Expert Systems With Applications 285 (2025) 127868

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

FSAKE: Few-shot graph learning via adaptive neighbor class knowledge embedding

Linhua Zou, Jie Jin, Dongqing Li, Hong Zhao*

School of Computer Science, Minnan Normal University, Zhangzhou, Fujian, 363000, China
The Key Laboratory of Data Science and Intelligence Application, Minnan Normal University, Zhangzhou, Fujian, 363000, China

ARTICLE INFO

Keywords:
Few-shot learning
Graph neural networks
Graph pooling

ABSTRACT

Few-shot graph learning tackles the challenge of categorization with limited samples by utilizing the relational information encoded on the graph. Recent studies have acquired considerable success in directing query node inference on graphs by exploiting class knowledge. However, these methods independently assess the importance of different class knowledge and ignore the interrelatedness of neighbor class knowledge. In this paper, we propose an adaptive neighbor class knowledge embedding (FSAKE) consisting of knowledge filtering (KF) and correction (KC), aiming to exploit the interconnected neighbor class knowledge to enhance the recognition of crucial class features. KF integrates neighbor information into the node scoring process during the graph pooling phase, ensuring the preservation of inter-class similar and intra-class representative nodes to learn important class knowledge further. Meanwhile, KC introduces an additional correction loss during the graph unpooling phase, combined with the classification loss, to supervise the deep propagation of class knowledge. Experimental results on four popular benchmark few-shot datasets illustrate the effectiveness of FSAKE. For instance, under the 5-way 1-shot setting, FSAKE outperforms other models on miniImageNet and CIFAR-FS with an accuracy improvement of at least 1.32% and 2.11%. Code is available at https://github.com/fhqxa/FSAKE.

1. Introduction

Few-shot learning (FSL) has been a hot research topic in recent years, which aims to build an efficient model based on limited labeled training examples (Li et al., 2023). Despite deep learning models having exhibited impressive success in computer vision, they mainly rely on huge amounts of labeled data (Zheng et al., 2024). However, obtaining abundant samples for real-world problems is often challenging (Yan et al., 2024). This scenario is known as the FSL problem, where only a few labeled samples are available for each category (Lim et al., 2024). Inspired by human rapid learning ability, FSL models have been developed to identify new instances from limited labeled samples and applied to various computer vision tasks (Zhao, Su, et al., 2024), such as image classification (Jin et al., 2024) and object detection (Xin et al., 2024).

Most existing FSL approaches can be divided into based on meta-learning and graph neural networks (GNNs) (Yu et al., 2022). The FSL approaches based on meta-learning aim to obtain meta-knowledge from different tasks and transfer it to new tasks (Zhu et al., 2022). Existing meta-learning approaches are classified into optimization-based and metric-based (Yu et al., 2022). The optimization-based methods meta-learned a good initialization that adapted to new tasks rapidly. Several remarkable optimization algorithms like gradient-based methods (Finn et al., 2017; Raghu et al., 2020) have been proposed to learn good initialization parameters for fast adaption to new tasks. In contrast, other metric-based approaches meta-learned representations that transferred across tasks without fine-tuning during the test phase. For example, similarity between samples of the support and query class is usually indicated via distance (Li et al., 2019; Snell et al., 2017) or relationship measurements (Sung et al., 2018; Xie et al., 2022). The aforementioned meta-learning approaches mainly focus on task-level relationships and ignore the interrelations among samples within the task.

Unlike the FSL methods based on meta-leaning, the FSL methods based on GNNs (Kim et al., 2019; Satorras & Estrach, 2018; Zhao, Huang, & Wang, 2024) focus on task-level relationships and explore the relationships among samples in tasks. FSL methods based on GNNs are categorized into node-level, edge-level, and graph-level (Lu et al., 2023; Yang et al., 2023). Node-level FSL methods (Satorras & Estrach, 2018; Zhang, Zhang, et al., 2022) transfer node features on the graph

* Corresponding author.
E-mail addresses: g2023061023@stu.mnnu.edu.cn (L. Zou), g2022061016@stu.mnnu.edu.cn (J. Jin), g2023061041@stu.mnnu.edu.cn (D. Li), hongzhaocn@163.com (H. Zhao).

https://doi.org/10.1016/j.eswa.2025.127868
Received 24 June 2024; Received in revised form 21 February 2025; Accepted 22 April 2025
Available online 7 May 2025

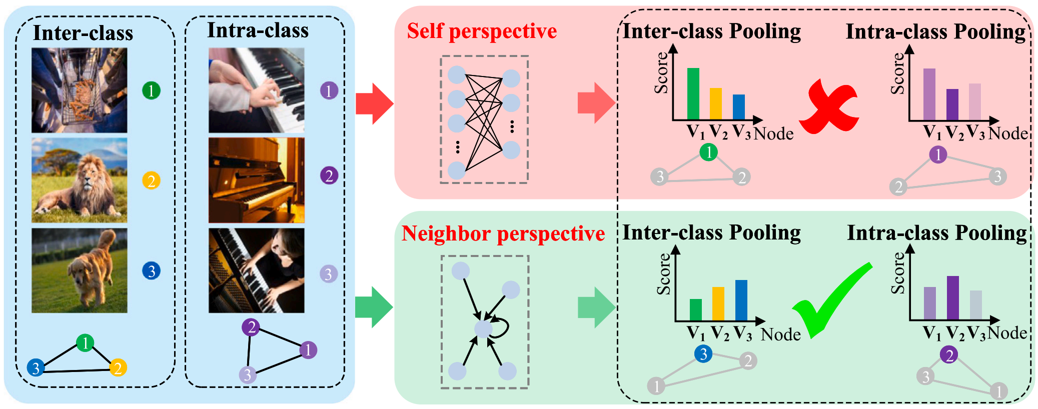

Fig. 1. Comparison of our graph pooling method with existing graph pooling methods for FSL. The blue parts indicate the initial inter-class and intra-class task graphs. The red part denotes that only the own node features are considered for selecting the pooling node. The green part illustrates our pooling node selection method that considers the node own features and neighbor class information.

to propagate support nodes information to query nodes. Edge-level FSL methods (Kim et al., 2019; Yang et al., 2020) are considered from modeling the edge features that control the degree of node information aggregation. Unlike the above methods that focus primarily on the correlation between query and support samples, graph-level FSL methods utilize the rich class knowledge among samples (Yang et al., 2023). For example, Chen, Li, et al. (2021) and Zhao, Su, et al. (2024) leverage the idea of pooling to compress the nodes in the original instance graph to obtain class knowledge with high-level representations. These models fuse instance knowledge and class knowledge to obtain rich sample representations.

However, most existing graph-level FSL approaches only focus on the intrinsic features of individual nodes and ignore valuable neighbor node class information. This limits the deep mining of significant class knowledge, as it is hard to correctly identify influential nodes for further deep learning. As shown in the "Self perspective" of Fig. 1, they calculate node importance scores based on the intrinsic features of individual nodes through linear or nonlinear networks. Inter-class: Node 1 has distinctly different features compared with other categories, resulting in a higher score and being selected for further learning. In fact, such easily distinguishable samples obviously do not require additional learning. Intra-class: Since both node 1 and node 3 contain hands, the model may prefer to choose this shortcut for further learning. As a result, node 1 receives a higher score and is selected for further learning. Additionally, incorrectly learning or propagating class knowledge in the hierarchical structure of graph-level FSL tends to accumulate and accelerate classification errors.

In this paper, we propose a few-shot graph learning with loss correction based on adaptive neighbor class knowledge embedding (FSAKE). FSAKE comprises knowledge filtering (KF) and knowledge correction (KC) strategies. First, we propose KF to extract significant and representative class knowledge. We utilize the feature information of the nodes themselves and aggregate the information of first-order neighbors to evaluate the importance of nodes. This approach ensures that nodes primarily share common features and classes similar to each other are retained during the graph pooling phase. As shown in the "Neighbor perspective" of Fig. 1, our method combines the neighbor class information to select the correct inter-class similar samples and intra-class representative samples. For inter-class, we should select difficult samples for further learning, as in node 3. For intra-class, the intra-class representative samples should be selected for further learning, as in node 2. Next, we recognize the potential risk for incorrect propagation in hierarchical pooling processes and propose KC to counteract this. In particular, we introduce a correction loss function, which works with classification loss to reduce the incorrect propagation of class knowledge.

We present experimental results on four benchmark datasets to illustrate the effectiveness of FSAKE. For instance, under the 5-way 1-shot setting, FSAKE has achieved at least 0.95%, 1.36%, 1.32%, and 2.11% better than the baseline model on miniImageNet, CUB-200-2011, tieredImageNet, and CIFAR-FS, respectively. The main contributions of this paper are summarized as follows:

- We propose a knowledge filtering strategy that comprehensively considers the neighbor information and its features of nodes during the pooling phase. This allows correctly learning inter-class similar and intra-class representative class knowledge, enhancing the model ability to recognize crucial class features.
- We introduce a knowledge correction strategy that combines correction loss and classification loss, which dynamically adjusts the propagation of class knowledge during the unpooling phase. This strategy optimizes the propagation process of class knowledge.
- Extensive experiments on four common FSL benchmark datasets validate that FSAKE is effective. The experimental results demonstrate that FSAKE has improved remarkably in few-shot classification tasks.

The remaining sections are organized as follows. In Section 2, we review the related work. Next, we detail the FSAKE model in Section 3. Experimental settings and results analysis are presented in Section 4. Finally, we draw a brief conclusion and discuss future work in Section 5.

2. Related work

In this section, we review relevant literature on few-shot learning based on meta-learning and graph neural networks.

2.1. Few-shot learning based on meta-learning

Meta-learning is the learning paradigm for most FSL approaches, which exploit episodic training to obtain a generalized model for new tasks (Yu et al., 2022). According to meta-learning content, existing methods are grouped into optimization-based and metric-based (Li et al., 2023).

