

Article

MTKD-RL: Multi-Teacher Knowledge Distillation Method for Reinforcement Learning Based on Few-Shot Node Classification

Dianjun Xie ¹, Wenai Song ^{2,*}, Ruize Guo ¹, Biaokai Zhu ¹ and Yiran Li ¹

¹ Department of Network Security and Protection, Shanxi Police College, Taiyuan 030400, China; 250045@sxpc.nuc.edu.cn (D.X.); 250044@sxpc.edu.cn (R.G.); 250024@sxpc.edu.cn (B.Z.); 250048@sxpc.edu.cn (Y.L.)

² School of Software, North University of China, Taiyuan 030051, China

* Correspondence: songwenai@nuc.edu.cn

Abstract

Few-shot node classification aims to effectively predict new class nodes using only a small number of labeled samples, which is an important research direction in graph data mining. Existing self-training-based few-shot node classification methods are constrained by the bias and local optima of a single-teacher model. Meanwhile, multiple-teacher models often suffer from knowledge conflicts and redundant information, which degrade distillation efficiency and model generalization performance. To address these issues, we propose a multi-teacher knowledge distillation method with reinforcement learning for few-shot node classification (MTKD-RL). This framework is composed of a multi-teacher distillation network and a multi-teacher weight optimization module to deliver complementary supervision information from multiple perspectives. The reinforcement learning agent dynamically assigns adaptive weights to different teachers based on their prediction performance and the discrepancy between teacher and student models, which greatly enhances pseudo-label quality and distillation performance. Experiments on nine graph network datasets demonstrate that our method achieves consistent accuracy improvements ranging from 1.4% to 6.2% in few-shot node classification.

Keywords: few-shot node classification; knowledge distillation; reinforcement learning

1. Introduction

As the core task of graph data analysis [1], the node classification task is widely used in social network analysis, citation network analysis and other scenarios [2–4], and its goal is to predict the class label of unlabeled nodes using graph structure and node features. However, there are long-tail distribution and label sparsity problems in graph networks in real scenarios, which seriously restrict the generalization performance of traditional graph neural network models [5,6]. Therefore, few-shot node classification (FSNC) emerges as The Times require, which relies on a very small number of labeled nodes to train the model and realize the prediction of new categories of nodes [7,8], which has attracted the attention of academia and industry in recent years.

Existing few-shot node classification methods are mainly divided into meta-learning-based methods [9–11] and self-training-based methods [12]. By constructing multiple sets of meta-training tasks, meta-learning methods achieve rapid classification of new categories by means of model-independent meta-learning or metric learning. Huang et al. [13] proposed the G-META method, which uses local subgraphs as learning units to improve cross-task generalization ability while retaining topology and feature information. Ding et al. [14] designed



Academic Editor: Tehmina Amjad

Received: 16 May 2026

Revised: 22 June 2026

Accepted: 24 June 2026

Published: 28 June 2026

Copyright: © 2026 by the authors.

Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

graph prototype network GPN, which used metric learning to construct category prototypes to effectively depict node representation. VNT [15] uses Transformer as encoder, introduces virtual nodes as soft hints, and uses a small number of labels to quickly modulate the embedding to adapt to new tasks. COLA [16] mines information from graph-augmented views, builds semantic similarity meta-tasks unsupervised, and improves performance with contrastive loss. However, the meta-learning method is time-consuming to construct meta-tasks and does not make full use of the unlabeled information of the base class. Therefore, the self-training method is introduced into few-shot node classification, which improves the performance of the model with lower computational cost by expanding the pseudo-label supervision signal for unlabeled nodes. Wu et al. [12] proposed the IA-FSNC method to select high-quality pseudo-labels to expand the support set based on information entropy. Wu et al. [17] proposed the PLD-FSNC method to realize the transfer of features and classification ability from teachers to students with the help of knowledge distillation, so as to enhance the effectiveness of the model.

Although certain progress has been made through self-training methods, there are still significant limitations. Firstly, single-teacher knowledge distillation is prone to being affected by its own biases and local optima, making it difficult to comprehensively capture the semantic information and structural features of graph data, resulting in a single and insufficient supervisory signal provided to the student model. Secondly, if the multi-teacher model adopts a fixed-weight allocation strategy, it cannot adaptively balance the intensity of knowledge distillation among different teachers based on the training status, making it difficult to solve the problems of knowledge conflicts and redundancies among teachers, and preventing the student model from prioritizing the absorption of high-value knowledge, thereby limiting the distillation efficiency and generalization ability. These issues collectively constrain the performance breakthrough of few-shot node classification.

To solve the above problems, this paper proposes a multi-teacher knowledge distillation method for reinforcement learning based on few-shot node classification (MTKD-RL). Firstly, a multi-teacher distillation network composed of a representation teacher, a structure teacher and a semantic teacher was constructed to provide complementary supervision signals for the student model from three dimensions of local-global features, positional encoding structure and large-model-enhanced semantics respectively. Then, the reinforcement learning agent was used to dynamically generate adaptive weights according to the real-time performance of teachers and the distribution difference between teachers and students, so as to realize the dynamic allocation and efficient transfer of multi-teacher knowledge. On this basis, the pseudo-label generation and distillation process are optimized to improve the quality of pseudo-labels and the efficiency of knowledge transfer. Finally, multiple groups of few-shot experiments are carried out on nine mainstream graph datasets. The results show that the proposed method achieves significantly better classification performance than the existing methods under various settings.

The core innovations are as follows:

- (1) The proposed MTKD-RL method transfers the comprehensive knowledge of the pre-trained teacher (representational teacher, structural teacher, and semantic teacher) to the student model for learning. These three sets of complementary supervision signals provide the student model with rich and multi-perspective guidance, thereby improving the quality of pseudo-label selection.
- (2) The weight optimization module of reinforcement learning adaptively adjusts the knowledge contribution of each teacher based on the real-time training status, effectively alleviating the knowledge conflicts and information redundancy among multiple teachers, and enhancing the distillation efficiency and model generalization ability.
- (3) Through comprehensive experimental evaluations on nine benchmark graph networks datasets, the effectiveness of the proposed method has been verified.

2. Problem Definition

The attribute graph is represented as $G = \{A, X\}$, where A is the adjacency matrix, $X \in R^{|v| \times d}$ is the node feature matrix, and where d is the feature dimension. The entire node class is divided into disjoint base classes C_{base} and new classes C_{novel} . Specifically, $C = C_{base} \cup C_{novel}$ and $C_{base} \cap C_{novel} = \emptyset$ [18].

Few-shot node classification task based on self-training: The model first trains from the base dataset C_{base} to acquire prior knowledge, initializes parameters for a single-task learning in the base dataset, pre-trains the first-order embeddings $Z = g(A, X)$ and probability matrix of all nodes $P = \text{softmax}(Z)$ in the base dataset, and fully utilizes the information of the base dataset [19]. During testing, a classification model is constructed on the support set $S = \{(x_i, y_i)\}_{i=1}^n$ of the new class dataset C_{novel} , and FSNC evaluates the performance of the trained model on the query set $Q = \{(x_i, y_i)\}_{i=N \times K+1}^{N \times b}$ of the new class dataset.

3. Multi-Teacher Knowledge Distillation Method for Reinforcement Learning Based on Few-Shot Node Classification

This section specifically elaborates the proposed multi-teacher knowledge distillation method for reinforcement learning based on few-shot node classification (MTKD-RL); the method mainly consists of two parts: multi-teacher distillation network and multi-teacher weight optimization module. The multi-teacher network consists of a representation teacher, a structure teacher and a semantic teacher. The representation teacher distills the global and local feature information of the convolutional layer of the graph, the structure teacher distills the global position to encode the structural information, and the semantic teacher distills the enhanced semantic feature information of the graph nodes to provide comprehensive supervision and guidance information. The multi-teacher weight optimization module supervised the pseudo-label soft prediction process of the student model through the soft labels predicted by the multi-teacher to improve the quality of the pseudo-label. The weights between teachers were dynamically generated through the reinforcement learning agent, and the student network preferentially learned the current more valuable teacher knowledge, which significantly improved the distillation effect. The model structure diagram is shown in Figure 1.

