FCGNN: Fuzzy Cognitive Graph Neural Networks with Concept Evolution for Few-Shot Learning

Linhua Zou, Dongqing Li, Chengxi Jiang, *Member, IEEE*, Yu Wang, *Senior Member, IEEE*, Hong Zhao

***Abstract*—Graph neural networks (GNNs) have recently emerged as a promising approach for solving few-shot learning (FSL) problems, enabling generalization to new categories by establishing associations among limited samples. However, most previous methods focus on local interactions within a layer or between successive layers, overlooking the continuous evolution of node states across deeper layers. Especially in FSL scenarios with scarce data, this oversight leads to excessive compression or loss of node features in deep networks. To address this challenge, we propose a Fuzzy Cognitive Graph Neural Network (FCGNN), which models the evolution of node fuzzy state representation across layers by integrating principles from fuzzy cognitive maps. First, we construct a fuzzy cognitive graph (FCG) based on Gaussian similarity to enhance the information flow within the layer. FCG establishes fuzzy relationships between nodes to enhance the connections between different samples within the layer. Then, we design a fuzzy concept evolution (FCE) module based on dual fuzzy state representation, incorporating the fuzzy activation state to illustrate the continuous changes in node states. FCE combines explicit and implicit fuzzy state representation to achieve the dynamic evolution of node concepts between layers. Finally, we map the node states after multiple graph iterations to the category space for FSL image classification. In experiments on four benchmark datasets, FCGNN achieves superior classification performance and outperforms several baseline methods by up to 2.73% on average accuracy.**

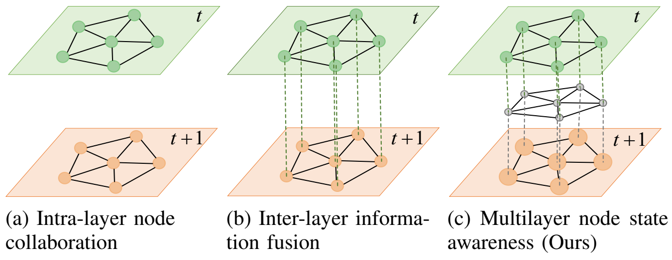
***Index Terms*—Few-shot learning, graph neural networks, fuzzy cognitive maps.**

I. Introduction

FEW-SHOT learning (FSL), a challenge that aims to train an effective model capable of recognizing new categories with limited labeled instances [1], [2], has attracted significant research interest in recent years. Inspired by human learning [3], researchers have adopted meta-learning [4] strategies with scenario training to tackle this challenge.

Meta-learning emphasizes across-task learning, which is an effective approach for solving the issue of data scarcity in FSL [5]. Existing meta-learning models can be categorized into data augmentation-based, optimization-based, and metric learning-based methods [6]. Data augmentation-based methods aim to address data scarcity by increasing training samples or improving feature representations [7], [8], [9]. Optimization-based methods primarily concentrate on adjusting model parameters to adapt to new tasks without requiring retraining [10], [11]. In practical applications, optimization-based methods are highly dependent on task-specific parameter initialization due to sensitivity to initialization and the instability of the optimization process. Additionally, data augmentation-based methods incur extra costs. Therefore, metric learning-based methods are widely used due to their simplicity and efficiency, which focus on learning a feature metric space to measure similarity in classification [12], [13], [14]. However, it typically obtains feature representations without considering sample-level category relationships, leading to limited generalization to different task scenarios.

Graph neural networks (GNNs), as an emerging approach, have gained attention in addressing the FSL problem for their ability to model relationships among samples effectively [15]. Compared with traditional meta-learning methods, GNNs-based FSL methods show great effectiveness and wide applicability in exploring the connections of finite samples [16]. Several approaches have recently been proposed to explore GNNs for FSL and achieve effective performance improvements [17], [18]. Some of these methods focus on intra-layer node collaboration, modeling node relationships through node labeling [19] or edge labeling [20]. Others consider the importance of inter-layer relationships, introducing pooling or



(a) Intra-layer node collaboration (b) Inter-layer information fusion (c) Multilayer node state awareness (Ours)

Fig. 1: Previous graph neural networks *vs* our proposed fuzzy cognitive graph neural networks (FCGNN). (a) Traditional methods primarily focus on node interactions within a single layer. (b) Some recent methods have started to implement inter-layer information fusion based on intra-layer node interactions. (c) Our method further explores multi-layer node state changes based on (a) and (b). This allows for the dynamic evolution and adjustment of node states across different layers.

This work was supported by the National Natural Science Foundation of China under Grant Nos. 62376114 and 62476195, and the Fujian Province Undergraduate Universities Teaching Reform Research Project under Grant No. FBJY20240103. (Corresponding author: Hong Zhao).

Linhua Zou, Dongqing Li, and Chengxi Jiang are with the School of Computer Science, Minnan Normal University, Zhangzhou, Fujian, 363000, China, and with the Key Laboratory of Data Science and Intelligence Application, Minnan Normal University, Zhangzhou, Fujian, 363000, China (e-mail: zlh1836065471@163.com, 18659328837@163.com, chengxi.jiang@outlook.com).

Yu Wang is with the School of Intelligence and Computing, Tianjin University, Tianjin 300350, China (e-mail: wang.yu@tju.edu.cn).

Hong Zhao is with the School of Computer Science, Minnan Normal University, Zhangzhou, Fujian, 363000, China, and with the Key Laboratory of Data Science and Intelligence Application, Minnan Normal University, Zhangzhou, Fujian, 363000, China, and also with the School of Computer and Data Science, Fuzhou University, Fuzhou 350108, China (e-mail: hongzhaocn@163.com)

gating mechanisms to model interactions between adjacent layers [21], [22].

However, these approaches that focus on intra-layer node collaboration usually assume that feature information is independent within each layer. They primarily concentrate on aggregating information from neighboring nodes within the same layer, which ignores interaction and information sharing among layers. The intuitive diagram of them is shown in Fig. 1(a). As shown in Fig. 1(b), although some improved GNNs models have started to attempt inter-layer information fusion, these methods neglect the continuous changes in node states across multiple layers. Especially in FSL scenarios with scarce data, this limitation may lead to information being gradually compressed or blurred in multilayer networks, making it impossible to efficiently transfer critical information between nodes from shallow to deep layers. Therefore, how to explore successive inter-layer state changes of nodes to enhance information transfer in multi-layer graph networks is an urgent problem to be solved.

Fuzzy cognitive maps (FCM) capture the system structure and state through iterative concept (node) updates, while dynamically adjusting relationship strengths (edges). This dual mechanism enables precise system perception and adaptive structural refinement. In this paper, we propose a framework (FCGNN) that deeply integrates the idea of FCM with GNNs. FCGNN maps such state evolution and relationship adjustment processes into graph representations for FSL, aiming to achieve dynamic evolution and adjustment of inter-layer node state (*e.g.*, Fig. 1(c)). Especially to alleviate the problems of information loss and over-compression in few-shot graph learning. First, we design a Gaussian similarity-based fuzzy cognitive graph (FCG) to enhance the information flow within the layer through fuzzy relationships. FCG leverages Gaussian similarity to construct a fuzzy cognitive graph, thus forming an end-to-end updatable graph network. Next, we introduce a fuzzy concept evolution (FCE) module based on dual fuzzy state representation for node feature update. FCE constructs explicit and implicit fuzzy state representation to highlight inter-layer node state changes, mitigating the problem of information over-compression in multi-layer networks. Finally, we employ a deep neural network to map the node states after multiple graph iterations to the category space, thus completing the image classification in the FSL task.

In summary, the main contributions of this paper are as follows:

1) From the perspective of node feature updates, we adapt node feature aggregation across multiple layers and explore the fuzzy state relationship of inter-layer nodes. To our knowledge, this is the first work that integrates FCM with GNNs to address the FSL task. This advancement achieves the perception of node state evolution across layers, which is rarely considered by most traditional FSL methods based on GNNs.

2) From the perspective of node relationship metrics, we develop a fuzzy cognitive graph using Gaussian similarity to model fuzzy relationships between nodes, enhancing intra-layer information transfer and reducing model complexity. Compared with traditional FSL methods based on GNNs, our node relationship metric is simple and non-parametric.

3) We demonstrate the effectiveness of the FCGNN by extensive experimental results on four benchmark datasets. On the 1-shot and 5-shot settings, FCGNN improves mainstream baselines by up to 2.73% and 2.22% on accuracy, respectively.

The remainder of this article is organized as follows. Section II reviews the work relevant to this study. Section III provides a detailed introduction to the FCGNN model. Section IV describes the experimental setup and results analysis, demonstrating the effectiveness of our method. Section V summarizes this article and discusses potential future research directions.

II. RELATED WORK

A. Few-Shot Learning

FSL attempts to recognize new categories leveraging only a limited number of annotated training samples, which is a challenging task [23]. Meta-learning, a mainstream framework for tackling FSL problems in scarce data scenarios, emphasizes cross-task learning to facilitate adaptation to new tasks. Existing FSL methods based on meta-learning frameworks can be roughly classified into data augmentation-based, optimization-based, and metric learning-based approaches [24].

Data augmentation-based FSL approaches address the problem of insufficient labeled data by adding samples or enhancing data features [25]. For instance, Zhang et al. [7] introduced an adversarial generator conditioned on tasks to assist few-shot classifiers in learning clearer decision boundaries. Hong et al. [8] leveraged the fusion of high-level features with low-level details to generate realistic and diverse images for novel categories. In contrast, Shi et al. [9] developed a feature enhancement method that is both globally and locally aware from the perspectives of channels and space.

Optimization-based FSL approaches focus on fine-tuning model parameters for adapting to new tasks without retraining the model. As a representative meta-learning training paradigm, Finn et al. [10] proposed a model-agnostic meta-learning method that rapidly adapts to new tasks with just a few gradient updates. Similarly, Lai et al. [11] introduced a novel meta-learning approach to learning task-adaptive classifiers that generate classifier weights for few-shot classification.