Optimization-based FSL approaches obtain good initialization by training a sequence of tasks to quickly identify new tasks (Shi et al., 2023). For instance, a Model-Agnostic Meta-Learning (MAML) framework is established by Finn et al. (2017) to learn a good initialization. On this basis, Raghu et al. (2020) found that feature reuse is key for learning through further exploring the effectiveness of MAML. Similarly, Rusu et al. (2019) proposed latent embedding optimization to decouple gradient-based adaptive processes.

Metric-based FSL methods focus on constructing an embedding space that distinguishes different classes by distance metric (Shi et al., 2023). Metric-based FSL methods include distance and relation metrics (Zhao, Su, et al., 2024). The former leverages distance formulas to calculate the similarity of two samples. For example, Prototypical Networks (Snell et al., 2017) adopted Euclidean distance, Matching Networks (Vinyals et al., 2016) and DN4 (Li et al., 2019) utilized cosine distance. In contrast, the latter focuses on learning relation modules via similarity-based metric learning with deep neural networks (Hui et al., 2019; Sung et al., 2018). For example, Kang et al. (2021) combined global and local classifiers to learn relation embeddings. Zhang et al. (2020) embedded the Earth Mover's distance to compute the structural relation among dense samples. Additionally, Xie et al. (2022) simultaneously consider the pairwise relations and the collective distribution properties of embedded features.

Optimization-based FSL methods adapt to new tasks, while metric-based FSL methods obtain a task-invariant metric that is suitable for all tasks. Unlike these methods that only investigate task-level relationships, our approach explores the relationships among samples within the task through GNNs while preserving task-level relationships.

2.2. Few-shot learning based on graph neural networks

GNNs is a powerful technique for exploring the relationship between support classes and query classes (Satorras & Estrach, 2018). Recently, there has been increasing attention to utilizing GNNs for FSL tasks (Kim et al., 2019; Yang et al., 2020; Zhao, Su, et al., 2024). GNNs-based FSL methods are divided into node-level FSL, edge-level FSL, and graph-level FSL (Lu et al., 2023; Yang et al., 2023).

Node-level FSL methods propagated the node features over the graph, passing information from support nodes to query nodes. Satorras and Estrach (2018) first exploited GNNs for FSL, which created a node feature vector by concatenating the sample feature and one-dimensional label vector. Based on the episodic training mechanism and inductive setting, the query nodes on the graph were predicted by training meta-graph parameters. Unlike the inductive setting, Liu et al. (2019) introduced the transductive setting, exploiting both query and support samples for simultaneous prediction of the whole query set. Building upon these advances, Zhang, Zhang, et al. (2022) proposed a GNN-guided data hallucination network to generate discriminatively strengthened synthetic data.

Edge-level FSL methods constructed edge features and edge update networks through different methods based on node features. Kim et al. (2019) fully explored the internal information of the graph by defining class labels and edge labels instead of relying solely on node labels. Furthermore, Tang et al. (2021) integrated GNN with conditional random field (CRF) into a framework. They update edge features by modeling graph affinity as paired marginal probabilities in CRF. In contrast, Yang et al. (2020) leveraged distributed features to enrich the representation of edge features. They updated the point and distribution graphs to encode node and distribution features into edge features.

Graph-level FSL methods exploit class knowledge to direct query node inference on the graph. Chen, Yang, et al. (2021) utilized existing categories to create new abstract categories and explicitly learned rich class knowledge to direct query sample reasoning on graphs. Moreover, Yu et al. (2022) regarded GNNs as a feature optimization module from different perspectives. They utilized prototype GNN and instance GNN to acquire discriminative class knowledge. Similarly, Chen, Li, et al. (2021) and Zhao, Su, et al. (2024) down-sampled support nodes through intra-class and inter-class graph pooling layers to extract class knowledge from samples.

Node-level FSL methods aim to propagate node information on the graph for node classification, which is a simple and effective method. Compared with edge-level FSL methods, it usually uses the similarity between node features to construct the adjacency matrix, leading to insufficient exploration of the correlation between nodes. This tends to lead to inappropriate feature aggregation. In contrast, Edge-level FSL methods carefully design edge features that can control the aggregation of node information. On the basis of these two methods, graph-level FSL methods that incorporate class knowledge can match higher reliability categories for query samples. Moreover, the average performance of these methods for 5-shot classification tasks has significantly improved.

Inspired by the aforementioned graph-level FSL methods based on class knowledge, we explore more representative and accurate class knowledge in conjunction with neighbor node class information. Unlike Chen, Li, et al. (2021) and Zhao, Su, et al. (2024), only considered the intrinsic features of individual nodes, our method adaptively integrates neighbor class information to provide an objective evaluation for the significance of various class knowledge. Additionally, we introduce a knowledge correction strategy to prevent the incorrect propagation of class knowledge during the training process.

3. Methodology

Starting this section, we present some fundamental concepts and definitions of graph neural networks relevant to our approach. Then we briefly introduce the entire model. Next, we detail two sub-modules of the model. Finally, we conduct model analysis and learning.

3.1. Basic definitions and concepts of graph neural networks

To better understand our proposed approach, we start with a brief introduction to graph neural networks (GNNs) and their basic concepts. GNNs are designed to operate graph structure data, where nodes represent entities and edges denote relationships between these entities.

Graph representation. A graph is formally defined as $G = (\mathbf{V}, \mathbf{E})$, where $\mathbf{V}$ is the set of nodes and $\mathbf{E}$ is the set of edges. Every node $v_i \in \mathbf{V}$ may have associated features, typically represented as follows:

$$v_i = (f_\phi(x_i)), \tag{1}$$

where $f_\phi(\cdot)$ denotes the feature extractor. In the context of our work, node v_i is a 133-dimensional feature vector, which consists of 128-dimensional feature vectors of an image and the corresponding 5-dimensional one-hot label vector.

The relationships between nodes are represented by the adjacency matrix $\mathbf{A}$, where $a_{ij} \in \mathbf{A}$ indicates the presence and strength of the connection between nodes v_i and v_j. The adjacency matrix $\mathbf{A}$ is generally updated by a neural network as follows:

$$a_{i,j} = g_{\overline{\theta}}(abs(v_i - v_j)), \quad i, j = 1, \dots, n, \tag{2}$$

where $abs(\cdot)$ denotes the absolute difference between the features of two nodes, $g_{\overline{\theta}}(\cdot)$ represents utilizing the neural networks to update the adjacent matrix, n denotes the total number of nodes. In our work, the $a_{i,j}$ represents the relationships between samples in the FSL task and is updated in this way.

Message passing in GNNs. The node features of standard GNNs are iteratively updated by aggregating neighborhood information. The general update rule for a node v_i in GNNs can be expressed as follows:

$$\mathbf{h}_i^{(l+1)} = \sigma\left(\sum_{j\in\mathcal{N}(i)} \mathbf{W}^{(l)}\mathbf{h}_j^{(l)} + \mathbf{b}^{(l)}\right), \tag{3}$$

where $\mathbf{h}_i^{(l)}$ is the representation of node v_i at layer l, $\mathcal{N}(i)$ denotes the set of neighbors of node v_i, $\mathbf{W}^{(l)}$ and $\mathbf{b}^{(l)}$ are learnable parameters, and σ refers to a non-linear activation function. In the specific implementation of our proposed method, we utilize a modified graph convolutional neural network for updating node features.

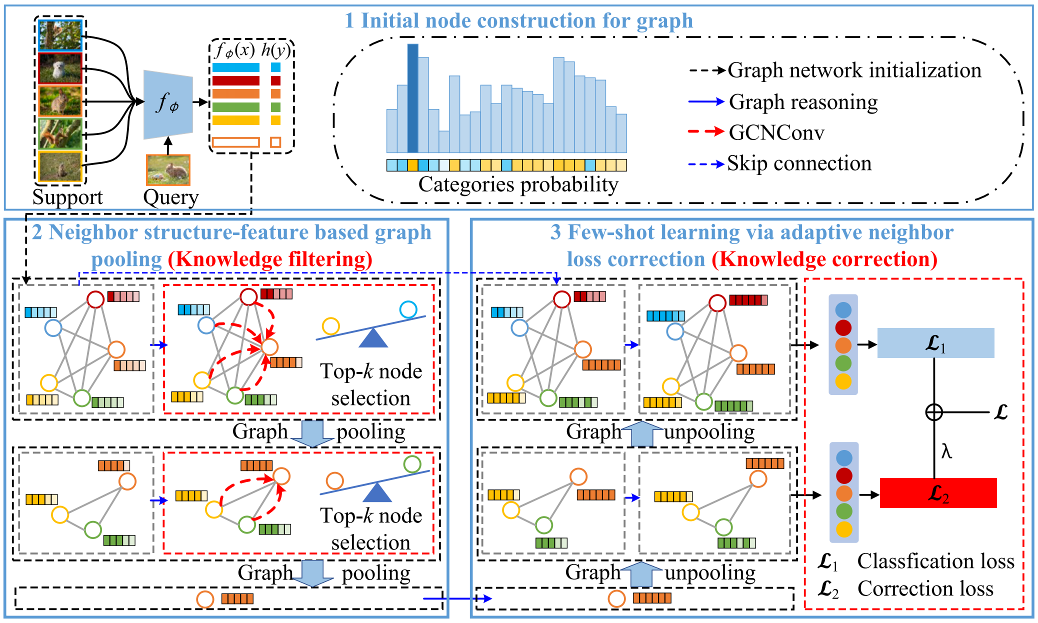
1 Initial node construction for graph
f_ϕ
$f_\phi(x)$ $h(y)$
Support
Query
Categories probability
Graph network initialization
Graph reasoning
GCNConv
Skip connection
2 Neighbor structure-feature based graph pooling (Knowledge filtering)
Top-k node selection
Graph pooling
3 Few-shot learning via adaptive neighbor loss correction (Knowledge correction)
Graph unpooling
$\mathcal{L}_1$
$\mathcal{L}_2$
$\mathcal{L}$
λ
$\mathcal{L}_1$ Classfication loss
$\mathcal{L}_2$ Correction loss

Fig. 2. The overview of FASKE in the 5-way 1-shot task. FSAKE mainly consists of KF and KC. The small rectangular grid next to a node indicates a class representative feature of a class. More rectangles mean more representative features of a class are obtained. The color denotes the importance of a class feature, with darker colors indicating more representative features.