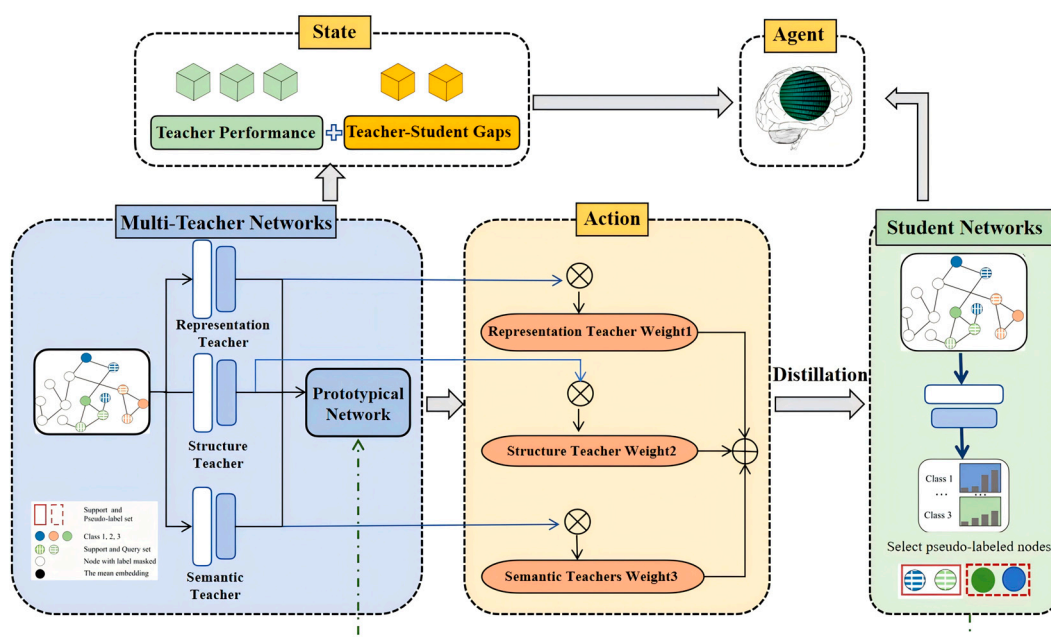


Figure 1. Overall framework of multi-teacher knowledge distillation method for reinforcement learning based on few-shot node classification (MTKD-RL).

3.1. Multi-Teacher Distillation Network

The multi-teacher network is composed of a representation teacher, a structure teacher and a semantic teacher. We adopt the three teachers because graph data inherently contains feature, topology and semantic information that cannot be fully captured by a single model. The single-teacher network is easily limited by its own bias and local optimum, and it is hard to perceive the multi-faceted attributes of graph nodes under the few-shot setting with scarce labeled data. The representation teacher, structure teacher and semantic teacher provide complementary supervision from different dimensions such as feature representation, network structure coding and semantic information respectively, so that the student network can obtain more comprehensive and richer learning objectives and reduce the risk of overfitting. The mutually supplementary knowledge from three teachers also alleviates the problem of insufficient supervision signals caused by limited labeled samples. This enables the student network to learn a more complete and detailed knowledge system.

Local representation teacher distillation: Nodes in the graph network and their neighboring nodes have a high probability of the same class, and the local structure representing nodes in the teacher model is transferred to the student model [20]. Thus, the local representation distillation loss function is defined as follows.

$$L_{local} = \sum_{i \in C} \sum_{j \in N(i)} \alpha_{ij} \|f_i^S(A, X) - f_i^T(A, X)\|_2^2 \quad (1)$$

where $f_S(\cdot)$ represents the embedding of the new class node output by the student model, and $N(i)$ indicates the first-order neighbors of the node in the graph network. α_{ij} is the heat kernel function, which calculates the similarity of the embeddings of the i -th node in the teacher and the j -th node.

Global representation teacher distillation: The global structure of the graph network is preserved by global representation distillation. Specifically, the embedding vectors of the new dataset nodes are extracted using the representation teacher model, and then the mean square error between the embedding vectors of the teacher model and the student model is calculated for each node to maintain the global structure [21]. Therefore, the global representation distillation loss function is as follows.

$$L_{global} = \|f^S(A, X) - f^{T_1}(A, X)\|_2^2 \quad (2)$$

Combining the local representation of distillation loss L_{local} with the global representation of distillation loss L_{global} , the representation distillation loss function is defined as

$$L_{RD} = (1 - \beta)L_{local} + \beta L_{global} \quad (3)$$

where β is the weight parameter, used to adjust the ratio of global distillation and local distillation.

Teacher-based structure distillation: Through structure distillation, the global position encoding structure information is maintained. Specifically, a double-layer position-aware graph neural network (P-GNN) is used to perform position-aware encoding on all nodes in the graph network [22,23]. Any node is set as the initial anchor point, capturing the global position encoding information of the node, aligning the structure features encoded by the student model with those encoded by the structure teacher, to maintain the position-aware encoding.

The first layer P-GNN is for local neighborhood structure encoding, performing local structure encoding on nodes in the global graph network, integrating its own initial features, neighbor features, and local distance information to generate local structure embeddings:

$$h_v^{(1)} = PGNN_1(a_v, \{a_u, d(u, v) | u \in N(v)\}) \quad (4)$$

where a_v is the initial structural feature of the node v , $N(v)$ is the second-order neighbor set of the node v in the graph network, and $d(u, v)$ is the local shortest path distance between nodes u, v .

The second-layer P-GNN performs global position-aware encoding using any node as the anchor point, encodes the global position of the node in the graph network, and generates the final structural embedding:

$$h_v^{PGNN} = PGNN_2(h_v^{(1)}, \{h_a^{(1)}, d(a, v) | a \in A_v = S \setminus \{v\}\}) \quad (5)$$

where $h_v^{(1)}$ is for the first layer local structure embedding of the anchor point, and $d(a, v)$ is the shortest path distance in the global graph network of nodes a and node v . $f^{T_2}(\cdot)$ is defined as the position-aware embedding output by the teacher distillation of the two-layer PGNN structure, using the teacher model to extract the embedding vectors of nodes in the new dataset, and maintaining the position-aware embedding by calculating the mean square error between each node's embedding vector in the structure teacher and the student model embedding vectors:

$$L_{struct} = ||f^S(A, X) - f^{T_2}(A, X)||_2^2 \quad (6)$$

Semantic teacher distillation: We leverage large language models (LLMs) to enrich the semantic content of raw nodes in graph networks. In this work, we adopt mainstream pre-trained language models and open-source LLMs for graph node semantic enhancement. We mainly use Sentence-BERT to transform node text into standard semantic embedding vectors, and employ LLaMA 3 to further explore deep-level semantic information. All enhanced textual content is converted into low-dimensional dense vectors, which serve as the input features of the semantic teacher within our multi-teacher distillation framework [24]. BERT, DeBERTa, E5 and ChatGLM are also widely used alternatives in existing research. The enhanced semantic features are then fed into the graph neural network, and the outputs are defined as the node embeddings generated by semantic teacher distillation. We calculate the mean square error between the embedding vectors of each node from the teacher and student models to preserve the high-order semantic information:

$$L_{Semantic} = ||f^S(A, X') - f^{T_3}(A, X')||_2^2 \quad (7)$$

3.2. Multi-Teacher Weight Optimization Module

After being trained by multiple teachers, the student model outputs the embedding vectors of the support set, query set and unlabeled nodes; based on the probability matrix P , the information entropy of all unlabeled nodes $h = \{h_1, h_2, \dots, h_Q\}$ is calculated, and the m nodes with the minimum information entropy in each category are selected to assign pseudo-labels to them:

$$\sum_{q=1}^Q 1(h_q < \tau_k) \cdot 1(\text{argmax}(p_q) = k) \quad (8)$$

where k represents the k -th category, and it is the information entropy of the m -th node in the k -th category. $\text{argmax}()$ is used to select the category with the highest prediction probability.