Metric-based FSL approaches aim to learn pairwise similarity metrics between queries and support samples. For example, MatchingNet [12] utilized the cosine distance to measure the distance between the query sample and the support sample, while ProtoNet [13] employed the Euclidean distance to measure the distance between the support sample prototype and the query sample. DeepEMD [26] calculated the optimal matching distance between images using the Earth Mover's Distance. Unlike these fixed distance measurement methods, it has recently been proposed to exploit deep distance measurement to learn sample relationships [14]. On this basis, RENet [27] learns relational embeddings that integrate global and local classifiers.

Although traditional meta-learning works have shown excellent performance in the FSL field, they primarily focus on rapid task adaptation and overlook the relationships among samples within a task. Our approach combines optimization and metric-based paradigms, enabling a comprehensive understanding of support and query set interactions. This facilitates the full exploration of sample relationships within tasks across tasks.

B. Graph Neural Networks

GNNs, a powerful technique designed to aggregate information from neighboring nodes within a graph, was first produced by Gori et al. [28]. Recently, with the growing emphasis on fully leveraging the interrelationship between the support and query sets in FSL, there has been increasing interest in applying GNNs to tackle FSL tasks [17], [19], [20]. Satorras et al. [17] first introduced GNNs into FSL tasks by developing an end-to-end trainable architecture that constructs different graph structures for each task. They utilized an episodic training mechanism to adjust the meta-graph parameters for predicting the labels of query nodes on the graph. Subsequently, Liu et al. [19] integrated the transduction setting into graph-based FSL by utilizing both query and support set samples for the transformational inference of label propagation graphs. In addition to leveraging the labeling information of the samples for propagation, Kim et al. [20] defined category labels and edge labels to thoroughly explore the internal information of the graph. Additionally, TPRN [29] treated each support-query pair relationship as a graph node and generated distinct relation embeddings for these pairs. Building on this, TPRN-D [30] decouples the training of the embedding network from few-shot graph modules with different tasks. In contrast, DPGN [31] explicitly integrates distributional relations into graph networks for FSL, constructing alternately updated point and distribution graphs. Similarly, Yu et al. [32] modeled the same dual graph update mechanism, except that they used it to optimize instance features and prototype features.

Although the above methods leverage message propagation in GNNs to fully explore relationships between support and query samples, they primarily focus on intra-layer node interactions. This neglect of inter-layer interaction in nodes may lead to over-fitting or over-smoothing [33], [34], particularly in few-shot scenarios with data scarcity. In contrast, some approaches [21], [22] have started to incorporate inter-layer relationships to mitigate these issues. For example, HGNN [21] employed multi-level feature fusion and residual techniques to obtain inter-layer node relationships. Thus exploring the hierarchical relationships between samples. Building on this, CSTS [22] exploded the hierarchical relationships among samples while integrating useful information from different layers in a multilayer pooling process. Furthermore, AGNN [33] introduced a layer memory attention mechanism to pass the output features and adjacency matrix of the current layer to the next layer as edge knowledge. Unlike AGNN, MGGN [18] integrated a graph network and a modified gated recursive unit to sense the trend of edge feature changes and establish inter-layer dependencies.

Existing studies reveal that inter-layer relationship modeling effectively alleviate over-fitting and over-smoothing in GNNs-based FSL scenarios. However, most prior solutions focused primarily on local interactions between neighboring layers. Therefore, our method explicitly accounts for the continuous state changes of nodes across multiple layers. Compared with previous approaches, this method enables node feature information to be retained across sequential GNNs stages.

C. Fuzzy Cognitive Maps

The combination of fuzzy logic and deep learning has attracted growing interest from researchers [35], [36]. FCM, a product of the combination of fuzzy logic and neural networks, is used to represent and analyze the causal relationships between different concepts in complex systems [37]. Recently, FCM has emerged as a powerful system state prediction model in various fields [38], [39], [40]. For example, Wang et al. [38] introduced a deep FCM for multivariate time series prediction. They simultaneously exploit the advantages of deep neural networks in prediction and FCM in interpretation. PRV-FCM [39] developed an approach utilizing FCM and meta-heuristic algorithms to generate prescriptive models that support decisions across various fields. Hoyos et al. [40] established three federated learning methods based on FCM. They were applied to predict mortality and treatment prescriptions for severe dengue. In addition, recent approaches have integrated fuzzy logic with GNNs to enhance knowledge representation under uncertainty [41]. For example, Wei et al. [42] proposed a dynamic fuzzy sampler utilizing Gaussian fuzzy systems, demonstrating the great potential of fuzzy ideas in graph learning. Fan et al. [43] designed an uncertainty-aware superpoint graph transformer that propagates supervised signals through attention-based fuzzy subsets.

Building upon the strengths of FCM, our work introduces the concept of multi-layer state evolution. Then constructs a fuzzy cognitive graph based on Gaussian similarity. Thus forming a fuzzy cognitive graph neural network that can be applied and extended to FSL tasks. Unlike traditional GNNs that mainly focus on intra- or inter-layer interactions, our approach considers continuous interactions of node state between multiple layers and intra-layer information transfer.

III. PROPOSED FCGNN MODEL

We present the overall framework of FCGNN at the beginning of this section. Next, we describe the three modules of FCGNN in detail. Finally, we provide the pseudo-code for FCGNN, followed by an analysis and discussion.

A. Overview

FCGNN primarily consists of three modules: A node fuzzy relation calculation module, a node fuzzy state representation update module, and a module for few-shot image classification. The general framework of our proposed FCGNN model is illustrated in Fig. 2.

1) Gaussian similarity-based fuzzy cognitive graph construction (FCG): Based on Gaussian similarity, we establish fuzzy relationships between nodes to construct the fuzzy cognitive graph. Additionally, we utilize the FCG module to obtain the fuzzy relationships among nodes in the current layer for updating the graph network. This approach enhances the flow of information within the layer.

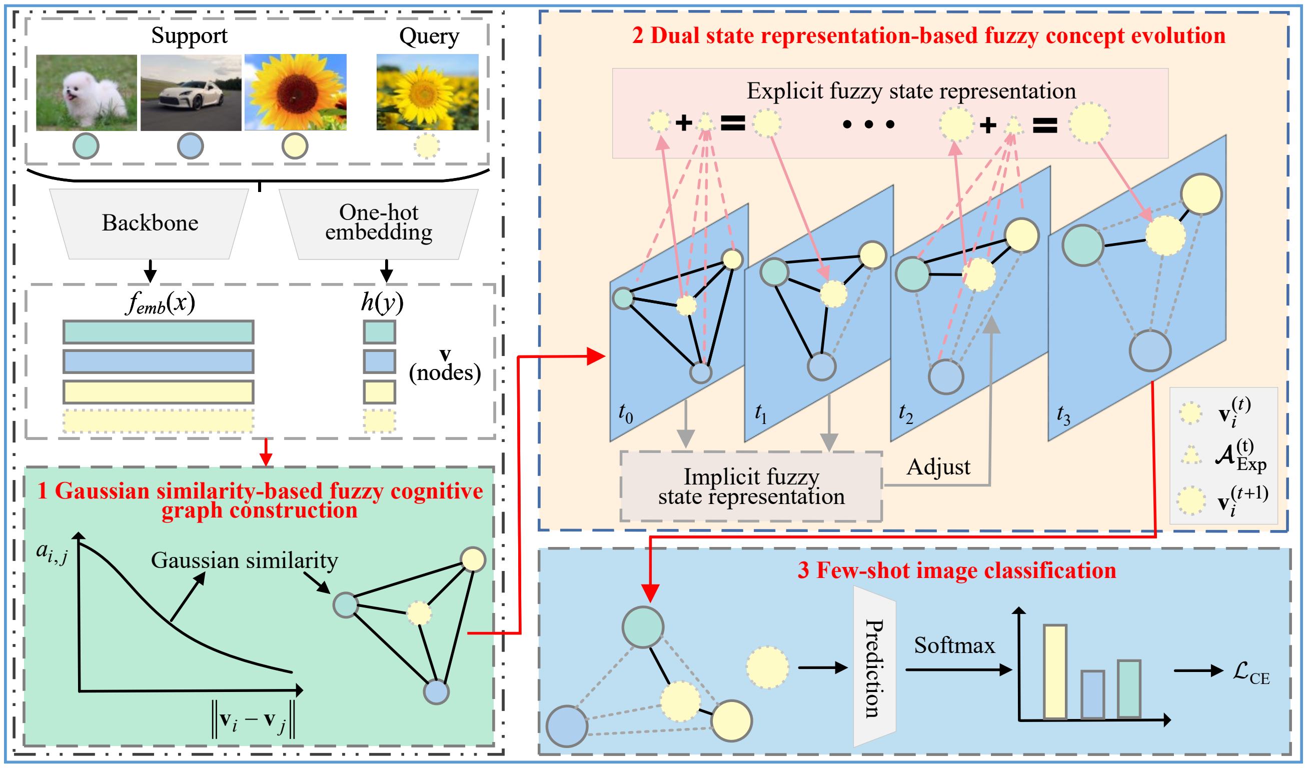

Support
Query
Backbone
One-hot embedding
$f_{emb}(x)$
$h(y)$
v (nodes)
1 Gaussian similarity-based fuzzy cognitive graph construction
$a_{i,j}$
Gaussian similarity
$\|\mathbf{v}_i - \mathbf{v}_j\|$
2 Dual state representation-based fuzzy concept evolution
Explicit fuzzy state representation
t_0
t_1
t_2
t_3
Implicit fuzzy state representation
Adjust
$\mathbf{v}_i^{(t)}$
$\mathcal{A}_{\text{Exp}}^{(t)}$
$\mathbf{v}_i^{(t+1)}$
3 Few-shot image classification
Prediction
Softmax
$\mathcal{L}_{\text{CE}}$

Fig. 2: Framework of FCGNN.