3.2. Framework overview

In this paper, we propose FSAKE for FSL utilizing the structure of Graph U-Net (Gao & Ji, 2022). Fig. 2 depicts the basic model diagram of FSAKE, which contains two major parts: Knowledge filtering and Knowledge correction.

(1) Neighbor structure-feature based graph pooling (Knowledge filtering, KF): We integrate both the graph structure and node feature in the graph pooling phase. Specifically, we aggregate information about the node and its first-order neighbors to objectively evaluate the class knowledge, which obtains more accurate and representative class knowledge in the graph down-sampling phase.

(2) Few-shot learning via adaptive neighbor loss correction (Knowledge correction, KC): We add a correction loss function to the penultimate layer of the graph up-sampling phase since class knowledge may be propagated incorrectly between layers. Supervising the classification results in the penultimate layer minimizes inter-layer incorrect propagation in the last layer by leveraging a skip connection.

3.3. Neighbor feature-structure based graph pooling

In the graph pooling phase, we first initialize the graph network. Then, update the graph network using graph reasoning within the same pooling layer. Next, we evaluate the node influence by combining the neighbor node class information. Finally, we perform graph pooling according to different node selection strategies to further learn the class features of influential nodes.

Let each classification task $\mathcal{T}$ consists of a support set $S = \{(x_i, y_i)\}_{i=1}^{NK}$ and a query set $Q = \{(x_i, y_i)\}_{i=NK+1}^{NK+NH}$, where x_i is the sample and y_i is the corresponding class label. The support set S consists of K samples drawn from N classes (N-way K-shot task), which serve as labeled instances. Similarly, the query set Q is formed by selecting additional H samples from the same N classes, which are the unlabeled samples to be predicted. The data instance in the task constitutes an undirected acyclic graph $G^{(l)} = (\mathbf{V}^{(l)}, \mathbf{E}^{(l)})$, where $0 \le l \le l_{max}$ and l_{max} represents the number of graph pooling layers. In graph $G^{(l)}$, $\mathbf{V}^{(l)} = \{v_1^{(l)}, \dots, v_i^{(l)}, \dots, v_{n^{(l)}}^{(l)}\}$ is a set of nodes, where $n^{(l)}$ denotes the number of nodes in the lth layer. Conveniently, the edge set $\mathbf{E}^{(l)}$ is represented exploiting an adjacency matrix $\mathbf{A}^{(l)} \in \mathbb{R}^{n^{(l)} \times n^{(l)}}$, where $a_{i,j}^{(l)} \in \mathbf{A}^{(l)}$ denotes similarity between the two connected nodes $v_i^{(l)}$ and $v_j^{(l)}$.

Graph network initialization. For a N-way K-shot task $\mathcal{T} = S \cup Q$, all the input samples x_i are put into the feature extractor $f_\phi(\cdot)$ to extract the sample features as Fig. 2. The node $v_i^{(0)}$ is generated by concatenating the label encoding with the feature of a sample x_i from $\mathcal{T}$ as follows:

$$v_i^{(0)} = (f_\phi(x_i), h(y_i)), \tag{4}$$

where $f_\phi(\cdot)$ denotes the feature extractor and $h(\cdot)$ represents the label encoder. For a support sample, its label encoding is represented by a one-hot encoding vector. A uniform distribution vector $\left[\frac{1}{N}, \dots, \frac{1}{N}\right]$ is utilized to represent the query sample. Consequently, the node set is denoted by $\mathbf{V}^{(0)} \in \mathbb{R}^{n^{(0)} \times d}$, where d represents the node feature dimension. Given the support set label, the adjacency matrix $\mathbf{A}^{(0)} = \{a_{i,j}^{(0)}\}_{i,j=1}^{n^{(0)}} \in \mathbb{R}^{n^{(0)} \times n^{(0)}}$ is initialized as follows:

$$a_{i,j}^{(0)} = \begin{cases} 1, & \text{if } y_i = y_j \text{ and } v_i^{(0)}, v_j^{(0)} \in S, \\ 0, & \text{if } y_i \ne y_j \text{ and } v_i^{(0)}, v_j^{(0)} \in S, \\ 0.5, & \text{otherwise.} \end{cases} \tag{5}$$

Graph reasoning. The graph reasoning layer explores the relationships among nodes within the same graph pooling layer. Although GNNs have demonstrated considerable success in FSL, the common challenge of oversmoothing among existing models restricts their expressive ability (Wang et al., 2024). Therefore, we leverage a low-level adjacent matrix $\mathbf{A}^{(l)}$ and a high-level adjacent matrix $\mathbf{A}_{new}^{(l)}$ to aggregate the learned low-level and high-level relationships among nodes. Specifically, we calculate $\mathbf{A}_{new}^{(l)}$ based on the current node feature matrix $\mathbf{V}^{(l)}$ in the lth layer as follows:

$$\mathbf{A}_{new}^{(l)} = \mathcal{M}_{\theta_1}(abs(v_i^{(l)} - v_j^{(l)})), \quad i, j = 1, \dots, n^{(l)}, \tag{6}$$

where $abs(\cdot)$ represents an absolute difference between the two node features, $\mathcal{M}_{\theta_1}(\cdot)$ represents utilizing the non-linear Multilayer Perceptron (MLP) to update the adjacent matrix. Subsequently, we learn a high-level node feature matrix $\mathbf{V}_{new}^{(l)}$ corresponding to $\mathbf{A}_{new}^{(l)}$ in the lth layer by information passing:

$$\mathbf{V}_{new}^{(l)} = \mathcal{A}\left(\tilde{\mathbf{D}}_{new}^{(-\frac{1}{2})} \tilde{\mathbf{A}}_{new}^{(l)} \tilde{\mathbf{D}}_{new}^{(-\frac{1}{2})} \mathbf{V}^{(l)} \mathbf{W}_{new}^{(l)}\right), \tag{7}$$

where $\mathcal{A}(\cdot)$ denotes an activation function (i.e., ReLU), $\tilde{\mathbf{D}}_{new}^{-\frac{1}{2}}$ represents the degree matrix of $\tilde{\mathbf{A}}_{new}^{(l)}$, $\tilde{\mathbf{A}}_{new}^{(l)} = \mathbf{A}_{new}^{(l)} + \mathbf{I}$ is the adjacency matrix of the graph with added self-loop, and $\mathbf{W}_{new}^{(l)}$ represents a trainable weight matrix responsible for linearly transforming the feature space dimensionality. In this way, we obtain a high-level adjacency matrix $\mathbf{A}_{new}^{(l)}$ and node feature matrix $\mathbf{V}_{new}^{(l)}$ based on $\mathbf{A}^{(l)}$ and $\mathbf{V}^{(l)}$.

The complete process of obtaining a new node feature matrix $\hat{\mathbf{V}}^{(l)}$ by graph reasoning in the lth layer is as follows:

$$\hat{\mathbf{V}}^{(l)} = \mathcal{M}_{\theta_2}\left(\mathcal{C}\left(\mathbf{V}_{new}^{(l)}, \tilde{\mathbf{D}}^{(-\frac{1}{2})}\tilde{\mathbf{A}}^{(l)}\tilde{\mathbf{D}}^{(-\frac{1}{2})}\mathbf{V}^{(l)}\mathbf{W}^{(l)}\right)\right), \tag{8}$$

where operation $\mathcal{C}(\cdot,\cdot)$ denotes concatenating low-level and high-level node feature matrices in the lth layer, $\mathcal{M}_{\theta_2}(\cdot)$ represents utilizing the MLP to adjust the feature space dimensionality.

Neighbor structure-feature based node score computation. Graphs usually contain many nodes and edges, which carry much structural information. Therefore, we leverage GCNConv (Kipf & Welling, 2017) to consider structural information and feature information for evaluating the node importance. Node importance score $s^{(l)}$ in the lth layer is calculated as follows:

$$s^{(l)} = \sigma\left(\tilde{\mathbf{D}}_{new}^{(-\frac{1}{2})}\tilde{\mathbf{A}}_{new}^{(l)}\tilde{\mathbf{D}}_{new}^{(-\frac{1}{2})}\hat{\mathbf{V}}^{(l)}\mathbf{W}_{s}^{(l)}\right), \tag{9}$$

where $\mathbf{W}_{s}^{(l)}$ represents trainable weight matrix for score calculation. Then, transform the output to a score ranging from 0 to 1 through a sigmoid function $\sigma(\cdot)$.

Node selection strategy. We adopt different node selection strategies in 5-way 1-shot and 5-way 5-shot tasks, respectively. Although Chen, Li, et al. (2021) utilized a k-nearest strategy to select the nearest nodes to the centroid is suitable for the 5-way 5-shot task, it is not ideal for the 5-way 1-shot task. The significant score disparity in the 5-way 1-shot task makes computing the mean of scores from 5 nodes per task meaningless. Persisting with this strategy could result in erroneous node selection. On the contrary, we propose employing absolute scores directly as the node selection criterion in the 1-shot task. The process of selecting the reserved indices $idx_S^{(l+1)}$ for the support nodes in the $(l+1)$-th layer is as follows:

$$idx_S^{(l+1)} = \begin{cases} \text{Max}(s^{(l)}, k^{(l+1)}), & \text{if 1-shot,} \\ \text{Nearest}(c^{(l)}, s^{(l)}, k^{(l+1)}), & \text{if 5-shot,} \end{cases} \tag{10}$$

where Max$(\cdot)$ retrieves the indices of the top $k^{(l+1)}$ nodes with the largest absolute score, Nearest$(\cdot)$ obtains the indices of the top $k^{(l+1)}$ nearest nodes to the centroid of the lth layer. In the 5-shot task, $c^{(l)}$ is the mean of the scores of the support nodes for each class in the lth layer.