The set y' is defined as the collection of the original labels and the pseudo-labels. Soft label predictions p_{ik}^s and hard label predictions y'_{ij} are conducted for the pseudo-label set; the hard prediction results will serve as the supervisory signal, and the cross-entropy loss will be calculated:

$$L_{CE} = - \sum_{i \in P} \sum_{j=1}^N y'_{ij} \ln p_{ij} \quad (9)$$

$$p_{ik}^s = \frac{\exp(z_{i,k}/T)}{\sum_{k'} \exp(z_{i,k'}/T)}$$

The support set and the pseudo-label set are input into the multi-teacher network respectively to obtain the embedding vectors. The weighted average is used to calculate the embedding mean of each category, and then it is input into the prototype network for classification [25]. The expression of the predicted probability of the i -th pseudo-label node belonging to the k -th category in the multi-teacher model is

$$\begin{aligned} p_{ik}^{T_1} &= \frac{\exp(-d(f_T^i(A, X), v_k^{T_1})/T)}{\sum_{k'} \exp(-d(f_T^i(A, X), v_k^{T_1})/T)} \\ p_{ik}^{T_2} &= \frac{\exp(-d(f_{T,PGNN}^i(A, X), v_k^{T_2})/T)}{\sum_{k'} \exp(-d(f_{T,PGNN}^i(A, X), v_k^{T_2})/T)} \\ p_{ik}^{T_3} &= \frac{\exp(-d(f_{T,LLM}^i(A, X), v_k^{T_3})/T)}{\sum_{k'} \exp(-d(f_{T,LLM}^i(A, X), v_k^{T_3})/T)} \end{aligned} \quad (10)$$

Total loss of multi-teacher distillation: In traditional static multi-teacher distillation, all weights w_i^m are usually fixed (such as simple averaging $1/M$) or pre-calculated based solely on the performance of the teachers, and cannot change dynamically during training. This results in inability to adapt to the requirements of different training stages or different samples.

$$L_{MTKD} = L_{CE} + \alpha \sum_{m=1}^M w_i^m D_{KL}(p_i^s, p_i^{T_m}) + \beta(w_1^1 L_{RD} + w_1^2 L_{struct} + w_1^3 L_{Semantic}) \quad (11)$$

where L_{pse} is the cross-entropy task loss; the student model is supervised to directly fit the true labels y_i ; and the logits layer distillation loss measures the KL divergence between the predicted distribution of the student y_i^s and that of the teacher $y_i^{T_m}$. The number of teacher parameters M is three.

Utilizing reinforcement learning agents to dynamically generate teacher weights w_i^m :

Status: In the framework based on reinforcement learning knowledge distillation, the state representation of the input samples x_i is constructed as an embedding vector s_i . This state is composed of the performance of the teacher model (including feature representation, logit vector, and cross-entropy loss) and the difference between the teacher and student models (i.e., feature similarity and probability KL divergence):

(1) **Teacher feature representation** $f_i^{T_m}$: For a given input sample x_i , the m -th teacher feature representation output by the second-to-last layer is defined as $f_i^{T_m} \in R^{d_m}$, where d_m represents the embedding dimension of the m -th teacher.

(2) **Teacher logit vector**: For a given input sample x_i , the logit vector of the m -th teacher after the final output layer is defined as $p_i^{T_m} \in R^C$ (where C is the number of categories), and this logit vector directly predicts the confidence distribution of categories over the entire category space.

(3) **Teacher cross-entropy loss**: For a labeled y_i input sample x_i , the cross-entropy loss of the m -th teacher is defined as $H^{T_m} = H(y_i^{T_m}, y_i)$. The cross-entropy loss is used to measure the fitting performance between the teacher's prediction results and the true labels.

(4) **Teacher–student feature similarity**: Given an input sample x_i , the cosine similarity between the student feature representation $f_i^S \in R^d$ and the teacher feature representation $f_i^{T_m} \in R^{d_m}$ can be expressed by the following formula:

$$\cos_i^{T_m} = r(f_i^S) \cdot f_i^{T_m} / ||r(f_i^S)|| ||f_i^{T_m}|| \quad (12)$$

(5) **Teacher–student probability KL divergence**: Given an input sample x_i , the KL divergence $KL_i^{T_m} = D_{KL}(p_i^S, p_i^{T_m})$ between the student's logit vector and the teacher's logit

vector. The KL divergence is used to measure the degree of difference between the student and the teacher in the probability space.

Concatenate (1) to (5) to construct the state embedding vector. When given an input sample x_i , the state embedding vector s_i^m of the m -th teacher can be expressed as

$$s_i^m = [f_i^{T_m} || p_i^{T_m} || L_{CE}^{T_m} || \cos_i^{T_m} || D_{KL}(\cdot)_i^{T_m}] \quad (13)$$

Action: For each teacher network T_m , an intelligent agent $\pi_{\theta_m}(s_i^m)$ is constructed, where the parameters θ_m of the trainable intelligent agent are represented. The intelligent agent model $\pi_{\theta_m}(s_i^m)$ consists of multiple linear layers, with ReLU activation function in the middle, and softmax function as the output layer; its output result is the weight vector of the teacher w_i^m , and each weight value takes values within the interval (0, 1) in the continuous action space. This process can be expressed as $w_i^m = \pi_{\theta_m}(s_i^m)$.

Reward mechanism: For each teacher network T_m , construct state embeddings s_i^m and guide the intelligent agent $\pi_{\theta_m}(s_i^m)$ to generate teacher weights w_i^m . Use the generated weights to perform weighted multi-teacher knowledge transfer training on the student network. We deploy a standalone lightweight intelligent agent based on multi-layer perceptron (MLP) to conduct adaptive weight decision-making. The agent is optimized via the Adam optimizer with a learning rate of 1e-4. It consists of an input layer, two hidden layers with ReLU activation function, and an output layer equipped with Softmax activation. It takes the state embeddings which combine teacher prediction performance and teacher–student feature and probability discrepancies as input, and outputs normalized weight values for the corresponding teacher. We construct the above state embeddings and feed them into the dedicated agent to generate real-time teacher weights. Afterwards, we utilize the generated weights to perform weighted multi-teacher knowledge transfer training on the student network. The reward function is tightly coupled with the training effect brought by the agent's weight decisions. Ideally, a well-optimized student network has lower classification loss, and its category probability distribution and feature distribution are highly aligned with those of the teacher network. Therefore, the cross-entropy loss between the student network outputs and real labels, the probability KL divergence loss, and the mean square error (MSE) loss between student and teacher feature maps are jointly adopted to construct the following reward function:

$$R_i^m = -L_{CE} - \alpha D_{KL}(p_i^S, p_i^{T_m}) - \beta (w_i^1 L_{RD} + w_i^2 L_{struct} + w_i^3 L_{Semantic}) \quad (14)$$

Policy-based reinforcement learning usually optimizes the agent based on the returned reward value after making a decision. If the agent selects a certain action and receives a higher reward, the policy gradient (PG) will increase the gradient value corresponding to that action. Using the PG method to optimize the parameters of the agent:

$$\theta_m \leftarrow \theta_m - \eta \sum_{i=1}^B \bar{R}_i^m \nabla_{\theta_m} \pi_{\theta_m}(s_i^m), m = 1, 2, \dots, M \quad (15)$$

4. Experimental Results and Analysis

4.1. Datasets

To verify the effectiveness of the proposed MTKD-RL method, experiments were conducted on nine graph network datasets. These include four citation network datasets (Cora [26], CiteSeer [26], CoraFull [26] and DBLP [27]), four Amazon product network datasets (Computers [28], Photo [28], Clothing [29] and Electronics [29]), and one scholar collaboration network dataset (Coauthor [29]). The citation network datasets have papers as nodes and edges representing the citation relationships between papers; the Amazon product network datasets have products as nodes and edges representing the relationships

where products are purchased together; and the collaboration network dataset has scholars in the field of computer science as nodes and edges representing the collaboration relationships between scholars. The experiments were conducted on real and widely applicable graph network datasets.