2) Dual state representation-based fuzzy concept evolution (FCE): The FCE module updates node features leveraging two fuzzy state representation. The explicit fuzzy state representation captures the continuous evolution of node states across layers, while the implicit fuzzy state representation controls the cumulative influence from neighboring nodes. These representations are measured through explicit and implicit fuzzy activation state, respectively.

3) Few-shot image classification: After several iterations of the graph, we employ the improved node features for few-shot image classification. The final module maps the evolved node to the category space to perform image classification.

B. Gaussian Similarity-based Fuzzy Cognitive Graph Construction

FSL is formalized as an N-way K-shot classification task, aiming to train a model to recognize N novel categories using only K labeled examples per category. Given a dataset $\mathcal{D}$, it is divided into a disjoint training set $\mathcal{D}_{\text{train}}$ and a test set $\mathcal{D}_{\text{test}}$ according to the categories. The training set $\mathcal{D}_{\text{train}}$ and the test set $\mathcal{D}_{\text{test}}$ are utilized to train and evaluate the model. Both sets further employ the meta-learning paradigm [1] to construct multiple episodic tasks that simulate FSL scenarios. Each episodic task $\mathcal{T} = \mathcal{S} \cup \mathcal{Q}$ contains a support set $\mathcal{S}$ and a query set $\mathcal{Q}$. The support set $\mathcal{S} = \{(x_i, y_i)\}_{i=1}^{N\times K}$ is used to simulate the N-way K-shot scenario for training, where y_i denotes the label associated with sample x_i, N denotes the number of selected categories, and K denotes the number of labeled samples belonging each category. The query set $\mathcal{Q} = \{(x_i, y_i)\}_{i=N\times K+1}^{N\times K+N\times Q}$ is used to evaluate the model, where Q denotes the number of unlabeled samples in each category.

In FCGNN, each episodic task $\mathcal{T}$ is represented as a graph structure G. The graph corresponding to the t-th layer is denoted by $G^{(t)} = (\mathbf{V}^{(t)}, \mathbf{A}^{(t)})$, where $0 \le t \le T$, T is the total number of graph layers. The node feature matrix $\mathbf{V}^{(t)} = [\mathbf{v}_1^{(t)}; \cdots; \mathbf{v}_i^{(t)}; \cdots; \mathbf{v}_n^{(t)}] \in \mathbb{R}^{n\times d^{(t)}}$ includes n node feature vectors, where $n = N \times K + N \times Q$ denotes the total number of samples in both the support and query sets, and $d^{(t)}$ denotes the dimension of the node feature vector in the t-th layer. Additionally, we utilize the Gaussian similarity among nodes to construct the adjacency matrix $\mathbf{A}^{(t)} = [a_{i,j}^{(t)}]_{n\times n}$, where each element $a_{i,j}^{(t)}$ denotes the relationship between the i-th and j-th nodes in the t-th layer.

Initially, we combine the original feature of sample x_i with its one-hot coding to obtain initial node feature vector $\mathbf{v}_i^{(0)}$ by

$$\mathbf{v}_i^{(0)} = (f_{emb}(x_i), h(y_i)), \tag{1}$$

where $f_{emb}(\cdot)$ is a feature extractor with M-dimensional output and $h(\cdot)$ is a one-hot encoder with N-dimensional output. For support samples, we utilize the traditional one-hot encoder [22]. For query samples, we employ a uniform distribution $\left[\frac{1}{N}, \cdots, \frac{1}{N}\right]$ that contains N values instead of using the one-hot vector representation. In this way, we obtain the initial node feature matrix $\mathbf{V}^{(0)} = [\mathbf{v}_1^{(0)}; \cdots; \mathbf{v}_i^{(0)}; \cdots; \mathbf{v}_n^{(0)}] \in \mathbb{R}^{n\times d^{(0)}}$, where $d^{(0)} = M + N$ represents the dimension of the initial node feature vector. In subsequent layers, the dimension of the node feature vector $\mathbf{v}_i^{(t)}$ in the t-th layer is given by $d^{(t)} = d^{(0)} + m \times t$, where m denotes a fixed increase in the feature dimensions for each layer.

Next, consider that successive changes in the state of subsequent nodes alter the node feature dimensions. Therefore, we normalize the node feature vector to avoid large values affecting subsequent similarity calculations. Specifically, for the node feature vector $\mathbf{v}_i^{(t)}$ in the t-th layer, we calculate its feature mean $\mu_i^{(t)}$ by

$$\mu_i^{(t)} = \frac{1}{d^{(t)}} \sum_{k=1}^{d^{(t)}} v_{i,k}^{(t)}. \tag{2}$$

We then calculate the standard deviation $\gamma_i^{(t)}$ of node feature vector $\mathbf{v}_i^{(t)}$ using its feature mean $\mu_i^{(t)}$ by

$$\gamma_i^{(t)} = \sqrt{\frac{1}{d^{(t)}-1} \sum_{k=1}^{d^{(t)}} \left(v_{i,k}^{(t)} - \mu_i^{(t)}\right)^2}. \tag{3}$$

Subsequently, we obtain normalized node feature vector $\mathbf{v}_i^{(t)'}$ in the t-th layer as

$$\mathbf{v}_i^{(t)'} = \frac{\mathbf{v}_i^{(t)} - \mu_i^{(t)}}{\gamma_i^{(t)} + \varepsilon}, \tag{4}$$

where ε is a small constant (*e.g.*, 10^{-6}) to avoid numerical instability when the standard deviation nears zero.

Finally, we compute the Gaussian similarity between nodes to obtain adjacency matrix $\mathbf{A}^{(t)} = [a_{i,j}^{(t)}]_{n\times n}$, where $a_{i,j}^{(t)}$ is calculated as

$$a_{i,j}^{(t)} = \exp\left(-\frac{\|\mathbf{v}_i^{(t)'} - \mathbf{v}_j^{(t)'}\|_2^2}{2\sigma^2}\right), \tag{5}$$

where $\|\cdot\|_2$ represents the ℓ_2-norm, and σ is the fuzzy strength hyperparameter. Increasing the value of σ enhances the fuzzy similarity among all nodes.

C. Dual State Representation-based Fuzzy Concept Evolution

In the following sections, we first present the implicit fuzzy state representation for dynamically adjusting the node relationships. Next, we demonstrate the process of generating the explicit fuzzy state representation.

(1) Implicit fuzzy state representation: After obtaining the adjacency matrix $\mathbf{A}^{(t)}$ in the t-th layer according to (5), we develop an implicit fuzzy state representation of inter-layer nodes to dynamically adjust the node relationships. Concretely, we propose an implicit fuzzy state representation of the nodes to indicate the degree of node information aggregation in the current layer, as shown in Fig. 3. This approach helps guide neighborhood interactions to regulate feature aggregation.

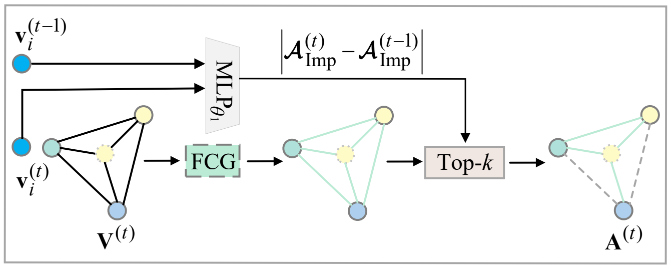


Fig. 3: Dynamic adjustment of node relationships based on implicit fuzzy state representation.

First, we calculate the implicit fuzzy state representation $\mathcal{A}_{\text{Imp}}^{(t)}$ of the nodes in the t-th layer by

$$\mathcal{A}_{\text{Imp}}^{(t)} = \mathcal{M}\left(\mathbf{V}^{(t)}\right) = \text{Sigmoid}\left(\text{MLP}_{\theta_1}\left(\mathbf{V}^{(t)}\right)\right), \tag{6}$$

where $\mathcal{M}(\cdot)$ serves as a membership function to compute the implicit fuzzy state representation of the nodes, $\text{MLP}_{\theta_1}(\cdot)$ is a multilayer trainable convolutional neural network, and function Sigmoid$(\cdot)$ controls the value of the implicit fuzzy state representation within the range [0,1].

Next, we calculate the implicit fuzzy state representation $\mathcal{A}_{\text{Imp}}^{(t-1)}$ and $\mathcal{A}_{\text{Imp}}^{(t)}$ according to (6). They reflect the degree of node information aggregation in the $(t-1)$-th layer and t-th layer, respectively. Subsequently, we obtain the implicit fuzzy state representation difference $\mathbf{S}^{(t)} = [s_1^{(t)}, \cdots, s_i^{(t)}, \cdots, s_n^{(t)}]$ for the nodes between the t-th layer and $(t-1)$-th layer by

$$\mathbf{S}^{(t)} = \left|\mathcal{A}_{\text{Imp}}^{(t)} - \mathcal{A}_{\text{Imp}}^{(t-1)}\right|, \tag{7}$$

where $|\cdot|$ denotes an absolute value operation. We sort the n values of $\mathbf{S}^{(t)}$ in ascending order and select the indices of the top-$k_{\text{node}}^{(t)}$ nodes. These top-$k_{\text{node}}^{(t)}$ nodes with a smaller value of $s_i^{(t)}$ are considered to have aggregated sufficient neighbor information. Therefore, we reduce or eliminate the neighbor information aggregation of these nodes in the next graph update. Specially, we adjust the edge weights of these $k_{\text{node}}^{(t)}$ nodes to obtain a new adjacency matrix $\mathbf{A}^{(t)}$ by

$$\mathbf{A}^{(t)} = \mathbf{A}^{(t)} \odot \mathbf{M}^{(t)}, \tag{8}$$

where $\odot$ is the element-wise product and $\mathbf{M}^{(t)}$ is the mask matrix constructed using the $k_{\text{node}}^{(t)}$ node index. In FCGNN, we utilize two different strategies to construct $\mathbf{M}^{(t)}$. One strategy is to make all elements equal to 1 except all edges connected to the index nodes. This allows us to directly select all edges of the index nodes for adjustment. Another strategy is to first select the index nodes and then further select the smaller edges connected to these nodes for adjustment. This enables a more precise adjustment of the edge weights associated with the index nodes. The specific adjustment strategies are detailed in the supplemental material.