Graph pooling. We design a learnable graph pooling layer to downsample graphs by combining features and structures of neighbor nodes, thus learning rich class knowledge representations. Specifically, the graph pooling layer retains all query nodes and selectively downsample support nodes. This approach extracts more representative class features and facilitates relational learning among query and support nodes. Then we obtain the adaptive feature embedding suitable for the different tasks utilizing this method.

We obtain the reserved support node indices idx_S through the above operations. Then, we combine the reserved support indices and unchanged query indices to form the indices $idx^{(l+1)}$ representing all nodes in the next layer: $idx^{(l+1)} = idx_S^{(l+1)} \cup idx_Q^{(l)}$, where $idx_Q^{(l)}$ denotes the unchanged indices of the query nodes in lth layer. Next, we multiply the eigenvalues of the retained nodes by their scores to indicate node importance and control information aggregation. We leverage the obtained indices $idx^{(l+1)}$ to derive $\mathbf{A}^{(l+1)}$ and $\mathbf{V}^{(l+1)}$ after applying the graph pooling layer as follows:

$$\begin{cases} \mathbf{A}^{(l+1)} = \mathbf{A}_{idx^{(l+1)}}^{(l)} \\ \mathbf{V}^{(l+1)} = \mathbf{V}_{idx^{(l+1)}}^{(l)} \odot s_{idx^{(l+1)}}^{(l)}, \end{cases} \tag{11}$$

where $\mathbf{A}_{idx^{(l+1)}}^{(l)}$ and $\mathbf{V}_{idx^{(l+1)}}^{(l)}$ are the retained adjacent matrix and node feature matrix in the lth layer, $s_{idx^{(l+1)}}^{(l)}$ refers to the projection score of reserved node in the lth layer, and "$\odot$" is the element-wise product.

3.4. Few-shot learning via adaptive neighbor loss correction

In the graph unpooling phase, we implement an equal number of decoding layers corresponding to the encoding stages. The graph unpolling layer and graph reasoning layer are the two components of each decoder block. Furthermore, we introduce a correction loss to supervise the inter-layer propagation of class knowledge.

Graph Unpooling Layer. Graph unpooling layer performs the upsampling operation that is the opposite of down-sampling. Based on the indices $idx^{(l)}$ of the reserved nodes, we generate the node feature matrix $\mathbf{V}^{(l+1)}$ by placing the pooled nodes back to the original position in the graph as follows:

$$\mathbf{V}^{(l+1)} = \mathcal{D}(\mathbf{0}_{n^{(l+1)}\times d}, \mathbf{V}^{(l)}, idx^{(l)}), \tag{12}$$

where $\mathbf{0}_{n^{(l+1)}\times d}$ represents an initially unpopulated feature matrix of the new graph, $\mathbf{V}^{(l)}$ represents a node feature matrix of the current graph, and $\mathcal{D}(\cdot,\cdot,\cdot)$ serves to populate the row vector of $\mathbf{V}^{(l)}$ to $\mathbf{0}_{n^{(l+1)}\times d}$ according to $idx^{(l)}$. Subsequently, we fuse the features of the same graph structure from the encoder part through skip connections and populate empty feature vectors of $\mathbf{V}^{(l+1)}$ via graph reasoning.

Correction Loss and Classification Loss. We design a knowledge correction strategy to complement the pooling phase, which mitigates the problem of incorrect propagation of class knowledge in the pooling phase. Concretely, we utilize the softmax function in the last two layers to map node features to categories. The classification probability $p_{i,j}$ is the node v_i belongs to the jth class is computed as follows:

$$p_{i,j} = \frac{v_{i,j}}{\sum_{j=1}^{N} v_{i,j}}, \tag{13}$$

where $v_{i,j}$ is the N-dimensional node feature of the unpooling layer after graph reasoning. Finally, select the support category index with the highest probability as the query node category.

Next, we optimize FSAKE leveraging cross entropy loss as follows:

$$\mathcal{L}_{CE} = -\frac{1}{NH}\sum_{i=NK+1}^{NK+NH}\sum_{j=1}^{N} y_{i,j}\log(p_{i,j}), \tag{14}$$

where NH is the number of query samples and N is the number of classes in a task, $y_{i,j} = 1$ if the jth class is the correct classification for sample v_i, otherwise $y_{i,j} = 0$.

Our final loss function combines classification loss and correction loss, both utilizing the standard cross-entropy loss function $\mathcal{L}_{CE}$. Finally, we minimize the total loss function to obtain the best parameter set θ^* as follows:

$$\theta^* = \arg\min_{\theta}\left\{\mathcal{L}_1 + \lambda\mathcal{L}_2\right\}, \tag{15}$$

where $\mathcal{L}_1$ denotes the classification loss in the last layer of the model, $\mathcal{L}_2$ denotes the correction loss in the penultimate layer of the model that is used to monitor the inter-layer propagation of class knowledge, and λ is a weight factor to trade-off the two losses.

3.5. Model analysis and learning

FSAKE leverages neighbor knowledge to obtain an adaptive embedding that learns rich and representative class knowledge. Algorithm 1 illustrates pseudo-code for the FSAKE forward inference process. Line 6 evaluates the node importance based on its local structure and feature information. Line 7 retains the node feature based on the node score. Line 11 restores the original graph structure through unpooling. Line 14 utilizes backpropagation of loss function to update model parameters.

Compared with traditional graph-level FSL methods that exploit graph pooling, FSAKE has the following merits. First, FSAKE leverages

Algorithm 1 FSAKE: Few-shot graph learning via adaptive neighbor class knowledge embedding

```
Input: The training epochs number is $E_{max}$. The training set $D_{\text{base}}$, the number of classes $N$, and the number of support examples $K$ for each class. The number of pooling layers $l_{max}$.
Output: The best parameter set $\theta^*$ of FSAKE.
1: Randomly select samples to build $\mathcal{T} = \mathcal{S} \cup \mathcal{Q}$ ;
2: Initialize $\mathbf{V}^{(0)}$ and $\mathbf{A}^{(0)}$ according to Eq. (4) and Eq. (5);
3: for $epoch = 1 : E_{max}$ do
4:     for $l = 0 : l_{max}$ do
5:         Update $\mathbf{V}^{(l)}$ by graph reasoning as Eq. (8);
6:         Calculate the node importance score $s^{(l)}$ according to Eq. (9);
7:         Select the reserved support node indexes $idx_S$ according to Eq. (10);
8:         Obtain $\mathbf{A}^{(l+1)}$ and $\mathbf{V}^{(l+1)}$ according to Eq. (11);
9:     end for
10:    for $l = 0 : l_{max}$ do
11:        Restore the graph to its original structure according to Eq. (12);
12:    end for
13:    Obtain the $p_{i,j}$ of the last two layers of nodes according to Eq. (13);
14:    Update all parameters according to Eq. (15);
15: end for
16: return $\theta^*$;
```

the node local structure and feature information to evaluate the node importance rather than from the node feature information alone. This method considers the graph structure information in the pooling layer to select influential nodes, enabling the model to learn a more representative and abundant graph-level representation. Second, it reduces the time complexity of subsequent reasoning because some nodes were removed after pooling. Finally, FSAKE degenerates to the traditional node-level FSL based on GNNs if the number of pooled nodes per layer remains constant. Therefore, FSAKE can extend traditional node-level FSL based on GNNs.

4. Experiments

This section begins with an introduction to the datasets and experimental setups. Subsequently, we present the results achieved by FSAKE across the datasets and compare them with the related methods. We then analyze the different node selection strategies and conduct ablation experiments. Finally, we perform a visualization experiment to demonstrate the effectiveness of FSAKE.

4.1. Datasets and experimental settings

Datasets. We perform few-shot classification experiments on four commonly used FSL benchmark datasets to showcase the effectiveness of FSAKE. These four datasets contain miniImageNet (Vinyals et al., 2016), tieredImageNet (Ren et al., 2018), CIFAR-FS (Bertinetto et al., 2019), and CUB-200-2011 (Wah et al., 2011). We follow the standard practice of splitting each dataset into training, validation, and test datasets like the previous method (Liu et al., 2024). Table 1 presents the fundamental data statistics of the four datasets.

Implementation details. We leverage a 4CONV as the feature extractor $f_\phi(\cdot)$, which is widely adopted in the compared GNNs based methods (Chen, Li, et al., 2021; Satorras & Estrach, 2018; Zhao, Su, et al., 2024). Finally, we obtain a 128-dimensional feature representation to construct the initial graph network. During the training phase, we utilize the Adam optimizer and set the initial learning rate and weight decay to 10^{-3} and 10^{-6}, respectively. Additionally, we pool uniformly twice in a 1-shot/5-shot task. We randomly select 500 tasks from the novel classes during the test phase for few-shot classification. The mean accuracy is calculated with a 95% confidence interval. Experiments are conducted on hardware with a GeForce RTX 2080 Ti GPU utilizing Pytorch.

Table 1
Dataset descriptions.

Datasets	Images	Classes	Train-val-test	Resolution
miniImageNet	60,000	100	64/16/20	84 × 84
tieredImageNet	779,165	608	351/97/160	84 × 84
CIFAR-FS	60,000	100	64/16/20	32 × 32
CUB-200–2011	11,788	200	100/50/50	84 × 84

Table 2
The mean accuracy (%) comparison of different few-shot classification models on miniImageNet, with 95% confidence intervals. Most reported results come from the original paper. * Denotes that the results are reimplemented from publicly available code.