4.2. Experimental Setup

To verify the effectiveness of the proposed MTKD-RL in the few-shot node classification task, a comparative experiment was conducted against the baseline graph neural network method, the few-shot node classification method based on meta-learning, and the knowledge distillation method and the few-shot node classification method based on self-training.

Graph neural network baseline method: GCN [30]: Updates node embedding features through neighborhood aggregation, applicable to node classification, graph classification, and link prediction tasks; GAT [31]: Learns more accurate node embedding features by calculating weights for adjacent nodes through an attention mechanism.

Few-shot node classification method based on meta-learning: Meta-GNN [32]: Incorporates the parameter initialization and gradient update of the meta-learning paradigm into sub-tasks to achieve rapid generalization ability; G-META [13]: Uses local subgraphs of supporting set instance nodes as learning units to learn transferable knowledge; GPN [14]: For the first time, applies metric learning prototype networks to the few-shot node classification task, calculating the similarity between query set instance nodes and prototypes for node classification; RALE [33]: Captures graph network dependencies and task-level dependencies through absolute and relative position embedding relationships in the graph network. VNT [15]: Uses a Transformer graph neural network as an encoder and generates virtual nodes through soft prompts to mitigate the impact of data scarcity on node classification performance. COLA [16]: Generates sub-task nodes through an encoder, selects nodes similar to query set instances to form a support set, and combines graph supervision contrastive loss to achieve few-shot node classification.

Knowledge distillation method: GraphAKD [34]: Introduces adversarial learning into the knowledge distillation network for the discriminator to capture the distribution differences between the teacher and the student, guiding the student model to precisely imitate the teacher's nodes and categories. SimKD [35]: Through distilling similarity knowledge, aligns the features of the student model with those of the teacher model, and the student uses the classification weights trained by the teacher to optimize the performance of the graph neural network. MTKD-RL-Vision [36]: By leveraging reinforcement learning to optimize the weight distribution of multiple teachers, takes both teacher prediction performance and teacher–student feature discrepancy as agent state input, and continuously updates adaptive teacher weights according to student feedback rewards to enhance knowledge matching performance on visual tasks.

Self-training-based few-shot node classification method: IA-FSNC [12]: Expands the support set instance samples using pseudo-labeled nodes, and utilizes unlabeled graph information for few-shot node classification. PLD-FSNC [17]: Learns the feature representation and classification ability of the teacher through representation distillation and pseudo-label distillation, guiding the student model for new classes.

The few-shot node classification task was set as N-way-K-shot [37,38], where N represents the number of categories of the support set instance nodes, and K is the number of samples in each category. On the Cora, CiteSeer, Computers and Photo datasets, N was set to two, while on the Clothing, Electronics, DBLP, Coauthor, and CoraFull datasets, it was set to five; the values of K were one, three, and five respectively; all models used the Adam optimizer to update the neural network parameters, and the global and local distillation parameters of the teacher were set to 0.3. To ensure the statistical robustness of

the experimental conclusions, a random partition strategy was adopted for each dataset, and 30 independent few-sample learning experiments were conducted; in each experiment, the support set was randomly sampled 30 times, and a total of 900 repeated experiments were completed. The average node classification accuracy was used as the evaluation index, and the results are shown in Tables 1–3. We conducted independent two-sample t-tests at the 95% confidence level to verify the statistical significance of performance differences between MTKD-RL and all baseline models. All obtained p -values of less than 0.05, which indicates that the performance improvement of our method is not caused by random errors and is statistically significant.

Table 1. Comparison of accuracy rates of each model on three datasets: Cora(2Way), Citeseer(2Way) and Computers(2Way).

Methods	Cora(2Way)			Citeseer(2Way)			Computers(2Way)		
	1Shot	3Shot	5Shot	1Shot	3Shot	5Shot	1Shot	3Shot	5Shot
GCN	73.3(2.15)	86.2(1.04)	90.1(0.68)	65.1(1.86)	77.3(1.61)	80.6(1.49)	81.7(3.02)	93.1(0.82)	95.2(0.46)
GAT	74.7(1.55)	87.6(1.34)	90.5(0.33)	66.3(0.45)	78.6(1.00)	81.0(0.45)	83.1(1.33)	93.5(0.74)	95.6(0.74)
Meta-GNN	75.8(1.92)	88.0(1.69)	90.8(0.27)	67.4(1.55)	79.0(1.26)	83.0(1.77)	84.1(2.12)	93.9(0.95)	94.4(0.51)
G-Meta	69.7(0.01)	83.0(0.01)	85.2(0.01)	65.5(0.01)	78.0(0.02)	82.8(0.01)	87.5(0.03)	92.9(0.01)	92.6(0.05)
GNP	74.1(0.13)	85.3(0.20)	88.1(0.27)	68.3(0.11)	82.2(0.13)	85.3(0.15)	89.6(0.87)	95.1(1.34)	94.5(0.48)
RALE	76.5(0.87)	87.6(0.77)	89.9(0.16)	70.7(0.88)	84.6(2.37)	87.1(0.51)	90.5(0.36)	96.3(0.40)	95.0(0.33)
VNT	86.1(0.97)	91.1(0.56)	91.8(0.81)	81.6(0.28)	86.0(0.51)	87.9(0.38)	91.3(0.50)	96.0(0.23)	96.1(0.28)
COLA	88.2(0.65)	92.4(0.30)	93.2(0.40)	82.7(0.61)	87.1(0.33)	89.0(0.85)	91.7(0.90)	96.4(0.55)	97.0(0.12)
GraphAKD	79.8(1.47)	85.7(0.54)	87.8(0.55)	71.2(2.90)	76.4(0.12)	77.4(2.01)	83.9(2.17)	91.3(2.02)	93.3(0.46)
SimKD	80.3(1.02)	87.7(0.21)	89.3(0.20)	75.0(0.05)	82.1(0.87)	83.2(0.52)	85.2(0.65)	92.9(0.33)	95.0(0.22)
MTKD-RL-Vision	82.6(0.07)	90.2(0.41)	91.6(0.71)	77.0(0.65)	83.4(0.35)	85.1(0.22)	87.4(0.34)	93.6(0.46)	95.8(0.30)
IA-FSNC	85.5(1.77)	92.0(0.67)	93.4(0.47)	78.4(2.63)	85.3(1.49)	86.9(1.20)	89.9(2.36)	95.8(0.35)	96.8(0.26)
PLD-FSNC	87.7(1.65)	92.8(0.77)	93.4(0.64)	80.1(2.90)	86.2(1.37)	87.0(1.28)	90.5(2.46)	95.8(0.77)	96.9(0.54)
CMLCKD	90.1(0.40)	94.6(0.10)	94.4(0.24)	84.0(0.47)	88.6(0.40)	91.2(0.61)	93.3(1.02)	98.3(0.32)	98.1(0.37)

Table 2. Comparison of the accuracy rates of each model on the datasets: Photo(2Way), Coauthor(2Way), CoraFull(2Way).