(2) Explicit fuzzy state representation: FCM emphasizes the dynamic adjustment of node states and relationships through iterative updates. We simulate it by designing an explicit fuzzy state representation to quantify the inter-layer state evolution of nodes. FCGNN considers multiple layers of continuous changes in node state as shown in Fig. 4. We next outline a complete process for updating the graph network utilizing explicit fuzzy state representation.

First, we obtain a new node feature matrix $\widetilde{\mathbf{V}}^{(t)}$ in the t-th layer by

$$\widetilde{\mathbf{V}}^{(t)} = \mathbf{A}^{(t)}\mathbf{V}^{(t)}, \tag{9}$$

where $\mathbf{V}^{(t)}$ denotes the node feature matrix in the t-th layer.

Next, we calculate the explicit fuzzy state representation $\mathcal{A}_{\text{Exp}}^{(t)}$ of the nodes in the t-th layer by

$$\mathcal{A}_{\text{Exp}}^{(t)} = \varphi\left(\text{MLP}_{\theta_2}\left(\mathbf{V}^{(t)} \oplus \widetilde{\mathbf{V}}^{(t)}\right)\right), \tag{10}$$

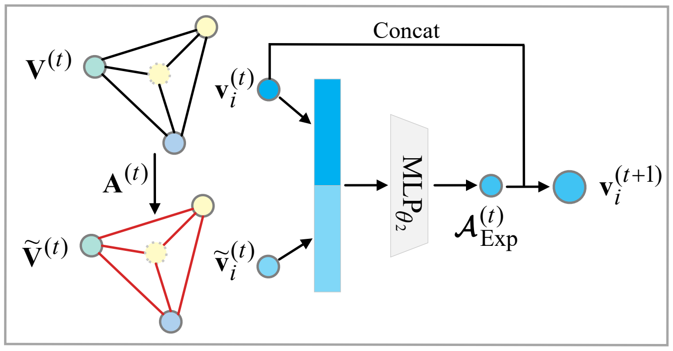


Fig. 4: Node feature update based on explicit fuzzy state representation.

where $\oplus$ denotes row-wise feature concatenation, $\text{MLP}_{\theta_2}(\cdot)$ denotes the learnable convolutional network layer that performs dimensional transformation, and $\varphi(\cdot)$ denotes a nonlinear function known as Leaky-ReLU. We employ the explicit fuzzy state representation $\boldsymbol{\mathcal{A}}_{\text{Exp}}^{(t)} \in \mathbb{R}^{n\times m}$ to quantify node state change after information aggregation in the t-th layer, where m denotes the feature dimension of the explicit fuzzy state representation.

Finally, we derive the node feature matrix $\mathbf{V}^{(t+1)}$ in the $(t+1)$-layer as

$$\mathbf{V}^{(t+1)} = \mathcal{C}\left(\mathbf{V}^{(t)}, \boldsymbol{\mathcal{A}}_{\text{Exp}}^{(t)}\right), \tag{11}$$

where $\mathcal{C}(\cdot)$ denotes connectivity operation at the feature dimension level. Following the above node feature update approach, we obtain the node feature matrix $\mathbf{V}^{(T)} = [\mathbf{v}_1^{(T)}; \cdots; \mathbf{v}_i^{(T)}; \cdots; \mathbf{v}_n^{(T)}] \in \mathbb{R}^{n\times d^{(T)}}$ in the final layer.

In summary, we first utilize implicit fuzzy state representation to guide inter-layer node information aggregation. Next, we design an explicit fuzzy state representation to reveal continuous node state changes during graph updates.

D. Few-Shot Image Classification

In FCGNN, we adopt an end-to-end single-stage training strategy, where the feature extractor is trained jointly with the graph update module instead of being pre-trained. For the i-th node, we map the node feature vector $\mathbf{v}_i^{(T)}$ to an N-dimensional vector $\mathbf{z}_i = [z_{i,1}, \cdots, z_{i,j}, \cdots, z_{i,N}]$, where each element $z_{i,j}$ represents the logit for the i-th node corresponding to j-th category. Specifically, the vector $\mathbf{z}_i$ is obtained by

$$\mathbf{z}_i = \text{P}_{\theta_3}\left(\mathbf{v}_i^{(T)}\right), \tag{12}$$

where $\text{P}_{\theta_3}(\cdot)$ is a trainable neural network that maps this node feature vector to category space. Next, the classification probability $p_{i,j}$ that the i-th node belongs to the j-th category is calculated by

$$p_{i,j} = \frac{\exp(z_{i,j})}{\sum_{j=1}^{N}\exp(z_{i,j})}. \tag{13}$$

Finally, we obtain the optimal parameter set $\boldsymbol{\theta}^*$ by optimizing the cross-entropy loss as

$$\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}^*}{\arg\min}\left(-\frac{1}{N\times Q}\sum_{i=N\times K+1}^{N\times K+N\times Q}\sum_{j=1}^{N} y_{i,j}\log(p_{i,j})\right), \tag{14}$$

where $y_{i,j} = 1$ if the i-th node belongs to the j-th category and $y_{i,j} = 0$ otherwise.

E. Model Analysis and Discussion

Algorithm 1 demonstrates the complete training process of FCGNN for a few-shot classification task $\mathcal{T}$. First, we obtain the initial node feature vectors (line 1). Next, we construct the graph structure by obtaining the node relationships using Gaussian similarity (line 4). We then refine the node relationships based on the implicit fuzzy state representation of the nodes (lines 6 and 7). Subsequently, the node features are updated in conjunction with the explicit fuzzy state representation (lines 9 and 10). Finally, we map the updated node feature to the category space and return the optimal parameter set for the few-shot task (lines 12 and 13).

Algorithm 1: Detailed Training Process of FCGNN.

Input: The few-shot classification tasks $\mathcal{T} = \mathcal{S} \cup \mathcal{Q}$. The training epoch number E. The number of graph network layers T.

Output: The optimal parameter set $\boldsymbol{\theta}^*$ of FCGNN.

```
1:  Obtain $\mathbf{v}_i^{(0)}$ by (1);
2:  for $epoch = 0 : (E-1)$ do
3:      for $t = 0 : (T-1)$ do
4:          Obtain $\mathbf{A}^{(t)}$ by (5);
5:          if $t > 0$ then
6:              Calculate $\boldsymbol{\mathcal{A}}_{\text{Imp}}^{(t-1)}$ and $\boldsymbol{\mathcal{A}}_{\text{Imp}}^{(t)}$ by (6);
7:              Dynamically adjust node relationships to obtain
                new $\mathbf{A}^{(t)}$ by (8);
8:          end if
9:          Calculate $\boldsymbol{\mathcal{A}}_{\text{Exp}}^{(t)}$ by (10);
10:         Calculate $\mathbf{V}^{(t+1)}$ by (11);
11:     end for
12:     Calculate $p_{i,j}$ by (13);
13:     Optimize the cross-entropy loss of FCGNN by (14);
14: end for
15: return $\boldsymbol{\theta}^*$;
```

Compared with the iterative dynamics of classical FCM, which involve repeating state updates on a static graph structure in the temporal dimension, the hierarchical updates of FCGNN involve feature transformations on node representations in the spatial dimension. The correspondence between these two is essentially a mapping of the temporal evolution process of FCM into the spatial propagation path of GNNs.

Compared with traditional graph-based FSL methods, FCGNN has the following advantages. First, FCGNN combines FCM and introduces the concept of fuzzy state representation to emphasize the inter-layer state changes in nodes. Unlike previous methods that implicitly update nodes, FCGNN quantifies the inter-layer node state changes and dynamically adjusts the information aggregation of nodes. This alleviates the information loss due to the over-compression of node features during traditional graph updating. Second, FCGNN employs Gaussian similarity to construct the graph structure. This non-parametric metric for assessing node relationships simplifies and accelerates the reasoning process. Finally, FCGNN degrades to the traditional node feature update mode if the inter-layer state changes of nodes are not considered. Therefore, FCGNN can extend the graph-based FSL approach that utilizes the traditional node feature update mode.

IV. EXPERIMENTS

In this section, we first briefly describe the relevant experimental setups. Next, we conduct four experiments to thoroughly analyze and evaluate our model: 1) Comparison with related models; 2) Ablation experiment; 3) Sensitivity analysis of Gaussian fuzzy strength hyperparameter; 4) Analysis of inter-layer node state evolution; 5) Analysis of computational efficiency; 6) Analysis of network depth and over-smoothing.

A. Experiment Setups

(1) Datasets: We evaluate our model on four widely used few-shot datasets. miniImageNet [12], derived from the ImageNet [44] dataset, is one of the most commonly employed benchmarks for FSL tasks. It consists of 100 categories, each containing 600 images. CUB [45] is a fine-grained dataset for bird image classification. It contains images of 200 different bird species with a total of 11,788 images. tieredImageNet [46] extends miniImageNet with a more complex category hierarchy. It includes 608 categories organized into 20 superclasses, each containing multiple subcategories, resulting in 779,165 images. CIFAR-FS [47], based on the CIFAR-100 dataset, presents a more challenging image classification task than the original CIFAR-100. It consists of 100 categories, with 600 images per category. We follow the dataset partitioning proposed by [15] as shown in Table I.

TABLE I
DATASET DIVISION DESCRIPTIONS.

Dataset	Train	Val	Test
miniImageNet	64	16	20
CUB	100	50	50
tieredImageNet	351	97	160
CIFAR-FS	64	16	20

(2) Implementation Details: We use a Conv4 as the feature extractor, resulting in 128-dimensional features from the four datasets. The image sizes for the three datasets are resized to 84 × 84, except for the CIFAR-FS dataset, which retains the image size of 32 × 32. In addition, we perform typical data augmentation strategies (*e.g.*, RandomCrop, etc.) during the data processing. All experiments were performed using the Adam optimizer with an initial learning rate and a weight decay of 1×10^{-3} and 1×10^{-6}, respectively. The number of GNNs layers was set to 4 for all datasets. Experiments are conducted on hardware equipped with a GeForce RTX 4090D GPU using PyTorch.