Model	GNNs	5-way 1-shot	5-way 5-shot
HTS (Zhang, Huang, et al., 2022)	N	58.96 ± 0.18	75.17 ± 0.14
SAPENet (Huang & Choi, 2023)	N	55.71 ± 0.20	71.81 ± 0.16
HyperShot (Sendera et al., 2023)	N	53.18 ± 0.45	69.62 ± 0.20
SSL-ProtoNet (Lim et al., 2024)	N	52.58 ± 0.45	70.87 ± 0.36
GNN (Satorras & Estrach, 2018)	Y	54.72 ± 0.70	75.41 ± 0.47
TPN (Liu et al., 2019)	Y	53.75 ± 0.87	69.43 ± 0.78
EGNN (Kim et al., 2019)	Y	58.98 ± 0.36	76.37 ± 0.30
EPNet (Rodríguez et al., 2020)	Y	59.32 ± 0.88	72.95 ± 0.64
TRPN (Ma et al., 2020)	Y	57.84 ± 0.51	78.57 ± 0.44
SPN (Koniusz & Zhang, 2022)	Y	55.36 ± 0.70	71.23 ± 0.60
HybridGNN (Koniusz & Zhang, 2022)	Y	55.63 ± 0.20	72.48 ± 0.16
HGNN* (Chen, Li, et al., 2021)	Y	60.91 ± 0.73	77.54 ± 0.66
FSAKE (ours)	Y	**61.86 ± 0.72**	**79.66 ± 0.62**

Baselines. In our experiments, we compared our proposed method with several related FSL methods to comprehensively evaluate its performance. These methods can be broadly categorized into two groups: non-graph-based methods and graph-based methods. (1) Non-graph-based methods: ProtoNet (Snell et al., 2017), RelationNet (Sung et al., 2018), HTS (Zhang, Huang, et al., 2022), SAPENet (Huang & Choi, 2023), HyperShot (Sendera et al., 2023), SSL-ProtoNet (Lim et al., 2024). These methods primarily rely on distance metrics or meta-learning strategies to enhance the representation and discriminative power of the model. (2) Graph-based methods: GNN (Satorras & Estrach, 2018), TPN (Liu et al., 2019), EGNN (Kim et al., 2019), EPNet (Rodríguez et al., 2020), TRPN (Ma et al., 2020), TRPN-D (Ma et al., 2022), SPN (Koniusz & Zhang, 2022), HybridGNN (Koniusz & Zhang, 2022), HGNN (Chen, Li, et al., 2021), AGNN (Cheng et al., 2023), CSTS (Zhao, Su, et al., 2024). These methods generally explore the relationship between support and query sets by modeling the relationship between samples as a graph structure.

4.2. Comparisons with related methods

We assess the effectiveness of FSAKE by comparing it with the graph and non-graph (Y/N) models on four benchmarks. HGNN (Chen, Li, et al., 2021) serves as our baseline and is mostly relevant to our approach. The best results are highlighted in bold.

The experimental results on miniImageNet are illustrated in Table 2. We obtain the following observations by comparing FSAKE with other FSL models:

(1) We observe that many graph models outperform non-graph models with the same utilization of the shallow feature extractor 4CONV. For instance, HybridGNN obtains 2.45% and 2.86% better performance than the non-graph method HyperShot in 1-shot and 5-shot tasks, respectively. FSAKE obtains 8.68% and 10.04% better performance than the non-graph method HyperShot in 1-shot and 5-shot tasks, respectively. The potential reason why graph models always outperform non-graph models when utilizing shallow feature extractors may be the graph inference process optimizes embedding.

Table 3
The mean accuracy (%) comparison of different few-shot classification models on CUB-200–2011, with 95% confidence intervals. Most reported results come from the original paper. * Denotes that the results are reimplemented from publicly available code.

Model	GNNs	5-way 1-shot	5-way 5-shot
HTS (Zhang, Huang, et al., 2022)	N	67.32 ± 0.24	78.09 ± 0.15
HyperShot (Sendera et al., 2023)	N	66.13 ± 0.26	80.07 ± 0.22
SAPENet (Huang & Choi, 2023)	N	70.38 ± 0.23	84.47 ± 0.14
GNN (Satorras & Estrach, 2018)	Y	68.56 ± 0.85	83.12 ± 0.47
EGNN (Kim et al., 2019)	Y	67.25 ± 0.42	82.15 ± 0.30
HybridGNN (Koniusz & Zhang, 2022)	Y	69.02 ± 0.22	83.20 ± 0.15
AGNN (Cheng et al., 2023)	Y	75.81	88.22
CSTS (Zhao, Su, et al., 2024)	Y	60.83 ± 0.45	77.12 ± 0.44
HGNN* (Chen, Li, et al., 2021)	Y	75.64 ± 0.69	89.19 ± 0.49
FSAKE (ours)	Y	**77.00** ± **0.70**	**89.66** ± **0.50**

Table 4
The mean accuracy (%) comparison of different few-shot classification models on tieredImageNet, with 95% confidence intervals. Most reported results come from the original paper. * Denotes that the results are reimplemented from publicly available code.

Model	GNNs	5-way 1-shot	5-way 5-shot
HTS (Zhang, Huang, et al., 2022)	N	53.20 ± 0.22	72.38 ± 0.19
SAPENet (Huang & Choi, 2023)	N	57.61 ± 0.22	75.42 ± 0.18
SSL-ProtoNet (Lim et al., 2024)	N	55.14 ± 0.49	74.23 ± 0.40
GNN (Satorras & Estrach, 2018)	Y	64.56 ± 0.87	81.89 ± 0.47
TPN (Liu et al., 2019)	Y	57.53 ± 0.96	72.85 ± 0.74
EGNN (Kim et al., 2019)	Y	63.25 ± 0.42	80.15 ± 0.30
EPNet (Rodríguez et al., 2020)	Y	59.97 ± 0.95	73.91 ± 0.75
TRPN (Ma et al., 2020)	Y	59.26 ± 0.50	79.66 ± 0.45
HybridGNN (Koniusz & Zhang, 2022)	Y	56.05 ± 0.21	72.82 ± 0.18
TRPN-D (Ma et al., 2022)	Y	61.01 ± 0.49	80.98 ± 0.42
CSTS (Zhao, Su, et al., 2024)	Y	64.84 ± 0.26	82.95 ± 0.44
HGNN* (Chen, Li, et al., 2021)	Y	63.95 ± 0.74	82.84 ± 0.61
FSAKE (ours)	Y	**65.27** ± **0.73**	**83.33** ± **0.62**

(2) Compared with GNN, TPN, and EGNN methods that do not utilize class knowledge, FSAKE achieves better improvement on the miniImageNet dataset. For example, FSAKE increases by 7.14%, 8.11%, and 2.88% than GNN, TPN, and EGNN methods in the 1-shot task, respectively. Similarly, FSAKE improves by 4.25%, 10.23%, and 3.29% than GNN, TPN, and EGNN methods in the 5-shot task, respectively. It demonstrates FSAKE utilizes class knowledge to enrich the feature expression and facilitates few-shot classifications.

(3) The FSAKE model is superior to the baseline model HGNN in 1-shot and 5-shot tasks by 0.95% and 2.12%, respectively. It suggests that FSAKE obtains more representative class knowledge by aggregating and exploiting information from neighboring nodes.

Table 3 illustrates the experimental results on the more challenging fine-grained CUB-200-2011 dataset. The following observations are obtained:

(1) We observe that even on the more challenging few-shot fine-grained classification task, our proposed FSAKE method is still effective. For instance, the accuracy of FSAKE is 7.98% and 16.17% higher than the HybridGNN and CSTS model in the 1-shot task. It explains that even for fine-grained classification tasks, FSAKE exploits relationships among fine-grained samples to mine rich class knowledge for sample inference.

(2) Another observation is that FSAKE outperforms other graph models by exploiting class knowledge. For example, the performance is improved from 75.64% to 77.00% and from 89.19% to 89.66% in 1-shot and 5-shot tasks, respectively. This confirms exploiting the graph topological information in the pooling layer improves the results.

Table 4 presents the main classification results on tieredImageNet. On the tieredImageNet dataset with a category hierarchy, we obtain the following observation:

(1) FSAKE achieves better performance compared with graph and non-graph methods that utilize a hierarchical structure. For instance, FSAKE outperforms CSTS and HTS by 0.43% and 12.07% in the 1-shot task, respectively. This suggests that even though FSAKE does not explicitly utilize the hierarchy among categories, the hierarchical pooling approach effectively explores the hierarchical structure among samples.

Table 5
The mean accuracy (%) comparison of different few-shot classification models on CIFAR-FS, with 95% confidence intervals. Most reported results come from the original paper. * Denotes that the results are reimplemented from publicly available code.

Model	GNNs	5-way 1-shot	5-way 5-shot
ProtoNet (Snell et al., 2017)	N	55.50 ± 0.70	72.00 ± 0.60
RelationNet (Sung et al., 2018)	N	55.00 ± 1.00	69.30 ± 0.80
HTS (Zhang, Huang, et al., 2022)	N	64.71 ± 0.21	76.45 ± 0.17
SSL-ProtoNet (Lim et al., 2024)	N	60.41 ± 0.52	76.52 ± 0.38
CSTS (Zhao, Su, et al., 2024)	Y	62.47 ± 0.47	81.82 ± 0.42
HGNN* (Chen, Li, et al., 2021)	Y	67.67 ± 0.73	**86.16** ± **0.56**
FSAKE (ours)	Y	**69.78** ± **0.69**	85.92 ± 0.55

(2) FSAKE demonstrates excellent performance on tieredImageNet, which is a large dataset containing significantly more data. For example, FSAKE is superior to the suboptimal method CSTS by 0.43% and 0.38% in 5-way 1-shot and 5-shot tasks, respectively. This means that FSAKE is an effective technique for improving few-shot classification.

(3) Compared with the traditional pooling method HGNN, FSAKE outperforms by 1.32% and 0.49% in 5-way 1-shot and 5-shot tasks, respectively. This proves that our method obtains a more efficient pooling effect using the neighborhood class information.