Methods	Photo(2Way)			Coauthor(2Way)			CoraFull(2Way)		
	1Shot	3Shot	5Shot	1Shot	3Shot	5Shot	1Shot	3Shot	5Shot
GCN	84.6(2.81)	94.0(0.78)	94.8(0.49)	86.5(0.74)	92.9(0.11)	93.0(0.28)	58.8(1.75)	77.4(0.88)	83.5(0.62)
GAT	85.6(2.22)	94.6(0.50)	95.1(0.55)	87.1(0.50)	93.0(0.28)	93.7(0.19)	60.1(1.20)	78.2(0.36)	84.2(0.51)
Meta-GNN	86.5(3.01)	92.9(0.77)	93.7(0.38)	87.4(0.31)	93.3(0.25)	94.0(0.15)	61.7(1.96)	79.8(0.36)	85.7(0.74)
G-Meta	82.3(0.04)	88.6(0.04)	89.5(0.15)	86.8(0.19)	91.7(0.23)	93.9(0.13)	64.6(0.29)	73.8(0.24)	77.4(0.42)
GNP	85.6(0.86)	92.3(0.86)	93.6(1.12)	88.3(0.15)	94.5(0.12)	95.1(0.19)	66.5(0.04)	75.8(0.07)	79.1(2.68)
RALE	87.4(0.68)	93.5(0.03)	94.5(0.07)	87.3(0.10)	95.5(0.02)	95.9(0.06)	66.8(1.33)	76.9(0.11)	80.6(0.12)
VNT	90.1(0.35)	94.7(0.20)	95.1(0.30)	89.7(0.65)	93.5(0.33)	95.0(0.20)	74.3(0.65)	84.1(0.31)	84.9(0.03)
COLA	91.9(0.70)	96.0(0.68)	96.9(0.43)	90.4(0.20)	95.9(0.06)	95.7(0.11)	75.3(0.31)	85.0(0.05)	88.7(0.02)
GraphAKD	83.9(1.09)	91.8(0.56)	94.1(1.23)	85.3(0.35)	91.6(0.60)	92.5(1.46)	63.7(1.09)	78.3(0.92)	82.0(0.44)
SimKD	83.2(1.07)	93.0(0.43)	95.2(0.67)	88.0(0.34)	93.6(0.44)	94.8(0.07)	67.2(0.03)	83.9(0.64)	87.2(0.20)
MTKD-RL-Vision	86.1(0.90)	94.7(0.50)	95.3(0.40)	88.1(0.52)	93.7(0.40)	94.8(0.03)	70.6(0.21)	82.5(0.41)	88.3(0.17)
IA-FSNC	90.9(1.81)	95.5(0.45)	96.8(0.27)	88.2(0.70)	94.4(0.11)	94.8(0.07)	72.1(1.63)	81.9(0.82)	85.8(0.59)
PLD-FSNC	91.6(2.26)	96.3(0.67)	96.8(0.39)	89.2(0.71)	94.5(0.19)	95.0(0.19)	73.5(1.47)	84.1(0.61)	87.4(0.45)
CMLCKD	94.2(0.20)	98.2(0.36)	98.9(0.30)	93.5(0.50)	97.2(0.16)	96.9(0.03)	77.7(0.22)	87.5(0.36)	91.2(0.06)

We formulate the multi-teacher knowledge distillation process as a standard reinforcement learning interactive loop. The agent continuously interacts with the distillation environment, where it takes the fused state as input, outputs adaptive teacher weights as actions, and receives reward feedback to update its parameters. We define a training episode as a complete mini-batch containing B graph samples. The whole training follows an alternating optimization paradigm: we first fix all agent parameters to train the student network via weighted multi-teacher distillation, and then optimize the agents using the collected state, action and reward data. The iterative loop runs until both the student network and RL agents converge stably.

Table 3. Comparison of the accuracy rates of each model on the datasets: Clothing(5Way), Electronics(5Way), DBLP(5Way).

Methods	Clothing(5Way)			Electronics(5Way)			DBLP(5Way)		
	1Shot	3Shot	5Shot	1Shot	3Shot	5Shot	1Shot	3Shot	5Shot
GCN	45.6(2.05)	55.3(1.22)	57.9(0.89)	37.0(1.44)	48.6(1.70)	51.1(2.30)	35.1(1.84)	39.0(2.30)	47.7(1.70)
GAT	47.6(2.38)	58.6(1.26)	60.3(0.70)	39.4(0.73)	50.1(0.51)	53.6(0.89)	37.7(0.77)	45.7(1.11)	48.6(1.41)
Meta-GNN	52.8(3.00)	62.5(0.50)	64.3(0.80)	43.5(0.56)	53.2(0.18)	56.1(0.30)	46.9(0.42)	62.2(0.80)	66.1(0.98)
G-Meta	47.1(0.37)	57.8(1.00)	62.3(0.62)	39.4(0.43)	50.1(0.77)	53.2(0.70)	40.1(1.08)	58.6(1.25)	61.8(0.70)
GPN	68.6(0.50)	75.9(0.68)	77.9(0.53)	56.1(0.25)	70.9(0.35)	72.0(0.46)	60.6(0.90)	73.2(0.20)	75.0(0.34)
RALE	71.6(0.30)	79.1(0.57)	81.6(0.22)	62.0(0.54)	73.1(0.07)	76.5(0.09)	66.8(0.56)	74.9(0.34)	77.6(0.05)
VNT	78.6(0.54)	81.7(0.50)	84.3(0.05)	67.4(0.12)	77.6(0.03)	80.7(0.02)	68.4(0.45)	77.0(0.02)	80.0(0.20)
COLA	79.7(0.34)	83.6(0.34)	86.3(0.01)	69.3(0.34)	79.3(0.02)	81.6(0.01)	69.0(0.30)	78.5(0.05)	81.9(0.10)
GraphAKD	74.9(1.03)	80.6(0.61)	83.0(0.11)	64.8(0.22)	74.6(0.11)	77.4(0.60)	62.1(0.70)	72.5(0.45)	75.8(0.34)
SimKD	75.6(1.20)	79.4(1.56)	84.1(0.04)	66.1(0.65)	75.0(0.31)	78.3(0.36)	64.7(0.66)	73.1(0.30)	77.1(0.20)
MTKD-RL-Vision	76.9(1.50)	80.5(0.30)	85.1(0.03)	69.2(0.32)	76.6(0.20)	80.3(0.60)	66.5(0.50)	76.6(0.24)	80.2(0.11)
IA-FSNC	78.3(0.80)	82.3(0.05)	84.8(0.02)	70.3(0.24)	78.3(0.06)	81.3(1.02)	68.0(0.20)	77.1(0.30)	81.9(0.21)
PLD-FSNC	79.6(0.97)	83.7(0.01)	87.5(0.17)	71.6(0.17)	79.0(0.22)	82.5(1.36)	69.5(0.12)	78.9(0.20)	82.2(0.15)
CMLCKD	82.3(0.50)	87.9(0.34)	90.2(0.11)	75.2(0.05)	82.2(0.23)	84.2(0.13)	72.3(0.06)	80.7(0.02)	84.8(0.31)

Training computational overhead: All experiments are executed on a server with NVIDIA Tesla V100S. During the training phase, MTKD-RL alternately executes multi-teacher knowledge distillation and RL agent optimization. The main computation comes from the graph convolution operation of the three teachers. Our method achieves promising performance with only one or two graph convolution layers on most datasets. Fewer convolutional layers effectively reduce the computational cost of graph aggregation. Although the dynamic weight adjustment of the RL module introduces minor extra computation, the overhead is negligible compared with the graph convolution and embedding calculation of the teacher networks. In summary, MTKD-RL achieves significant performance improvements in few-shot node classification with acceptable computational complexity, and has good practical application value.

4.3. Experimental Results

As shown in Tables 1–3, we conduct multi-teacher knowledge distillation comparison experiments for the few-shot node classification task across nine public graph datasets. The experimental results show that our MTKD-RL obtains performance gains ranging from 1.4% to 6.2% on all datasets under different few-shot settings. Compared with existing multi-teacher distillation methods (such as GraphAKD, SimKD), MTKD-RL can better balance the distillation intensity among different teachers and generate high-quality multi-teacher weights suitable for graph-structured data. Compared with the cutting-edge methods in few-shot graph learning (Meta-GNN, G-Meta, RALE, COLA, IA-FSNC, etc.), MTKD-RL achieves improved performance for few-shot node classification. The performance benefits mainly stem from two aspects. On the one hand, its multi-teacher distillation network can provide complementary and comprehensive topological structure and high-level semantic supervision signals to the student model, effectively alleviating the problem of insufficient few-shot annotations. On the other hand, the multi-teacher weight optimization module balances the knowledge distillation intensity among different teachers and strengthens the interaction between students and teachers, and thereby achieves reasonable weight allocation and improves the accuracy of few-shot node classification.