(3) Baselines: To evaluate the proposed model, we compared it with the following three types of methods. The first is the optimisation-based FSL approaches: MAML [10], LEO [4], MetaTEDL [48]. The second is the metric-based FSL approaches: MatchingNet [12], ProtoNet [13], RelationNet [14], SAPENet [49], GLFA [9]. The last methods are the same type as ours, which are graph network-based FSL approaches: GNN [17], TPN [19], EGNN [20], HybridGNN [32], TRPN-D [30], CSTS [22], MGGN [18], etc. Furthermore, we compare our approach with graph methods based on dynamic graph and residual structure: HGNN [21], MTSGM [24], AGNN [33], FSAKE [50]. For a fair comparison, all baseline results are reported from the original article or related articles.

B. Comparison with Related Models

In this subsection, we compare and analyze the experimental results of FCGNN with other related models. "Type" denotes other types of methods related to our method, where "O" denotes optimization-based methods, metric-based is labeled as "M", and graph-based is labeled as "G". Furthermore, we utilize "G*" to represent graph methods based on dynamic graph and "G†" to represent graph methods based on residual structure. The best results are marked in bold.

(1) Results on miniImageNet: Table II summarizes the results of the comparative experiments conducted in the 5-way few-shot classification task on miniImageNet.

TABLE II
RESULTS OF 5-WAY FEW-SHOT CLASSIFICATION AVERAGE ACCURACY (%) WITH 95% CONFIDENCE INTERVALS ON MINIIMAGENET.

Method	Type	Backbone	miniImageNet 1-shot	miniImageNet 5-shot
MAML [10]	O	Conv4	54.69 ± 0.89	66.62 ± 0.83
LEO [4]	O	WRN28	61.76 ± 0.08	77.59 ± 0.12
MetaTEDL [48]	O	ResNet12	60.40 ± 0.80	78.00 ± 0.60
MatchingNet [12]	M	Conv4	43.56 ± 0.84	55.31 ± 0.73
ProtoNet [13]	M	Conv4	49.42 ± 0.78	68.20 ± 0.66
RelationNet [14]	M	Conv4	50.40 ± 0.80	65.30 ± 0.70
SAPENet [49]	M	Conv4	55.71 ± 0.20	71.81 ± 0.16
GNN [17]	G	Conv4	54.72 ± 0.70	75.41 ± 0.47
TPN [19]	G	Conv4	55.51 ± 0.86	69.43 ± 0.65
EGNN [20]	G	Conv4	59.63 ± 0.52	76.37 ± 0.48
HybridGNN [32]	G	Conv4	55.63 ± 0.20	72.48 ± 0.16
HGNN [21]	G*	Conv4	60.91 ± 0.73	77.54 ± 0.66
CSTS [22]	G	Conv4	62.38 ± 0.48	79.77 ± 0.44
MGGN [18]	G	Conv4	64.55 ± 0.81	**81.89 ± 0.51**
TRPN-D [30]	G	Conv4	62.98 ± 0.50	81.24 ± 0.42
AGNN [33]	G†	Conv4	64.25 ± 0.42	80.74 ± 0.36
FSAKE [50]	G†	Conv4	61.86 ± 0.72	79.66 ± 0.62
FCGNN (Ours)	G	Conv4	**64.59 ± 0.53**	81.88 ± 0.45

From the comparative performance, we can draw the following conclusions.

1) FCGNN outperforms optimization-based and metric-based methods in both 1-shot and 5-shot settings. For example, the average accuracy of FCGNN improves over LEO by 2.83% and 4.29% on the 1-shot and 5-shot settings, respectively. In addition, FCGNN outperforms RelationNet and SAPENet by 14.19% and 8.88% on the 1-shot setting, respectively. These improvements demonstrate that GNNs better capture the sample relationship between the support set and the query set in FSL.

2) It can be observed that FCGNN far outperforms traditional graph FSL methods. Even compared with some recent methods that consider inter-layer relationships, our method has advantages. For instance, FCGNN outperforms EGNN and

CSTS by 4.96% and 2.21% on the 1-shot setting, respectively. These results further illustrate that FCGNN obtains effective performance improvement by considering continuous state changes in multiple layers.

3) We observe that FCGNN surpasses other methods of the same type. Even under the 5-shot setting, compared with the best comparison method MGGN, it is only slightly behind. MGGN needs a complex gating mechanism to update the graph network, and FCGNN simply exploits the state changes of the nodes in the layer updates. This shows that our method is simpler and easier to implement while maintaining competitive performance.

4) Compared with graph methods that use residual structure, FCGNN demonstrates superior performance. For instance, FCGNN achieves 1.14% and 2.22% performance gains over AGNN and FSAKE under the 5-shot setting. We speculate that this is due to the continuous state transfer mechanism of FCGNN, which preserves deep feature relationships and reduces information loss.

(2) Results on CUB: Table III presents the results of our comparative experiments on the 5-way few-shot classification task applied to CUB.

TABLE III
RESULTS OF 5-WAY FEW-SHOT CLASSIFICATION AVERAGE ACCURACY (%) WITH 95% CONFIDENCE INTERVALS ON CUB.

Method	Type	Backbone	CUB	
			1-shot	5-shot
MAML [10]	O	Conv4	55.92 ± 0.95	72.09 ± 0.76
ProtoNet [13]	M	ResNet12	68.92 ± 0.71	85.92 ± 0.43
RelationNet [14]	M	Conv4	62.45 ± 0.98	76.11 ± 0.69
GLFA [9]	M	ResNet12	76.52 ± 0.37	90.27 ± 0.38
SAPENet [49]	M	Conv4	70.38 ± 0.23	84.47 ± 0.14
GNN [17]	G	Conv4	68.56 ± 0.85	83.12 ± 0.47
EGNN [20]	G	Conv4	67.25 ± 0.42	82.15 ± 0.30
DPGN [31]	G	ResNet12	75.71 ± 0.47	91.48 ± 0.33
HybridGNN [32]	G	Conv4	69.02 ± 0.22	83.20 ± 0.15
HGNN [21]	G^*	Conv4	69.43 ± 0.49	87.67 ± 0.45
AGNN [33]	$G^\dagger$	Conv4	75.81 ± 0.48	88.22 ± 0.35
MTSGM [24]	G^*	Conv4	75.91 ± 0.22	87.75 ± 0.14
CSTS [22]	G	Conv4	60.83 ± 0.45	77.12 ± 0.44
FSAKE [50]	$G^\dagger$	Conv4	77.00 ± 0.70	89.66 ± 0.50
FCGNN (Ours)	G	Conv4	**82.00± 0.48**	**92.55 ± 0.32**

Based on these results, we can make the following observations and conclusions.

1) FCGNN demonstrates significant performance improvements on a more challenging fine-grained dataset. FCGNN achieves the highest experimental results on CUB. For instance, FCGNN surpasses the next best method by 5.00% and 2.89% under 1-shot and 5-shot settings, respectively. This suggests that FCGNN has an advantage in capturing more detailed category differences.

2) Despite utilizing a relatively simple Conv4 feature extractor, FCGNN outperforms models that employ the deeper ResNet12 feature extractor. For example, FCGNN achieves 2.28% and 1.07% higher accuracy than GLFA and DPGN in the 5-shot setting, respectively. We attribute this performance gain to the integration of node state information at different layers, which helps mitigate the limitations of the shallow feature extractor.

3) Compared with graph methods that use a dynamic graph, FCGNN achieves the best performance. For example, FCGNN outperforms MTSGM using graph attention networks by 6.09% and 4.80% in the 1-shot and 5-shot settings, respectively. We speculate that the dynamic fuzzy state adjustment mechanism of FCGNN achieves a better relationship modeling than the traditional graph attention network.

(3) Results on tieredImageNet: Table IV displays the results of our comparative experiments on the 5-way few-shot classification task applied to tieredImageNet.

TABLE IV
RESULTS OF 5-WAY FEW-SHOT CLASSIFICATION AVERAGE ACCURACY (%) WITH 95% CONFIDENCE INTERVALS ON TIEREDIMAGENET.

Method	Type	Backbone	tieredImageNet	
			1-shot	5-shot
MAML [10]	O	Conv4	51.67 ± 1.81	70.30 ± 1.75
HTS [51]	O	Conv4	53.20 ± 0.22	72.38 ± 0.19
ProtoNet [13]	M	Conv4	53.34 ± 0.89	72.69 ± 0.74
RelationNet [14]	M	Conv4	54.48 ± 0.93	71.32 ± 0.78
SAPENet [49]	M	Conv4	57.61 ± 0.22	75.42 ± 0.18
GNN [17]	G	Conv4	43.56 ± 0.84	55.31 ± 0.73
TPN [19]	G	Conv4	59.91 ± 0.94	73.30 ± 0.75
EGNN [20]	G	Conv4	62.95 ± 0.52	80.24 ± 0.49
HybridGNN [32]	G	Conv4	56.05 ± 0.21	72.82 ± 0.18
HGNN [21]	G^*	Conv4	63.95 ± 0.74	82.84 ± 0.61
TRPN-D [30]	G	Conv4	61.01 ± 0.49	80.98 ± 0.42
CSTS [22]	G	Conv4	64.84 ± 0.26	82.95 ± 0.44
MGGN [18]	G	Conv4	65.92 ± 0.87	83.16 ± 0.37
FSAKE [50]	$G^\dagger$	Conv4	65.27 ± 0.73	83.33 ± 0.62
FCGNN (Ours)	G	Conv4	**66.76± 0.54**	**84.99 ± 0.42**

From these results, we draw several observations and conclusions.