Table 5 presents the main classification performance on CIFAR-FS. We obtain the following observations by comparing with other FSL models:

(1) We observe that FSAKE degrades performance over HGNN by 0.24% in the 5-way 5-shot task while outperforming HGNN by 2.11% in the 1-shot task. The potential reason why FSAKE does not always outperform HGNN on CIFAR-FS may be that CIFAR-FS has a lower pixel resolution than the other datasets. We speculate more low-resolution samples may bring false neighbor information in the 5shot task, leading to a decrease in classification performance.

(2) FSAKE still outperforms other methods in the 5-way 1-shot setting, demonstrating the effectiveness of FSAKE under extreme data scarcity. For example, the accuracy of FSAKE is better than SCTS and SSL-ProtoNet by 7.31% and 9.37% in the 5-way 1-shot task, respectively.

4.3. Analysis of different node selection strategies

To illustrate the impacts of utilizing different node selection strategies in FSAKE, we perform experiments on miniImageNet. As indicated in Fig. 3, "Nearest" represents the nearest neighbor strategy for selecting the node closest to the centroid, and "Max" represents the maximum value strategy for directly selecting the node with the highest score. We observe the following results:

(1) Fig. 3(a) indicates that utilizing the maximum value strategy is more suitable for selecting nodes under the 5-way 1-shot task in the FSAKE because there is a significant difference in scores for each node in few-shot scenarios with only one sample per class. The nearest neighbor strategy requires calculating the distance to the centroid of the score, which is just as susceptible to extreme values as the computation of the mean prototype. On the contrary, directly selecting the top k nodes with the highest score to retain is more effective and simpler. This approach selects samples from different categories similar to each other to further learn discriminative knowledge.

(2) The nearest neighbor strategy performed better than the maximum value strategy under the 5-way 5-shot task in FSAKE, as shown in Fig. 3(b). This confirms that the nearest neighbor strategy better exploits the interrelationships among nodes of the same class when the number of samples increases. Selecting representative nodes in each class helps the model acquire representative class knowledge.

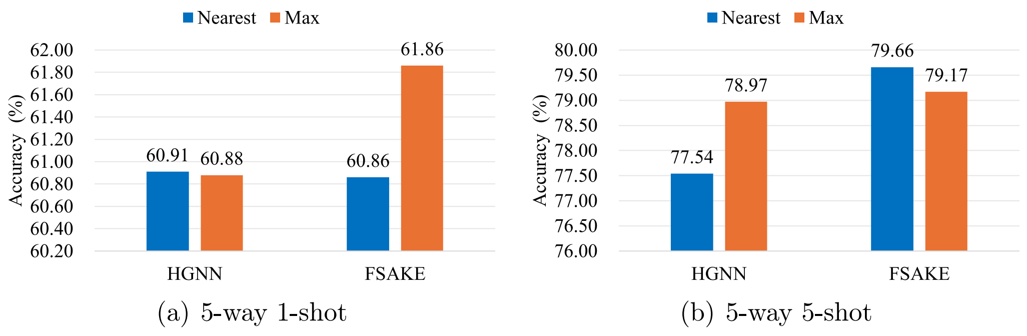

Fig. 3. Evaluation of different node selection strategies on miniImageNet. "Nearest" denotes the nearest neighbor strategy, and "Max" denotes the maximum value strategy.

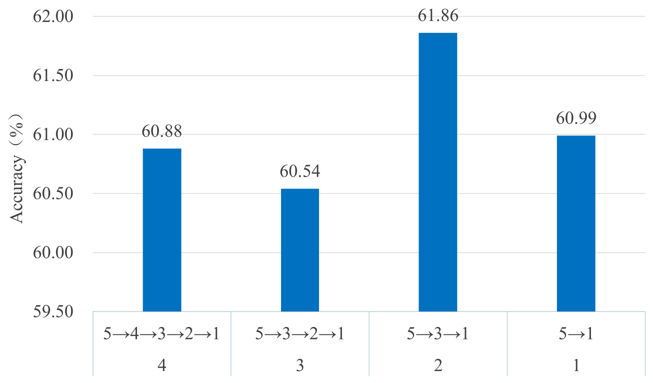

Fig. 4. Performance comparison of different pooling layers under 1-shot setting on miniImageNet. n→m means selecting the most important m from n samples.

(3) Compared with the baseline model HGNN, the improvement of FSAKE is more significant in the 5-way 1-shot task with a maximize value strategy. For instance, the accuracy of the HGNN with the maximum strategy decreases from 60.91% to 60.88%, while FSAKE improves from 60.86% to 61.86%. It is because the FSAKE model identifies important nodes by combining information from neighboring nodes, which provides a more objective assessment of node importance. This is particularly significant in cases of sample scarcity. In contrast, HGNN evaluating node importance alone may result in more outlier scores, leading to the failure of the maximum value strategy.

Overall, the FSAKE model utilizes the maximum value strategy to make the model focus on the most informative nodes when dealing with a limited sample size. In contrast, the nearest neighbor strategy utilizes the relationships among nodes better to enhance classification accuracy when increasing the number of samples.

4.4. Analysis of different pooling layers

To explore the performance of FSAKE utilizing different pooling layers, we conduct experiments on miniImageNet in a 1-shot setting. As shown in Fig. 4, we compare the impact of four different numbers of pooling layers on model performance. We draw the following conclusions from the results.

(1) Adding more pooling layers does not necessarily improve model performance. For example, four or three pooling layers are less effective than two and one. We speculate that this may be due to many pooling layers making the model too complex to optimize. This results in over-fitting for certain categories, leading to a decrease in model performance.

(2) The model tends to perform worse with one pooling layer than with two. This indicates that too few pooling layers may prevent the model from effectively capturing the hierarchical structure of the data, leading to reduced performance.

Table 6
Ablation study results on miniImageNet and CUB-200–2011.

KF	KC	miniImageNet		CUB-200–2011	
		1-shot	5-shot	1-shot	5-shot
–	–	60.91	77.54	75.64	89.19
✓	–	61.36	79.10	76.24	89.46
–	✓	61.07	78.72	76.78	89.53
✓	✓	**61.86**	**79.66**	**77.00**	**89.66**

(3) The experimental results indicate that the model achieves optimal performance when utilizing two pooling layers. In this setting, the model can capture essential data features and learn a higher level of abstract representation, which enhances classification accuracy.

4.5. Ablation study

To validate the effectiveness of various modules within FSAKE, we conduct ablation studies on miniImageNet and CUB-200-2011. Table 6 illustrates the performance of diverse component configurations in the 5-way 1-shot and 5-way 5-shot classification tasks. We draw the following conclusions from the results:

(1) KF is essential for FSAKE, as it integrates neighbor information in the pooling layer to comprehensively evaluate node importance. By capturing the interdependence and structural information among nodes, the shared common features among nodes are preserved. This further enables the model to extract representative class knowledge from samples and improve classification performance.

(2) KC plays a role in improving model performance, which introduces additional supervised signals into the deep feature representation of the model. For example, KC improves classification accuracy by 0.16% and 1.18% over the miniImageNet baseline in 5-way 1-shot and 5-shot tasks, respectively. It demonstrates that learning intermediate layer features improves the model understanding and class feature representation ability.

(3) We observe that FSAKE achieves better results than the baseline on miniImageNet and CUB-200-2011 with 61.86%/79.66% and 77.00%/89.66% under 1-shot/5-shot tasks, outperforming the baseline by 0.95%/2.12% and 1.36%/0.47%. This verifies that the simultaneous application of the KF and KC effectively improves the FSL classification performance.

(4) The ablation experiments on two different granularity datasets demonstrate the proposed module has generalization ability. For example, KF and KC improve the model performance from 60.91% to 61.86% on miniImageNet under a 1-shot task and from 75.64% to 77.00% on the more challenging fine-grained dataset with the same settings.

4.6. Parameter sensitive analysis

The hyperparameter λ with a range of 0 to 1 corresponds to the classification loss and correction loss trade-offs. Fig. 5 illustrates the

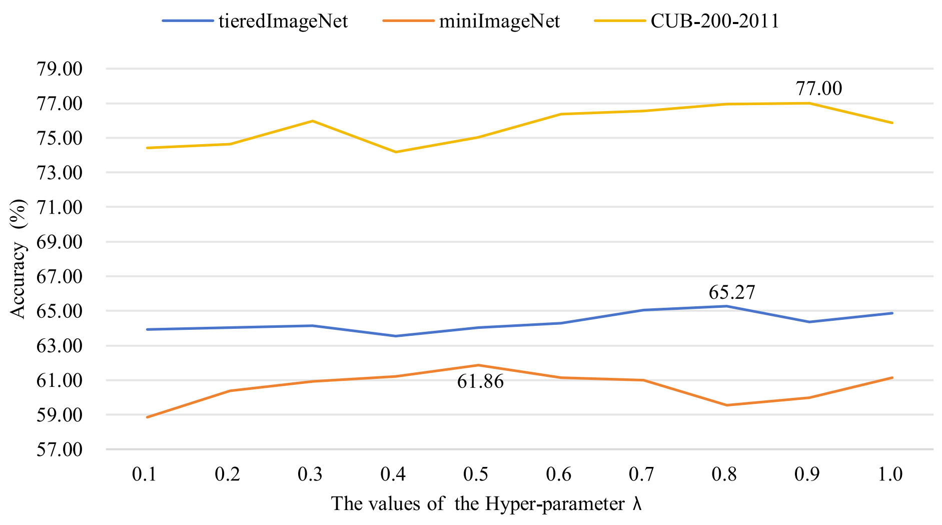

Fig. 5. Test accuracy (%) of the different values of parameter λ under 5-way 1-shot task on tieredImageNet, miniImageNet, and CUB-200-2011.

