo contrastive learning for graph-based semi-supervised learning[J]. arXiv preprint arXiv:2302.09532,2023.

[3] Liu Y, Wang X, Wu S and Xiao Z. Independence promoted graph disentangled networks[C]. Proceedings of the 34th AAAI Conference on Artificial Intelligence,2020,34(04):4916–23.

[4] Bianchi F. M, Grattarola D, Livi L and Alippi C. Graph neural networks with convolutional arma filters[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence,2021.

[5] J. Klicpera, A. Bojchevski, and S. Gunnemann, “Predict then propagate: Graph neural networks meet personalized pagerank [C]” in Proceedings of the 7th International Conference on Learning Representations (ICLR),2019.

[6] Li Y, Yin J and Chen L. Informative pseudo-labeling for graph neural networks with few labels[J]. Data Mining and Knowledge Discovery,2023,37(01):228–254.

[7] Liu Y, Yu L, Zhao S, et al. Graph Neural Networks With Adaptive Confidence Discrimination[J]. IEEE Transactions on Neural Networks and Learning Systems, 2024.

[8] Arazo E, Ortego D, Albert P, O’Connor N. E and McGuin-ness K. Pseudo-labeling and confirmation bias in deep semi-supervised learning[C]. Proceedings of the International Joint Conference on Neural Networks,2020.

[9] Liu Y, Zhao Y, Wang X, et al. Multi-scale subgraph contrastive learning[J]. arXiv preprint arXiv:2403.02719, 2024.

[10] Yang Z, Cohen W, and Salakhudinov R. Revisiting semi-supervised learning with graph embeddings[C]. Proceedings of the 33rd International Conference on Machine Learning,2016:40–48.

[11] Shchur O, Mumme M, Bojchevski A and G¨unnemann S. Pitfalls of graph neural network evaluation[J]. Neural Information Processing Systems Workshop,2018.

[12] Veličković P, Cucurull G, Casanova A, et al. Graph attention networks[C]. Proceedings of the International Conference on Learning Representations,2018.

[13] Du J, Zhang S, Wu G, et al. Topology adaptive graph convolutional networks[J]. arXiv preprint arXiv:1710.10370,2017.

[14] Lee J, Oh Y, In Y, et al. GraFN: Semi-supervised node classification on graph with few labels via non-parametric distribution assignment[C]. Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval,2022:2243-48.

[15] Guo K, Zhou K, Hu X, et al. Orthogonal graph neural networks[C]. Proceedings of the AAAI Conference on Artificial Intelligence,2022, 36(4): 3996-4004.

[16] Van Der Maaten L. Accelerating t-SNE using tree-based algorithms[J]. The Journal of Machine Learning Research, 2014, 15(1): 3221-3245.