1) FCGNN obtains a performance improvement of 13.56% and 1.92% on the 1-shot setting compared with the non-graph method HTS and the graph method CSTS, which use hierarchical structures. Additionally, FCGNN outperforms HGNN with dynamic pooling learning hierarchical structure by 2.81% and 2.15% under 1-shot and 5-shot settings. This shows that FCGNN achieves good performance on a hierarchical dataset without utilizing the hierarchical structure of the data.

2) With the same feature extractor, FCGNN is still able to outperform other types of methods under a large-scale dataset. This demonstrates that the fuzzy cognitive graph architecture we developed effectively addresses the FSL problem.

(4) Results on CIFAR-FS: The comparative experimental results for the 5-way few-shot classification task on the CIFAR-FS dataset are summarized in Table V.

From these results, we observe that FCGNN shows superior performance over suboptimal methods in both 1-shot and 5-shot settings. This demonstrates that our method can adapt to various few-shot settings while maintaining strong generalization capabilities. Furthermore, FCGNN outperforms the recent MetaTEDL and FSAKE approaches, achieving advantages of 1.06% and 2.58% in the 1-shot setting. This further illustrates the effectiveness of our method.

TABLE V
RESULTS OF 5-WAY FEW-SHOT CLASSIFICATION AVERAGE ACCURACY (%) WITH 95% CONFIDENCE INTERVALS ON CIFAR-FS.

Method	Type	Backbone	CIFAR-FS	
			1-shot	5-shot
MetaTEDL [48]	O	ResNet12	71.30 ± 0.90	85.20 ± 0.60
ProtoNet [13]	M	Conv4	55.50 ± 0.70	72.00 ± 0.60
RelationNet [14]	M	Conv4	55.00 ± 1.00	69.30 ± 0.80
HGNN [21]	G*	Conv4	67.67 ± 0.73	86.16 ± 0.56
CSTS [22]	G	Conv4	62.47 ± 0.47	81.82 ± 0.42
MTSGM [24]	G*	Conv4	72.18 ± 0.22	82.21 ± 0.17
FSAKE [50]	G†	Conv4	69.78 ± 0.69	85.92 ± 0.55
FCGNN (Ours)	G	Conv4	**72.36± 0.53**	**86.22 ± 0.42**

C. Ablation Experiment

To verify the effectiveness of the model components, we test them on miniImageNet and CUB under 1-shot and 5-shot settings. The experimental results of model variables are shown in Table VI. We can draw the following conclusions from these ablation results.

1) We observe that using the FCG module alone degrades the model performance. We conclude that the probabilistic connection utilizing FCG alone aggravates the over-smoothing drawback commonly caused by traditional graph networks. Specifically, FCG is a fixed metric instead of a depth metric and relies on good node representation to improve model performance. In contrast, combining it with the FCE module to construct a complete fuzzy cognitive graph significantly improves the model performance. We speculate that this is due to the fact that FCE preserves the inter-layer discriminative patterns of the nodes. Experiments demonstrate that these are two tightly coupled modules. The cross-layer fuzzy state representation generated by FCE can evolve gradually through the fuzzy relational modeling of FCG.

2) Employing the FCE in FCGNN leads to a performance improvement of 1.03% and 2.53% for a baseline on miniImageNet under 1-shot and 5-shot settings, respectively. Even on the more challenging fine-grained dataset, there is a significant improvement of 2.09% and 3.30%. This improvement confirms that FCE effectively captures the continuous state evolution of node features through two fuzzy state representations. FCE alleviates the feature degradation phenomenon of traditional GNNs due to inter-layer compression and improves the model performance.

TABLE VI
ABLATION STUDY OF FCGNN COMPONENTS ON MINIIMAGENET AND CUB.

Method	miniImageNet		CUB	
	1-shot	5-shot	1-shot	5-shot
Baseline	61.11 ± 0.46	75.25 ± 0.44	78.01 ± 0.41	88.36 ± 0.35
+FCG	59.33 ± 0.46	73.03 ± 0.40	75.71 ± 0.42	87.86 ± 0.32
+FCE	62.14 ± 0.53	77.78 ± 0.44	80.10 ± 0.43	91.66 ± 0.34
+FCG+FCE	64.59 ± 0.53	81.88 ± 0.45	82.00 ± 0.48	92.55 ± 0.32

D. Sensitivity Analysis of Gaussian Fuzzy Strength Hyperparameter

To further investigate the effect of different fuzzy similarity levels on model performance, we conduct performance comparisons across different Gaussian fuzzy strength hyperparameters σ under the 1-shot setting on miniImageNet, tieredImageNet, and CIFAR-FS, respectively. The experiment results are presented in Fig. 5.

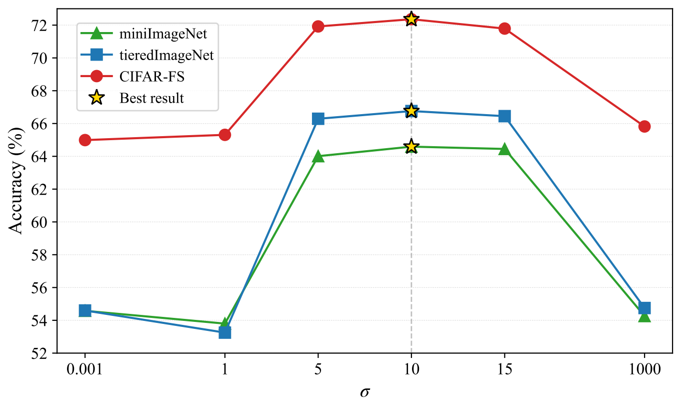


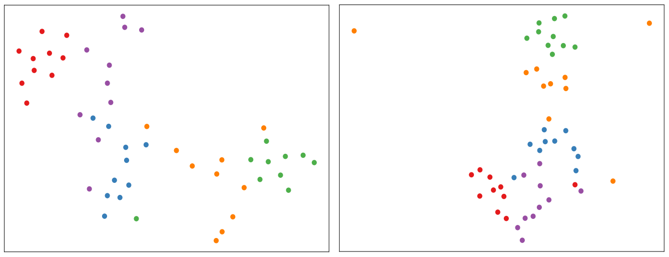
Fig. 5: Performance comparison of different Gaussian fuzzy strength hyperparameters (σ).

We observe that the performance of FCGNN severely decreased on three datasets when $\sigma = 0.001$ and $\sigma = 1000$. This allows us to infer that the performance of the model may greatly decline under limiting cases ($\sigma \to 0$, $\sigma \to \infty$). As σ approaches zero, the Gaussian similarity between different nodes converges to zero. We speculate that this hinders the model from sharing and utilizing category information across samples, leading to extreme under-smoothing that degrades model performance. As σ approaches infinity, the Gaussian similarity between different nodes converges toward 1. We speculate that this leads to node aggregation becoming a weighted average of node features, resulting in extreme over-smoothing that damages model performance. The experimental results indicate that FCGNN reaches the highest performance on three datasets at $\sigma = 10$. Therefore, we set $\sigma = 10$ for all datasets in our experiment.

E. Analysis of Inter-layer Node State Evolution

To illustrate the advantages of multi-layer node state awareness in FCGNN, we visualize the changes in node embeddings across different layers of both the Baseline and FCGNN using t-SNE. The visualizations of node embeddings for the input layer are displayed in Fig. 6, and the visualizations for the third to fourth layers are displayed in Fig. 7. From these visualizations, we can draw the following conclusions.

1) FCGNN employs a single-stage training model that optimizes both the graph network and the embedding network simultaneously. Despite utilizing a simple Conv4 embedding network, we find that FCGNN obtains a more compact initial node embedding, as shown in Figs. 6(a) and 6(b). This indicates that FCGNN, which considers node state changes across layers, is a more efficient model than the Baseline.

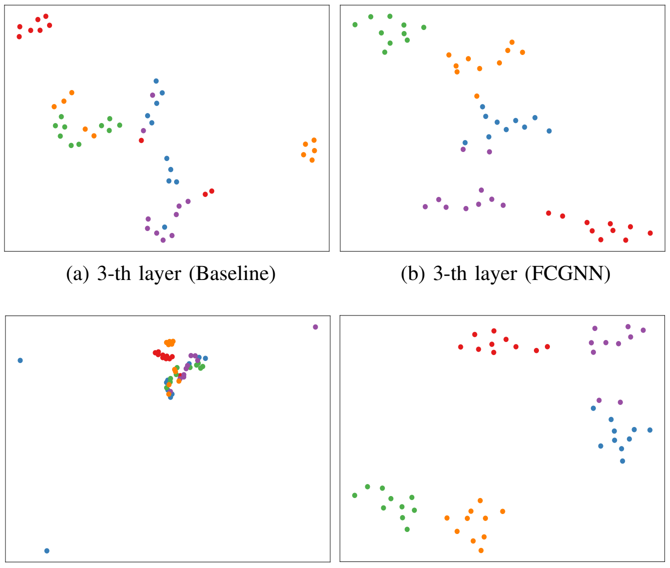


(a) Input (Baseline) (b) Input (FCGNN)

Fig. 6: The t-SNE visualization of input node embeddings for a 5-way 5-shot task with 10 query samples per category on the CUB test set. Different colors indicate different categories.

2) We observe that the baseline approach exhibits significant feature over-compression in the deeper network iterations. In the 3-th layer, nodes from different categories start to cluster together, as shown in Fig. 7(a). Conversely, the 4-th layer exhibits considerable overlap among the nodes, as shown in Fig. 7(c). We conclude that this is due to the traditional graph update approach not considering inter-layer changes in node states. Consequently, the abundant variability among node features is diminished due to excessive aggregation of neighboring information.

3) In contrast, FCGNN maintains clear category differentiation even after multiple aggregations. Specifically, the nodes from the 3-th to the 4-th layer demonstrate distinct category separation, as illustrated in Figs. 7(b) and 7(d). This suggests that FCGNN effectively mitigates the node feature over-compression observed in the baseline approach by dynamic modeling of inter-layer node state changes. We attribute this improvement to FCGNN ability to preserve the intrinsic features of nodes during multi-layer graph propagation, rather than overwriting or replacing the node information from previous layers in each layer.