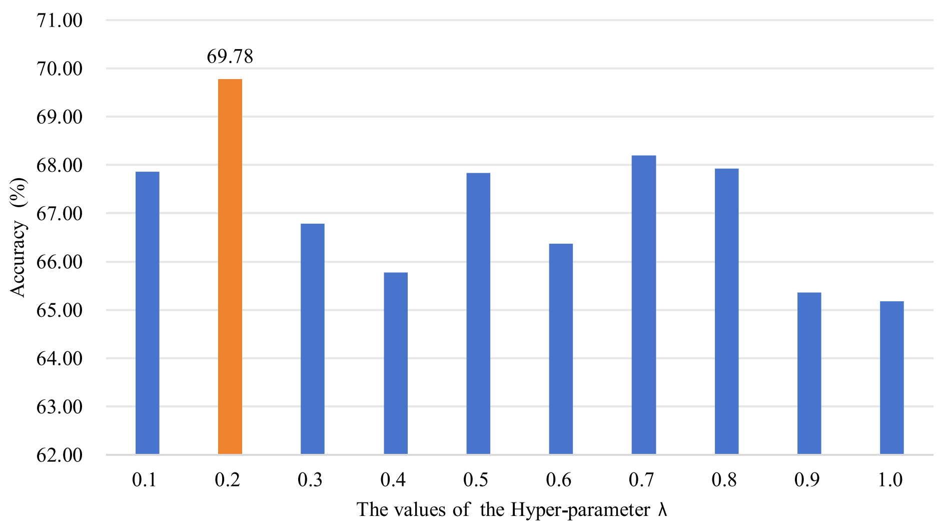

Fig. 6. Test accuracy (%) of the different values of parameter λ under 5-way 1-shot task on CIFAR-FS.

impact of the various parameter values on the accuracy of image classification on three datasets under the 5-way1-shot task. It demonstrates the FSAKE method is sensitive to changes in λ.

As shown in Fig. 6, the phenomenon of FSAKE sensitive to parameter λ is particularly evident on CIFAR-FS. Therefore, different datasets have different parameter settings. The best results are observed at λ = 0.5, 0.8, 0.9, and 0.2 on miniImageNet, tieredImageNet, CUB-200-2011, and CIFAR-FS, respectively. Additionally, the model performance tends to decrease when λ exceeds an optimal threshold. We speculate that this is due to larger parameter values leading to strict deep supervision of the model. Excessive correction leads to a decrease in classification performance.

4.7. Visualization

We perform a t-SNE visualization for sample embedding of the pooling and unpooling phase, randomly sampling a new task from the test set of miniImagenet. Figs. 7 and 8 display the embedding visualization results of the main layers in the FSAKE model.

Fig. 7 reveals the correct dissemination process of class knowledge, indicating FSAKE obtains significant and representative class knowledge. For instance, the embedding after two pooling layers is visualized in Figs. 7(b) and 7(c), from which we observe that correct nodes are retained in intra-class and inter-class. Fig. 7(b) demonstrates that the same support class is close to the query class, illustrating that intra-class pooling preserves intra-class representative samples to learn shared common features. Fig. 7(c) illustrates that classes similar to each other are retained, promoting discriminative class knowledge learning. These results further prove that combining neighbor node information to evaluate class knowledge promotes the learning of class knowledge.

Additionally, considering the potential problem of inter-layer incorrect propagation in class knowledge, we introduce KC to supervise its correction. This phenomenon of incorrect learning between layers is shown in Fig. 8. It is observable from Figs. 8(b) and 8(c) that some retained intra-class nodes are far away from similar query classes, and classes similar to each other are not retained in inter-class. This phenomenon suggests that nodes are incorrectly retained, leading to incorrect propagation of class knowledge. Although this phenomenon is mitigated slightly by combining instance knowledge during the unpolling process from Fig. 8(d), it still leads to a misclassification problem for a query class from Fig. 8(e). As shown in Fig. 8(f), KC effectively mitigates misclassification due to the mislearning of class knowledge during pooling. As displayed in Figs. 7(e) and 7(f), KC results in tighter class clustering and clear inter-class boundaries even if the phenomenon of incorrect inter-layer class knowledge propagation does not occur.

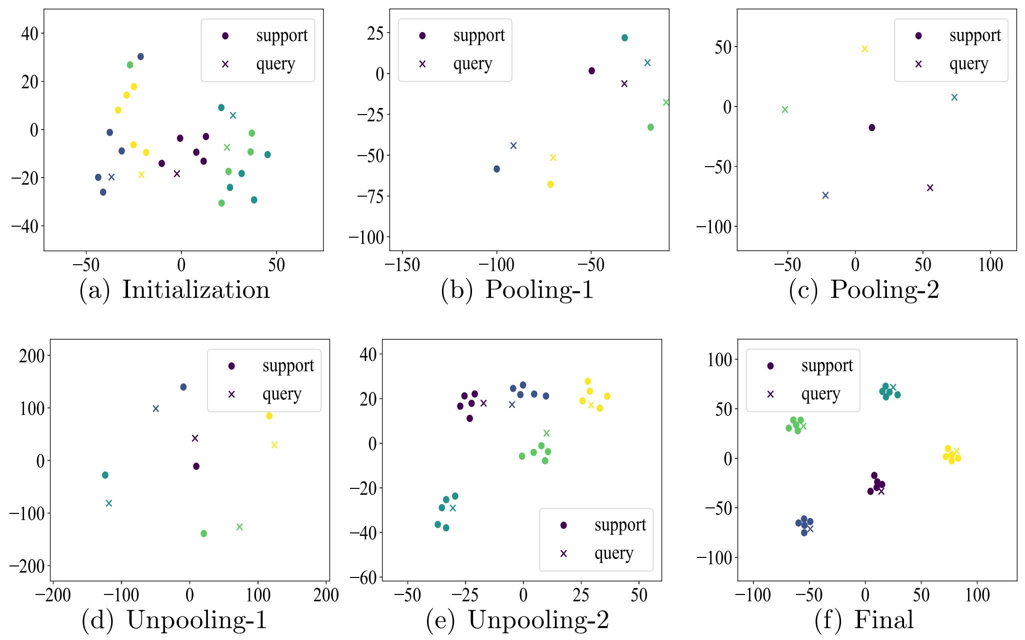

Fig. 7. The t-SNE visualization of correct pooling process for sample embeddings of miniImageNet under the 5-way 5-shot setting. Different colors represent different categories.

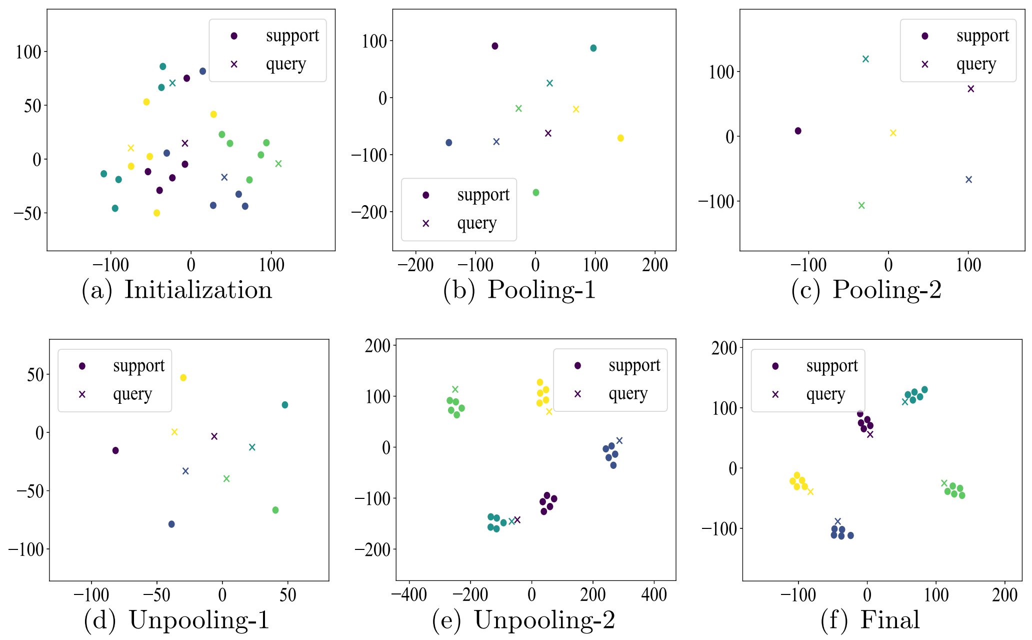

Fig. 8. The t-SNE visualization of error pooling process for sample embeddings of miniImageNet under the 5-way 5-shot setting. Different colors represent different categories.

In conclusion, the experimental results demonstrate the effectiveness of the FSAKE method. Although FSAKE is effective in mitigating the potential incorrect propagation of class knowledge during pooling by adding a correction loss, it was only applied in the penultimate pooling layer. It still needs improvement to achieve more detailed multi-level supervision.

5. Conclusions and future work

In this paper, we propose an adaptive neighbor class knowledge embedding framework for few-shot graph learning (FSAKE). FSAKE combines the knowledge filtering strategy (KF) with the correction strategy (KC) to obtain representative class knowledge, which guides inference about query nodes on the graph in the case of limited samples. KF incorporates neighbor information into the node scoring process, ensuring that representative samples are retained to further mine class knowledge. KC leverages correction loss and classification loss to deeply supervise the propagation of class knowledge. Extensive experiments are carried out on four standard few-shot classification benchmark datasets to validate the effectiveness of FSAKE.

In summary, while FSAKE has shown great potential, its performance depends on the learning of the correct class knowledge during the pooling process. Although we add a correction loss to correct for the incorrect propagation of class knowledge, the effectiveness of this approach may be compromised when faced with highly homogeneous available data. In the future, we will emulate the multi-level supervision strategy by utilizing the correction loss across multiple pooling layers, rather than limiting it to just the penultimate pooling layer. By implementing this improvement, we aim to mitigate existing challenges and enhance the adaptability of our approach to more complex data in future work.

CRediT authorship contribution statement

Linhua Zou: Conceptualization, Methodology, Software, Writing – original draft. **Jie Jin:** Investigation, Validation, Methodology. **Dongqing Li:** Investigation, Validation. **Hong Zhao:** Conceptualization, Methodology, Validation, Writing – review & editing, Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work was supported by the National Natural Science Foundation of China under Grant No. 62376114 and the Fujian Province Undergraduate Universities Teaching Reform Research Project under Grant No. FBJY20240103.