4.4. Ablation Experiment

The proposed MTKD-RL adopts the multi-teacher network distillation technique (C1) to provide comprehensive supervision and guidance information for the graph network, and combines the multi-teacher weight optimization module (C2) to dynamically generate the weights among teachers. The student network prioritizes learning the more valu-

able knowledge from the current teachers, thereby improving the distillation effect. The multi-teacher network distillation technique (C1) includes representative teachers (C1-1), structural teachers (C1-2), and semantic teachers (C1-3). If the weight of the representative teachers is set to 0.5, the weight of the structural teachers to 0.25, and the weight of the semantic teachers to 0.25, the cross-dataset evaluation of different component combinations on the performance of node classification is conducted (Table 4).

Table 4. Classification accuracy rates of different components when tested three times on all datasets.

C1-1	C1-2	C1-3	C2-3	Cora	Citeseer	Computers	Photo	Coauthor	CoraFull	Clothing	Electronics	DBLP
✓				89.4	82.7	91.3	92.5	91.2	81.5	80.7	76.1	70.9
	✓			90.4	83.2	93.7	93.0	92.1	82.3	81.1	77.3	74.9
		✓		88.1	81.6	90.7	93.2	90.6	80.7	79.5	75.5	71.7
✓	✓			91.6	84.7	94.2	94.1	93.4	84.2	82.0	78.4	73.0
✓		✓		92.1	85.1	95.0	93.5	94.3	83.1	83.8	79.7	77.0
	✓	✓		90.2	83.8	93.4	93.9	92.1	82.5	81.7	77.3	74.8
✓	✓		✓	92.3	85.3	95.1	94.5	94.5	86.7	84.7	79.1	77.4
✓		✓	✓	93.1	86.6	96.3	94.9	95.8	85.6	85.3	80.5	78.3
	✓	✓	✓	91.9	84.9	94.9	95.3	93.3	84.2	83.6	78.6	76.5
✓	✓	✓		92.7	86.6	97.1	96.7	95.5	85.5	86.2	80.0	78.8
✓	✓	✓	✓	94.6	88.6	98.3	98.2	97.2	87.5	87.9	82.2	80.7

The results show (taking three-shot as an example) that integrating all components compared to the single-component method leads to an improvement of about 3% on average, which fully proves the collaborative effect of the multi-teacher network distillation technique and the multi-teacher weight optimization module. The student network learns more comprehensive and rich learning goals from the representative teachers, structural teachers, and semantic teachers, strengthening the dynamic generation of weights among teachers, balancing the guidance information among teachers, solving the problem of multi-teacher knowledge conflicts, and maximizing the performance gain of multi-teacher distillation.

In addition, adding a weight optimization module based on reinforcement learning to the dual-teacher/multi-teacher can dynamically adjust the weights among teachers and balance the influence of each teacher on the student model; all three types of single components of a single teacher can provide effective supervision for the student network, and the performance differences in the three types of single components indicate that the structural information has the most significant gain for the graph node classification task, the representative information is second, and the semantic information can be an effective supplement. The three provide rich supervision information from different dimensions for the student network, laying the foundation for multi-teacher collaborative distillation.

Further, we analyze the inherent advantages of the proposed RL-based dynamic weighting mechanism compared with traditional fixed-weight strategies. In conventional multi-teacher distillation with fixed weights, the weight of each teacher remains constant throughout training. This static setting cannot adapt to the fact that different teachers have different generalization capabilities on various graph samples, nor respond to the dynamic changes in teacher–student feature alignment in different training epochs. As a result, conflicting knowledge and redundant supervision information from multiple teachers will degrade the distillation effect.

In our framework, the RL agent takes teacher performance and teacher–student gaps as state features, and outputs dynamic weights for each sample and training stage. Specifically, for samples where a certain teacher has superior prediction performance and high feature alignment with the student, the agent will automatically increase its weight to absorb high-quality knowledge. By contrast, weights of teachers with weak performance and large distribution discrepancies will be reduced to avoid interfering knowledge. This adaptive allocation effectively alleviates knowledge conflicts and information redundancy, enables

the student model to selectively learn valuable supervision signals, and fundamentally improves the efficiency and final performance of multi-teacher knowledge distillation.

4.5. Investigating the Impact of the Number of Graph Convolution Layers in Characterizing Teachers on the Distillation Effect

To further explore the influence of the number of graph convolution layers used to characterize teachers on the effect of knowledge distillation, the source model and the target model adopt the same number of graph convolution layers. By comparing the results of three knowledge distillation methods, GraphAKD, SimKD, and PLD-FSNC (as shown in Figure 2), the proposed MTKD-RL method can achieve excellent results with a relatively small number of graph convolution layers (only one layer for CiteSeer and Photo datasets, and two layers for the other six datasets). Moreover, it significantly outperforms GraphAKD, SimKD, and PLD-FSNC across different layer configurations, fully verifying the effectiveness of the distillation method. Secondly, as the number of layers increases, the performance of all methods on each dataset shows a downward trend. This indicates that when the depth of the convolution graph layers is large, the node embeddings may be damaged due to the problem of excessive smoothing (i.e., the node embedding values are highly similar). However, the proposed method exhibits a more gradual performance decline curve compared to other methods on all datasets, confirming the stability and robustness of the distillation method.

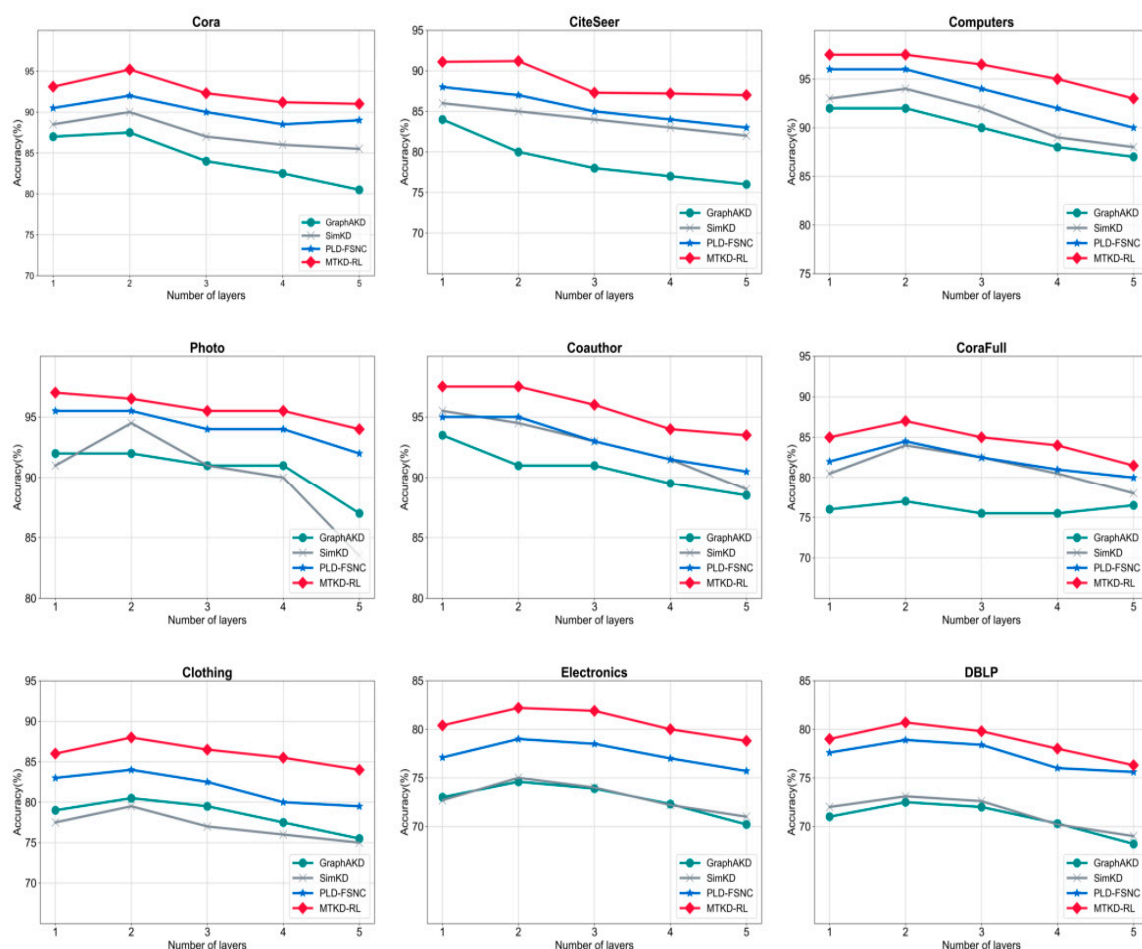


Figure 2. The distillation effects of the four methods at different graph convolutional layers when conducting 3-shot experiments on all datasets, to evaluate their robustness and adaptability to changes in the architecture depth.