(a) 3-th layer (Baseline) (b) 3-th layer (FCGNN)

(c) 4-th layer (Baseline) (d) 4-th layer (FCGNN)

Fig. 7: The t-SNE visualization of node embeddings in the 3-th and 4-th layer for a 5-way 5-shot task with 10 query samples per category on the CUB test set. Different colors indicate different categories.

F. Analysis of Computational Efficiency

In this subsection, we study the computational efficiency of FCGNN from two different perspectives: (1) Impact of parameter count across different models on performance. (2) Impact of different model components on computational cost. All experiments were conducted under the 1-shot setting on miniImageNet.

(1) Impact of Parameter Count across Different Models on Performance: To investigate the impact of the parameter count on model performance, we compare the performance and parameter count of FCGNN with several related graph models. The comparison results are presented in Fig. 8. From these comparative results, we observe that FCGNN shows significant performance improvement with a limited increase in parameter size. For example, FCGNN improves the accuracy by 2.21% with an increase of just 0.4M parameters over CSTS. In contrast, AGNN requires an additional 3.56M parameters compared with CSTS to achieve significant performance improvement, which is more than double the number of parameters in our method. This shows that despite the increased number of parameters caused by considering the fuzzy state changes of multilayer nodes, FCGNN maintains its efficiency due to the effect of the parameter-free fuzzy metric.

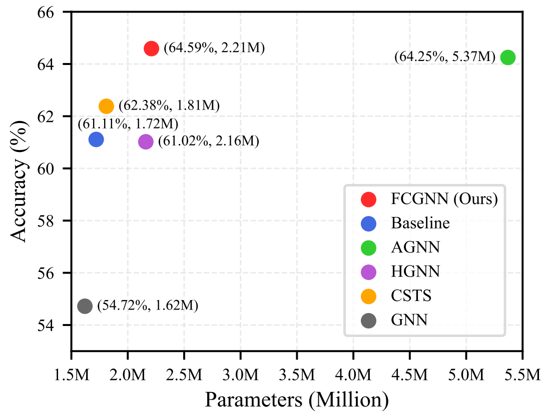


Fig. 8: Comparison of parameter count and performance on different models.

(2) Impact of Different Model Components on Computational Cost: To evaluate the computational cost of different modules, we compare the parameter count and runtime of different model variants, as shown in Table VII. Although introducing FCE alone increases the computational cost of the model, using Gaussian fuzzy similarity to construct a graph reduces the overall model parameter count and time complexity. We conclude that this is due to FCG replacing the original neural network metric with a parameter-free Gaussian similarity measure, which reduces the computational complexity of the model. Therefore, FCGNN improves the performance of FSL while maintaining reasonable computational costs.

TABLE VII
COMPARISON OF THE RUNTIME AND PARAMETER COUNT ON DIFFERENT MODEL VARIANTS.

Method	Parameters (Million)	Time (Second)
Baseline	1.722	0.371
Baseline + FCG	1.357	0.370
Baseline + FCE	3.449	0.392
Baseline + FCE + FCG (FCGNN)	2.215	0.387

G. Analysis of Network Depth and Over-smoothing

In this subsection, we first discuss the impact of graph network depth on model performance. Next, we introduce quantitative indicators to evaluate the effectiveness of FCGNN in mitigating the problem of over-smoothing. All experiments were conducted under the 1-shot setting on miniImageNet.

(1) Impact of Graph Network Depth: To investigate the impact of graph network depth on model performance, we compare the performance of FCGNN and the Baseline model across a range of depths from 2 to 12 layers, as shown in Fig. 9. We observe that the performance of the Baseline model first increases and then decreases as the network depth increases. We speculate that this performance degradation is caused by over-smoothing effects. In contrast, our FCGNN model maintains high and stable performance as it expands from 2 to 12 layers. This demonstrates our approach supports deeper graph network architectures. Furthermore, we adopt the 4-layer configuration for the main experiments. This network depth is designed to balance computational costs with maintaining competitive performance.

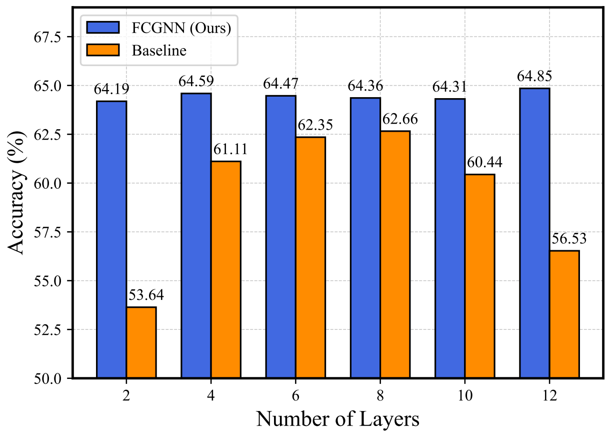


Fig. 9: Performance comparison of FCGNN and Baseline across different layers.

(2) Quantitative Analysis of Over-smoothing Problem: To quantitatively demonstrate that FCGNN mitigates the over-smoothing issue, we introduce node feature diversity as a quantitative evaluation metric, as shown in Fig. 10. Node feature diversity measures the degree of variation across all node feature representations in the graph, with its rapid decay indicative of over-smoothing [52]. The node feature diversity of the Baseline model exhibits a significant decline and instability as the number of layers increases. In contrast, our FCGNN model exhibits a slight decline while still outperforming the Baseline model. This shows that our approach uses the FCE module, which senses changes in node states, to counteract smoothing effects during information propagation. Therefore, our FCGNN model preserves the diversity of node features within deep architectures to maintain the high performance of the model.

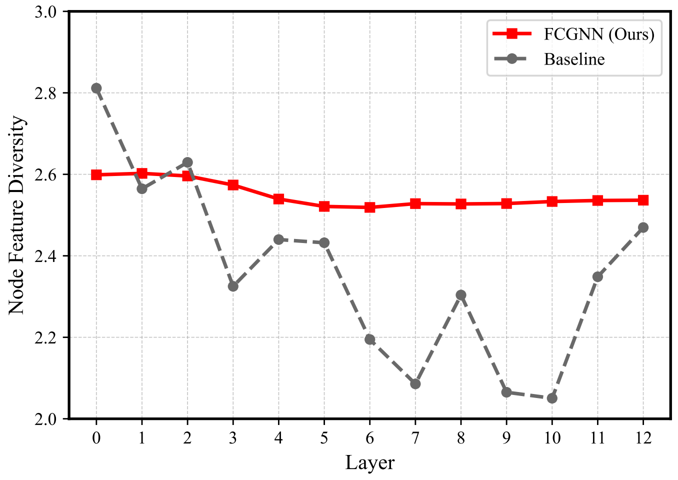


Fig. 10: Comparison of the changes in node feature diversity between FCGNN and the Baseline model.

V. CONCLUSION

This article proposes a fuzzy cognitive graph neural network (FCGNN) for few-shot learning, integrating the idea of concept updates in fuzzy cognitive maps with graph neural networks. Specifically, we propose constructing a fuzzy cognitive graph (FCG) based on Gaussian similarity to enhance intra-layer information flow. Meanwhile, FCGNN introduces a fuzzy concept evolution (FCE) module to dynamically capture the continuous evolution of node states across multiple layers. The complementary integration of FCG and FCE forms a complete fuzzy cognitive graph neural network, effectively mitigating issues of feature compression and information loss in deep graph networks. Although extensive experimental results on four benchmark datasets validated the effectiveness and superiority of FCGNN, determining the optimal fuzzy strength hyperparameter is time-consuming. In the future, we will determine suitable fuzzy strength hyperparameters based on the statistical properties of the data, such as the average, median, or standard deviation of node distances. Meanwhile, adaptive schemes for setting node-specific fuzzy strength hyperparameters will be investigated to enhance the adaptability and generalization capability of the model.

REFERENCES

[1] W. Li, Z. Wang, X. Yang, C. Dong, P. Tian, T. Qin, J. Huo, Y. Shi, L. Wang, Y. Gao, and J. Luo, "LibFewShot: A comprehensive library for few-shot learning," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 12, pp. 14 938–14 955, Dec. 2023.
[2] M. Hatano, R. Hachiuma, R. Fujii, and H. Saito, "Multimodal cross-domain few-shot learning for egocentric action recognition," in *Proc. Eur. Conf. Comput. Vis.*, 2025, pp. 182–199.
[3] B. M. Lake, R. Salakhutdinov, and J. B. Tenenbaum, "Human-level concept learning through probabilistic program induction," *Science*, vol. 350, no. 6266, pp. 1332–1338, 2015.
[4] A. A. Rusu, D. Rao, J. Sygnowski, O. Vinyals, R. Pascanu, S. Osindero, and R. Hadsell, "Meta-learning with latent embedding optimization," in *Proc. Int. Conf. Learn. Representations*, 2018, pp. 1–13.