Data availability

Data will be made available on request.

References

Bertinetto, L., Henriques, J. F., Torr, P., & Vedaldi, A. (2019). Meta-learning with differentiable closed-form solvers. In *International conference on learning representations* (pp. 1–15).

Chen, C., Li, K., Wei, W., Zhou, J. T., & Zeng, Z. (2021). Hierarchical graph neural networks for few-shot learning. *IEEE Transactions on Circuits and Systems for Video Technology*, *32*(1), 240–252.

Chen, C., Yang, X., Xu, C., Huang, X., & Ma, Z. (2021). ECKPN: Explicit class knowledge propagation network for transductive few-shot learning. In *IEEE conference on computer vision and pattern recognition* (pp. 6596–6605).

Cheng, H., Zhou, J. T., Tay, W. P., & Wen, B. (2023). Graph neural networks with triple attention for few-shot learning. *IEEE Transactions on Multimedia*, *25*, 8225–8239.

Finn, C., Abbeel, P., & Levine, S. (2017). Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning* (pp. 1126–1135).

Gao, H., & Ji, S. (2022). Graph U-Nets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *44*(9), 4948–4960.

Huang, X., & Choi, S. H. (2023). SAPENet: Self-attention based prototype enhancement network for few-shot learning. *Pattern Recognition*, *135*, Article 109170.

Hui, B., Zhu, P., Hu, Q., & Wang, Q. (2019). Self-attention relation network for few-shot learning. In *IEEE international conference on multimedia and expo* (pp. 198–203).

Jin, J., Zhong, Y., & Zhao, H. (2024). HMRM: Hierarchy-aware misclassification risk minimization for few-shot learning. *Expert Systems with Applications*, *251*, Article 123885.

Kang, D., Kwon, H., Min, J., & Cho, M. (2021). Relational embedding for few-shot classification. In *IEEE/CVF international conference on computer vision* (pp. 8822–8833).

Kim, J., Kim, T., Kim, S., & Yoo, C. D. (2019). Edge-labeling graph neural network for few-shot learning. In *IEEE conference on computer vision and pattern recognition* (pp. 11–20).

Kipf, T. N., & Welling, M. (2017). Semi-supervised classification with graph convolutional networks. In *International conference on learning representations* (pp. 1–14).

Koniusz, P., & Zhang, H. (2022). Power normalizations in fine-grained image, few-shot image and graph classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *44*(2), 591–609.

Li, W., Wang, L., Xu, J., Huo, J., Gao, Y., & Luo, J. (2019). Revisiting local descriptor based image-to-class measure for few-shot learning. In *IEEE conference on computer vision and pattern recognition* (pp. 7260–7268).

Li, W., Wang, Z., Yang, X., Dong, C., Tian, P., Qin, T., Huo, J., Shi, Y., Wang, L., Gao, Y., & Luo, J. (2023). LibFewShot: A comprehensive library for few-shot learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *45*(12), 14938–14955.

Lim, J. Y., Lim, K. M., Lee, C. P., & Tan, Y. X. (2024). SSL-ProtoNet: Self-supervised learning prototypical networks for few-shot learning. *Expert Systems with Applications*, *238*, Article 122173.

Liu, Y., Lee, J., Park, M., Kim, S., Yang, E., Hwang, S. J., & Yang, Y. (2019). Learning to propagate labels: Transductive propagation network for few-shot learning. In *International conference on learning representations* (pp. 1–14).

Liu, F., Yang, S., Chen, D., Huang, H., & Zhou, J. (2024). Few-shot classification guided by generalization error bound. *Pattern Recognition*, *145*, Article 109904.

Lu, J., Gong, P., Ye, J., Zhang, J., & Zhang, C. (2023). A survey on machine learning from few samples. *Pattern Recognition*, *139*, Article 109480.

Ma, Y., Bai, S., An, S., Liu, W., Liu, A., Zhen, X., & Liu, X. (2020). Transductive relation-propagation network for few-shot learning. In *International joint conferences on artificial intelligence* (pp. 804–810).

Ma, Y., Bai, S., Liu, W., Wang, S., Yu, Y., Bai, X., Liu, X., & Wang, M. (2022). Transductive relation-propagation with decoupling training for few-shot learning. *IEEE Transactions on Neural Networks and Learning Systems*, 6652–6664.

Raghu, A., Raghu, M., Bengio, S., & Vinyals, O. (2020). Rapid learning or feature reuse? Towards understanding the effectiveness of MAML. In *International conference on learning representations* (pp. 1–21).

Ren, M., Triantafillou, E., Ravi, S., Snell, J., Swersky, K., Tenenbaum, J. B., Larochelle, H., & Zemel, R. S. (2018). Meta-learning for semi-supervised few-shot classification. In *International conference on learning representations* (pp. 1–15).

Rodríguez, P., Laradji, I., Drouin, A., & Lacoste, A. (2020). Embedding propagation: Smoother manifold for few-shot classification. In *European conference on computer vision* (pp. 121–138).

Rusu, A. A., Rao, D., Sygnowski, J., Vinyals, O., Pascanu, R., Osindero, S., & Hadsell, R. (2019). Meta-learning with latent embedding optimization. In *International conference on learning representations* (pp. 1–17).

Satorras, V. G., & Estrach, J. B. (2018). Few-shot learning with graph neural networks. In *International conference on learning representations* (pp. 1–13).

Sendera, M., Przewięźlikowski, M., Karanowski, K., Zięba, M., Tabor, J., & Spurek, P. a. (2023). HyperShot: Few-shot learning by kernel hypernetworks. In *IEEE/CVF winter conference on applications of computer vision* (pp. 2469–2478).

Shi, B., Li, W., Huo, J., Zhu, P., Wang, L., & Gao, Y. (2023). Global-and local-aware feature augmentation with semantic orthogonality for few-shot image classification. *Pattern Recognition*, *142*, Article 109702.

Snell, J., Swersky, K., & Zemel, R. (2017). Prototypical networks for few-shot learning. In *Advances in neural information processing systems* (pp. 4080–4090).

Sung, F., Yang, Y., Zhang, L., Xiang, T., Torr, P. H., & Hospedales, T. M. (2018). Learning to compare: Relation network for few-shot learning. In *IEEE conference on computer vision and pattern recognition* (pp. 1199–1208).

Tang, S., Chen, D., Bai, L., Liu, K., Ge, Y., & Ouyang, W. (2021). Mutual CRF-GNN for few-shot learning. In *IEEE conference on computer vision and pattern recognition* (pp. 2329–2339).

Vinyals, O., Blundell, C., Lillicrap, T., Kavukcuoglu, K., & Wierstra, D. (2016). Matching networks for one shot learning. In *Advances in neural information processing systems* (pp. 3637–3645).

Wah, C., Branson, S., Welinder, P., Perona, P., & Belongie, S. (2011). The caltech-ucsd birds-200–2011 dataset. California Institute of Technology.

Wang, J., Liang, J., Liang, J., & Yao, K. (2024). GUIDE: Training deep graph neural networks via guided dropout over edges. *IEEE Transactions on Neural Networks and Learning Systems*, 4465–4477.

Xie, J., Long, F., Lv, J., Wang, Q., & Li, P. (2022). Joint distribution matters: Deep brownian distance covariance for few-shot classification. In *IEEE conference on computer vision and pattern recognition* (pp. 7972–7981).

Xin, Z., Chen, S., Wu, T., Shao, Y., Ding, W., & You, X. (2024). Few-shot object detection: Research advances and challenges. *Information Fusion*, *107*, Article 102307.

Yan, E., Zhang, T., Yu, J., Hao, T., & Chen, Q. (2024). A preliminary study on few-shot knowledge reasoning mechanism based on three-way partial order structure. *Information Sciences*, Article 120366.

Yang, J., Dong, Y., & Qian, J. (2023). Research progress of few-shot learning methods based on graph neural networks. *Journal of Computer Research and Development*, 1–26.

Yang, L., Li, L., Zhang, Z., Zhou, X., Zhou, E., & Liu, Y. (2020). DPGN: Distribution propagation graph network for few-shot learning. In *IEEE conference on computer vision and pattern recognition* (pp. 13390–13399).

Yu, T., He, S., Song, Y., & Xiang, T. (2022). Hybrid graph neural networks for few-shot learning. In *AAAI conference on artificial intelligence* (pp. 3179–3187).

Zhang, C., Cai, Y., Lin, G., & Shen, C. (2020). DeepEMD: Few-shot image classification with differentiable earth mover's distance and structured classifiers. In *IEEE conference on computer vision and pattern recognition* (pp. 12203–12213).

Zhang, M., Huang, S., Li, W., & Wang, D. (2022). Tree structure-aware few-shot image classification via hierarchical aggregation. In *European conference on computer vision* (pp. 453–470).

Zhang, X., Zhang, Y., Zhang, Z., & Liu, J. (2022). Discriminative learning of imaginary data for few-shot classification. *Neurocomputing*, *467*, 406–417.

Zhao, F., Huang, T., & Wang, D. (2024). Graph few-shot learning via restructuring task graph. *IEEE Transactions on Neural Networks and Learning Systems*, 1415–1422.

Zhao, H., Su, Y., Wu, Z., & Ding, W. (2024). CSTS: Exploring class-specific and task-shared embedding representation for few-shot learning. *IEEE Transactions on Neural Networks and Learning Systems*, 1–13.

Zheng, P., Guo, X., Chen, E., Qi, L., & Guan, L. (2024). Edge-labeling based modified gated graph network for few-shot learning. *Pattern Recognition*, *150*, Article 110264.

Zhu, P., Zhu, Z., Wang, Y., Zhang, J., & Zhao, S. (2022). Multi-granularity episodic contrastive learning for few-shot learning. *Pattern Recognition*, *131*, Article 108820.