4.6. Visualization of Entropy Dynamics During Self-Training Iteration

To better understand the regularization effect of the objective function based on KL divergence, the trend of the predicted entropy of PLD-FSNC (single teacher) and MTKD-RL (multi-teacher model) during the self-training iteration process is visualized. Specifically, on the Cora, CiteSeer, and Coauthor datasets, four rounds of iterative training were conducted for one-shot, three-shot, and five-shot datasets, and the average information entropy of the test prediction results was reported, as shown in Figure 3.

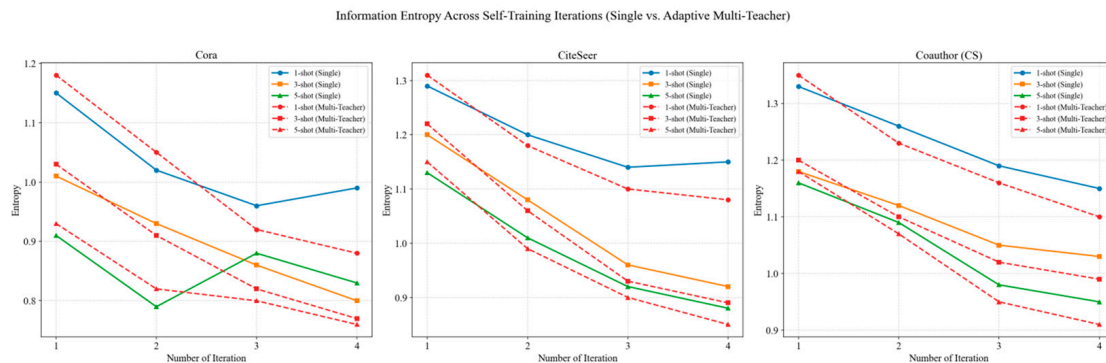


Figure 3. The results of four rounds of 1-shot, 3-shot and 5-shot iterative training on the Cora, CiteSeer and Coauthor datasets.

The higher the predicted entropy, the more uncertain and more uniform the model's prediction for node categories; the lower the entropy, the more confident the model's prediction and the more concentrated the probability distribution. During self-training, multiple teachers did not collaborate in the early stage, and the uncertainty was higher compared to the single-teacher model; as the iterative training progressed, the multi-teacher model based on representation, structure, and semantics had a more stable process of predicted entropy decrease, proving that the learning process of the multi-teacher model is more stable and has less pseudo-label noise, which can avoid the problem that the single-teacher model is easily affected by its own bias and the predicted entropy is prone to fluctuation. This guides the student model to learn more robust node feature representations; ultimately, the supervisory signal after multi-teacher fusion is more robust than that of the single-teacher, and the student model can learn more accurate distributions, with the predicted entropy being lower than that of the single-teacher model.

4.7. Parameter Sensitivity Analysis

The study verified that the number of pseudo-labels and the number of self-training iterations have a significant coupling effect on the model performance. Using the DBLP dataset (five-way–three-shot), it was found that the model accuracy reached its peak at $m = 50$ (as shown in Figure 4). When the number of pseudo-labels is too small, the insufficient supervision signal leads to model underfitting; when the number of pseudo-labels is too large or the number of iterations is too high, the cumulative effect of noisy pseudo-labels significantly reduces performance. This result indicates that the multi-teacher model in reinforcement learning has achieved an effective balance between the quality and quantity of pseudo-labels, and can stably improve the accuracy of few-shot node classification within a reasonable parameter range.

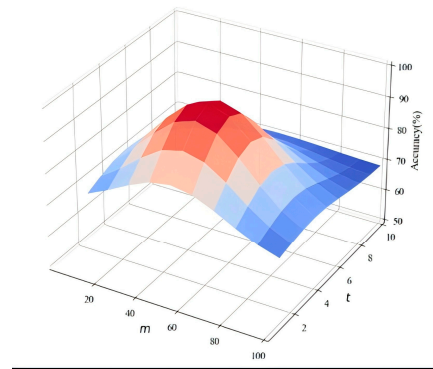


Figure 4. Parameter and sensitivity analysis of the 3-shot experiment conducted on the DBLP dataset.

5. Conclusions

This paper proposes a reinforcement learning multi-teacher knowledge distillation method MTKD-RL for few-shot node classification. It provides complementary supervision through three types of teachers—representation, structure, and semantics. By using reinforcement learning agents to dynamically optimize the weights of the teachers, it effectively improves the quality of pseudo-labels and the effect of knowledge distillation. The experimental results on nine benchmark graph datasets show that this method achieves the best classification performance in various few-shot settings and significantly outperforms existing self-training and knowledge distillation methods. In the future, by integrating multimodal graph data with lightweight distillation strategies, the method can be applied practically in real-time inference and edge-side deployment.

Author Contributions: D.X. contributed to methodology, formal analysis, investigation, data curation, visualization, writing—original draft, and writing—review and editing. W.S. contributed to conceptualization. B.Z. contributed to supervision, project administration, resources, and writing—review and editing. Y.L. contributed to validation. R.G. contributed to formal analysis. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China under the project "Research on Radio Frequency Fingerprint Identification of Low-Power Wide-Area Network Based on Multi-Feature Fusion and Convolutional Neural Networks" (Grant No. 62306207).

Data Availability Statement: Data are available from the corresponding author upon reasonable request.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Wang, S.; Ding, K.; Zhang, C.; Chen, C.; Li, J. Task-adaptive few-shot node classification. In Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, 14–18 August 2022; pp. 1910–1919.
2. Wang, Z.; Yin, Z.; Zhang, Y.; Yang, L.; Zhang, T.; Pissinou, N.; Cai, Y.; Hu, S.; Li, Y.; Zhao, L.; et al. Fg-smote: Towards fair node classification. *ACM SIGKDD Explor. Newsl.* **2025**, *26*, 99–108. [\[CrossRef\]](#)
3. Wei, M.; Lu, S.; Wang, Y. GraphSAM: A similarity-aware mixup method based on optimal transport for imbalanced node classification. *Expert Syst. Appl.* **2026**, *324*, 132527. [\[CrossRef\]](#)
4. Xu, F.; Huang, Y.; Wang, H.; Fan, Z. A novel heterogeneous data classification approach combining gradient boosting decision trees and hybrid structure model. *Pattern Recognit.* **2025**, *165*, 111614. [\[CrossRef\]](#)
5. Mao, Z.; Ju, W.; Yi, S.; Wang, Y.; Xiao, Z.; Long, Q.; Yin, N.; Liu, X.; Zhang, M. Learning knowledge-diverse experts for long-tailed graph classification. *ACM Trans. Knowl. Discov.* **2025**, *19*, 32. [\[CrossRef\]](#)
6. Chen, J.; Mi, R.; Wang, H.; Wu, H.; Mo, J.; Guo, J.; Lai, Z.; Zhang, L.; Leung, V.C. A review of few-shot and zero-shot learning for node classification in social networks. *IEEE Trans. Comput. Soc. Syst.* **2024**, *12*, 1927–1941.
7. Zhang, B.; Chen, M.; Song, J.; Li, S.; Zhang, J.; Wang, C. Normalize then propagate: Efficient homophilous regularization for few-shot semi-supervised node classification. *Proc. AAAI Conf. Artif. Intell.* **2025**, *39*, 13170–13178. [\[CrossRef\]](#)