[5] M. Jiang, J. Fan, J. He, W. Du, Y. Wang, and F. Li, “Contrastive prototype network with prototype augmentation for few-shot classification,” *Inf. Sci.*, vol. 686, 2025, Art. no. 121372.
[6] H. Gharoun, F. Momenifar, F. Chen, and A. H. Gandomi, “Meta-learning approaches for few-shot learning: A survey of recent advances,” *ACM Comput. Surv.*, vol. 56, no. 12, pp. 1–41, Jul. 2024.
[7] R. Zhang, T. Che, Z. Ghahramani, Y. Bengio, and Y. Song, “MetaGAN: An adversarial approach to few-shot learning,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 2371–2380.
[8] Y. Hong, L. Niu, J. Zhang, W. Zhao, C. Fu, and L. Zhang, “F2GAN: Fusing-and-filling gan for few-shot image generation,” in *Proc. 28th ACM Int. Conf. Multimedia*, 2020, pp. 2535–2543.
[9] B. Shi, W. Li, J. Huo, P. Zhu, L. Wang, and Y. Gao, “Global-and local-aware feature augmentation with semantic orthogonality for few-shot image classification,” *Pattern Recognit.*, vol. 142, 2023, Art. no. 109702.
[10] C. Finn, P. Abbeel, and S. Levine, “Model-agnostic meta-learning for fast adaptation of deep networks,” in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 1126–1135.
[11] N. Lai, M. Kan, C. Han, X. Song, and S. Shan, “Learning to learn adaptive classifier–predictor for few-shot learning,” *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 8, pp. 3458–3470, Aug. 2020.
[12] O. Vinyals, C. Blundell, T. Lillicrap, K. Kavukcuoglu, and D. Wierstra, “Matching networks for one shot learning,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 3630–3638.
[13] J. Snell, K. Swersky, and R. Zemel, “Prototypical networks for few-shot learning,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 4077–4087.
[14] F. Sung, Y. Yang, L. Zhang, T. Xiang, P. H. Torr, and T. M. Hospedales, “Learning to compare: Relation network for few-shot learning,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 1199–1208.
[15] Y. Huang, H. Hao, W. Ge, Y. Cao, M. Wu, C. Zhang, and J. Guo, “Relation fusion propagation network for transductive few-shot learning,” *Pattern Recognit.*, vol. 151, 2024, Art. no. 110367.
[16] X. Zhong, C. Gu, M. Ye, W. Huang, and C. W. Lin, “Graph complemented latent representation for few-shot image classification,” *IEEE Trans. Multimedia*, vol. 25, pp. 1979–1990, Jan. 2023.
[17] V. G. Satorras and J. B. Estrach, “Few-shot learning with graph neural networks,” in *Proc. Int. Conf. Learn. Representations*, 2018, pp. 1–13.
[18] P. Zheng, X. Guo, E. Chen, L. Qi, and L. Guan, “Edge-labeling based modified gated graph network for few-shot learning,” *Pattern Recognit.*, vol. 150, 2024, Art. no. 110264.
[19] Y. Liu, J. Lee, M. Park, S. Kim, E. Yang, S. J. Hwang, and Y. Yang, “Learning to propagate labels: Transductive propagation network for few-shot learning,” in *Proc. Int. Conf. Learn. Representations*, 2019, pp. 1–14.
[20] J. Kim, T. Kim, S. Kim, and C. D. Yoo, “Edge-labeling graph neural network for few-shot learning,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 11–20.
[21] C. Chen, K. Li, W. Wei, J. T. Zhou, and Z. Zeng, “Hierarchical graph neural networks for few-shot learning,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 32, no. 1, pp. 240–252, Jan. 2021.
[22] H. Zhao, Y. Su, Z. Wu, and W. Ding, “CSTS: Exploring class-specific and task-shared embedding representation for few-shot learning,” *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 36, no. 3, pp. 5721–5733, Mar. 2025.
[23] M. Dong, F. Li, Z. Li, and X. Liu, “PRSN: Prototype resynthesis network with cross-image semantic alignment for few-shot image classification,” *Pattern Recognit.*, vol. 159, 2025, Art. no. 111122.
[24] P. Zhao, Z. Ye, L. Wang, H. Liu, and X. Ji, “Multi-scale task-aware structure graph modeling for few-shot image recognition,” *Pattern Recognit.*, vol. 156, 2024, Art. no. 110855.
[25] J. Lu, P. Gong, J. Ye, J. Zhang, and C. Zhang, “A survey on machine learning from few samples,” *Pattern Recognit.*, vol. 139, 2023, Art. no. 109480.
[26] C. Zhang, Y. Cai, G. Lin, and C. Shen, “DeepEMD: Few-shot image classification with differentiable earth mover’s distance and structured classifiers,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 12 203–12 213.
[27] D. Kang, H. Kwon, J. Min, and M. Cho, “Relational embedding for few-shot classification,” in *Proc. IEEE Int. Conf. Comput. Vis.*, 2021, pp. 8822–8833.
[28] M. Gori, G. Monfardini, and F. Scarselli, “A new model for learning in graph domains,” in *Proc. Int. Jt. Conf. Neural Networks*, 2005, pp. 729–734.
[29] Y. Ma, S. Bai, S. An, W. Liu, A. Liu, X. Zhen, and X. Liu, “Transductive relation-propagation network for few-shot learning.” in *Proc. Int. Joint Conf. Artif. Intell.*, 2020, pp. 804–810.
[30] Y. Ma, S. Bai, W. Liu, S. Wang, Y. Yu, X. Bai, X. Liu, and M. Wang, “Transductive relation-propagation with decoupling training for few-shot learning,” *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 11, pp. 6652–6664, Nov. 2022.
[31] L. Yang, L. Li, Z. Zhang, X. Zhou, E. Zhou, and Y. Liu, “DPGN: Distribution propagation graph network for few-shot learning,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 13 390–13 399.
[32] T. Yu, S. He, Y. Z. Song, and T. Xiang, “Hybrid graph neural networks for few-shot learning,” in *Proc. AAAI Conf. Artif. Intell.*, 2022, pp. 3179–3187.
[33] H. Cheng, J. T. Zhou, W. P. Tay, and B. Wen, “Graph neural networks with triple attention for few-shot learning,” *IEEE Trans. Multimedia*, vol. 25, pp. 8225–8239, Jan. 2023.
[34] X. Guo, Y. Wang, T. Du, and Y. Wang, “ContraNorm: A contrastive learning perspective on oversmoothing and beyond,” in *Proc. Int. Conf. Learn. Representations*, 2023, pp. 1–12.
[35] J. Quan, F. Qiao, T. Yang, S. Shen, and Y. Qian, “A bi-selection method based on consistent matrix for large-scale datasets,” *IEEE Trans. Fuzzy Syst.*, early access, doi: 10.1109/TFUZZ.2025.3543893.
[36] Z. Liu, X. Zeng, J. Li, and F. Min, “Individual entity induced label concept set for classification: An information fusion viewpoint,” *Inf. Fusion*, vol. 111, 2024, Art. no. 102495.
[37] X. Liu, Y. Zhang, J. Qin, Z. Li, W. Liang, and Z. Li, “A review of fuzzy cognitive map learning algorithms and applications,” *Acta Automatica Sinica*, vol. 50, no. 3, pp. 450–474, 2024.
[38] J. Wang, Z. Peng, X. Wang, C. Li, and J. Wu, “Deep fuzzy cognitive maps for interpretable multivariate time series prediction,” *IEEE Trans. Fuzzy Syst.*, vol. 29, no. 9, pp. 2647–2660, Sept. 2021.
[39] W. Hoyos, J. Aguilar, and M. Toro, “PRV-FCM: An extension of fuzzy cognitive maps for prescriptive modeling,” *Expert Syst. Appl.*, vol. 231, 2023, Art. no. 120729.
[40] W. Hoyos, J. Aguilar, and M. Toro, “Federated learning approaches for fuzzy cognitive maps to support clinical decision-making in dengue,” *Eng. Appl. Artif. Intell.*, vol. 123, 2023, Art. no. 106371.
[41] W. Ding, T. Zhou, J. Huang, S. Jiang, T. Hou, and C. T. Lin, “FMDNN: A fuzzy-guided multi-granular deep neural network for histopathological image classification,” *IEEE Trans. Fuzzy Syst.*, vol. 32, no. 8, pp. 4709–4723, Aug. 2024.
[42] J. Wei, X. Zhang, W. Pedrycz, and W. Ding, “Dynamic fuzzy sampler for graph neural networks,” *IEEE Trans. Fuzzy Syst.*, vol. 33, no. 4, pp. 1357–1368, Apr. 2025.
[43] Y. Fan, Y. Wang, P. Zhu, L. Hui, J. Xie, and Q. Hu, “Uncertainty-aware superpoint graph transformer for weakly supervised 3d semantic segmentation,” *IEEE Trans. Fuzzy Syst.*, early access, doi: 10.1109/TFUZZ.2025.3543036.
[44] J. Deng, W. Dong, R. Socher, L. J. Li, K. Li, and F. F. Li, “ImageNet: A large-scale hierarchical image database,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2009, pp. 248–255.
[45] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie, “The caltech-ucsd birds-200-2011 dataset,” *California Inst. Technol.*, 2011.
[46] M. Ren, E. Triantafillou, S. Ravi, J. Snell, K. Swersky, J. B. Tenenbaum, H. Larochelle, and R. S. Zemel, “Meta-learning for semi-supervised few-shot classification,” in *Proc. Int. Conf. Learn. Representations*, 2018, pp. 1–15.
[47] L. Bertinetto, J. Henriques, P. Torr, and A. Vedaldi, “Meta-learning with differentiable closed-form solvers,” in *Proc. Int. Conf. Learn. Representations*, 2019, pp. 1–15.
[48] T. Liu, C. Wen, Q. Xiong, and J. Li, “Meta transfer evidence deep learning for trustworthy few-shot classification,” *Expert Syst. Appl.*, vol. 259, 2025, Art. no. 125371.
[49] X. Huang and S. H. Choi, “SAPENet: Self-attention based prototype enhancement network for few-shot learning,” *Pattern Recognit.*, vol. 135, 2023, Art. no. 109170.
[50] L. Zou, J. Jin, D. Li, and H. Zhao, “FSAKE: Few-shot graph learning via adaptive neighbor class knowledge embedding,” *Expert Syst. Appl.*, 2025, Art. no. 127868.
[51] M. Zhang, S. Huang, W. Li, and D. Wang, “Tree structure-aware few-shot image classification via hierarchical aggregation,” in *Proc. Eur. Conf. Comput. Vis.*, 2022, pp. 453–470.
[52] K. Han, Y. Wang, J. Guo, Y. Tang, and E. Wu, “Vision GNN: An image is worth graph of nodes,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2022, pp. 8291–8303.