8. Ma, X.; Zhai, J.; Chen, H. Contextual learning and feature tuning for few-shot node classification. *Inf. Sci.* **2025**, *719*, 122494. [[CrossRef](#)]
9. Chen, X.; Yue, K.; Liu, D.; Liu, W.; Duan, L.; Li, A. Structural entropy guided meta-learning for few-shot node classification. *IEEE Trans. Knowl. Data Eng.* **2026**, *38*, 3154–3167. [[CrossRef](#)]
10. Wang, S.; Dong, Y.; Ding, K.; Chen, C.; Li, J. Few-shot node classification with extremely weak supervision. In *Proceedings of the 16th ACM International Conference on Web Search and Data Mining*; Association for Computing Machinery: New York, NY, USA, 2023; pp. 276–284.
11. Xu, Z.; Ding, K.; Wang, Y.X.; Liu, H.; Tong, H. Generalized few-shot node classification: Toward an uncertainty-based solution. *Knowl. Inf. Syst.* **2024**, *66*, 1205–1229.
12. Wu, Z.; Zhou, P.; Wen, G.; Wan, Y.; Ma, J.; Cheng, D.; Zhu, X. Information augmentation for few-shot node classification. In *Proceedings of the International Joint Conference on Artificial Intelligence*, Vienna, Austria, 23–29 July 2022; pp. 3601–3607.
13. Huang, K.; Zitnik, M. Graph meta learning via local subgraphs. In *Advances in Neural Information Processing Systems 33 (NeurIPS 2020)*; Curran Associates, Inc.: Red Hook, NY, USA, 2020; pp. 5862–5874.
14. Ding, K.; Wang, J.; Li, J.; Shu, K.; Liu, C.; Liu, H. Graph prototypical networks for few-shot learning on attributed networks. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*; Association for Computing Machinery: New York, NY, USA, 2020; pp. 295–304.
15. Tan, Z.; Guo, R.; Ding, K.; Liu, H. Virtual node tuning for few-shot node classification. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*; Association for Computing Machinery: New York, NY, USA, 2023; pp. 2177–2188.
16. Liu, H.; Feng, J.; Kong, L.; Tao, D.; Chen, Y.; Zhang, M. Graph contrastive learning meets graph meta learning: A unified method for few-shot node tasks. In *Proceedings of the ACM Web Conference 2024*; Association for Computing Machinery: New York, NY, USA, 2024; pp. 365–376.
17. Wu, Z.; Zhou, P.; Wen, G.; Zhu, X. A pseudo-labeling approach based on knowledge distillation for graph few-shot learning. *Inf. Process. Manag.* **2025**, *62*, 104268. [[CrossRef](#)]
18. Zhang, L.; Wang, S.; Liu, J.; Chang, X.; Lin, Q.; Wu, Y.; Zheng, Q. MuL-GRN: Multi-level graph relation network for few-shot node classification. *IEEE Trans. Knowl. Data Eng.* **2022**, *35*, 6085–6098.
19. Wang, N.; Lai, Y.; Chen, C.; Zheng, Z. Prototypical networks with unlabeled data for few-shot node classification. *Eng. Appl. Artif. Intell.* **2025**, *145*, 110088. [[CrossRef](#)]
20. Yang, C.; Zhu, Y.; Lu, W.; Wang, Y.; Chen, Q.; Gao, C.; Yan, B.; Chen, Y. Survey on knowledge distillation for large language models. *ACM Trans. Intell. Syst. Technol.* **2025**, *16*, 143. [[CrossRef](#)]
21. Xiao, Z.; Xing, H.; Qu, R.; Li, H.; Cheng, X.; Xu, L.; Feng, L.; Wan, Q. Heterogeneous mutual knowledge distillation for wearable human activity recognition. *IEEE Trans. Neural Netw. Learn. Syst.* **2025**, *36*, 16589–16603. [[CrossRef](#)] [[PubMed](#)]
22. Venkatesh, K.; Dhasmana, G.; Deokar, S.U.; Thacker, C.; Saravanan, K.G.; Padmashini, R.K. Osprey optimization algorithm and position-aware graph neural network for enhanced vehicle-to-grid control system. In *Proceedings of the 5th International Conference on Trends in Material Science (ICTMIM)*; IEEE: Piscataway, NJ, USA, 2025; pp. 306–311.
23. Chen, J.; Wu, J.; Chen, J. Position-aware graph transformer for recommendation. *ACM Trans. Inf. Syst.* **2025**, *43*, 165. [[CrossRef](#)]
24. Chen, R.; Zhao, T.; Jaiswal, A.; Shah, N.; Wang, Z. Llaga: Large language and graph assistant. *arXiv* **2024**, arXiv:2402.08170.
25. Snell, J.; Swersky, K.; Zemel, R. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems 30 (NIPS 2017)*; Curran Associates, Inc.: Red Hook, NY, USA, 2017.
26. Liu, Y.; Li, M.; Li, X.; Giunchiglia, F.; Feng, X.; Guan, R. Few-shot node classification on attributed networks with graph meta-learning. In *Proceedings of the 45th ACM SIGIR Conference on Research and Development in Information Retrieval*; Association for Computing Machinery: New York, NY, USA, 2022; pp. 471–481.
27. Bhagat, S.; Cormode, G.; Muthukrishnan, S. Node classification. In *Social Network Data Analytics*; Springer: Boston, MA, USA, 2011; pp. 115–148.
28. Wu, Q.; Zhao, W.; Li, Z.; Wipf, D.P.; Yan, J. Nodeformer: A scalable graph structure learning transformer. In *Advances in Neural Information Processing Systems 35 (NeurIPS 2022)*; Curran Associates, Inc.: Red Hook, NY, USA, 2022; pp. 27387–27401.
29. Rong, Y.; Huang, W.; Xu, T. Dropedge: Towards deep graph convolutional networks. *arXiv* **2019**, arXiv:1907.10903.
30. Yao, L.; Mao, C.; Luo, Y. Graph convolutional networks for text classification. *Proc. AAAI Conf. Artif. Intell.* **2019**, *33*, 7370–7377. [[CrossRef](#)]
31. Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Lio, P.; Bengio, Y. Graph attention networks. *arXiv* **2017**, arXiv:1710.10903.
32. Zhou, F.; Cao, C.; Zhang, K.; Trajcevski, G.; Zhong, T.; Geng, J. Meta-GNN: On few-shot node classification in graph meta-learning. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*; Association for Computing Machinery: New York, NY, USA, 2019; pp. 2357–2460.
33. Liu, Z.; Fang, Y.; Liu, C.; Hoi, S.C. Relative and absolute location embedding for few-shot node classification on graph. *Proc. AAAI Conf. Artif. Intell.* **2021**, *35*, 4267–4275. [[CrossRef](#)]

34. Wang, J.; Zhou, M.; Zhang, S.; Gong, Z. Generalized few-shot node classification with graph knowledge distillation. *IEEE Trans. Comput. Soc. Syst.* **2024**, *12*, 1805–1815. [[CrossRef](#)]
35. Wang, L.; Kuk, Y. Knowledge distillation and student-teacher learning for visual intelligence. *IEEE Trans. Pattern Anal. Mach. Intell.* **2021**, *44*, 3048–3068. [[CrossRef](#)]
36. Yang, C.; Yu, X.; Yang, H.; An, Z.; Yu, C.; Huang, L.; Xu, Y. Multi-teacher knowledge distillation with reinforcement learning for visual recognition. *Proc. AAAI Conf. Artif. Intell.* **2025**, *39*, 9148–9156. [[CrossRef](#)]
37. Wang, S.; Tan, Z.; Liu, H.; Li, J. Contrastive meta-learning for few-shot node classification. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*; Association for Computing Machinery: New York, NY, USA, 2023; pp. 2386–2397.
38. Zhao, H.; Yang, B.; Cen, Y.; Ren, J.; Zhang, C.; Dong, Y.; Kharlamov, E.; Zhao, S.; Tang, J. Pre-training and prompting for few-shot node classification on text-attributed graphs. In *KDD '24: Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*; Association for Computing Machinery: New York, NY, USA, 2024; pp. 4467–4478.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.