Proceedings of Machine Learning Research **318** (2026)
The 39th Canadian Conference on Artificial Intelligence

Enhancing Few-shot Node Classification with High Order Graph Neural Networks

Md. Sirajum Munir Prince[†], Renata Dividino[†, *]
[†] Brock University, St Catharines, Ontario, Canada.

Abstract

Graph neural networks (GNNs) have achieved significant success in node classification tasks. However, their performance declines when trained on a few examples per class or when applied to unseen classes. Meta-learning tackles this problem by training models across many small learning tasks so that they can quickly adapt to new classes from limited data. In this setting, each task provides a small support set with a few labelled nodes per class, and the model is evaluated on a separate query set of unseen nodes from the same classes. However, applying meta-learning to graphs is particularly challenging due to the interconnected nature of graphs. Existing approaches often enrich tasks with additional contextual information or modify training objectives to better exploit neighbouring nodes and labels through supervised or self-supervised signals. However, the structural complexity of graphs makes it difficult to design stable and transferable tasks, as structural differences across tasks can lead to inconsistent feature representations. We argue that this limitation stems less from the meta-learning framework itself and more from the limited expressive power of standard GNNs. To overcome this, we leverage higher-order GNNs to generate richer node representations during both training and testing, improving the model's ability to generalize to new classes. Extensive experiments on multiple benchmark datasets demonstrate consistent improvements over state-of-the-art methods. The source code for this project is publicly available at https://github.com/sirajummprince/HIGH-META.

Keywords: Meta-learning, Few-shot Learning, Node Classification, High-Order Graph Neural Network

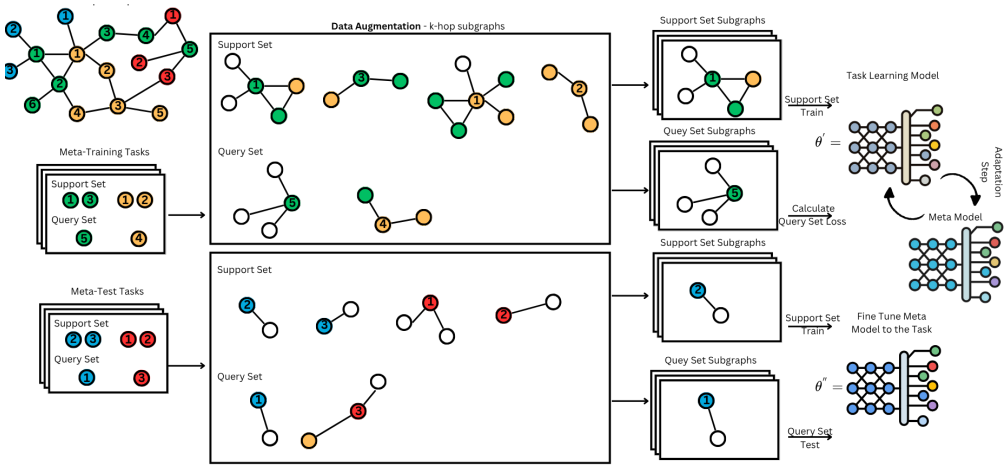
1. Introduction

Graph Neural Networks (GNNs) have emerged as a framework for processing graph-structured data, where nodes represent entities and edges capture their relationships. Through iterative message-passing operations, GNNs aggregate and transform neighbourhood information to generate low-dimensional node representations. These learned representations preserve both local structure and global connectivity patterns, enabling state-of-the-art performance across diverse tasks, including node classification, link prediction, and graph classification. However, GNNs typically require substantial amounts of labelled data to achieve strong performance. This creates significant challenges in practical scenarios where labelled samples are scarce. Consider a social network where nodes represent users and edges represent interactions. A GNN-based classification model might classify users into interest categories such as *sports enthusiasts* or *tech professionals* based on their profiles and connections. Standard GNN training would require thousands of labelled examples per category. But what if a new interest category emerges, such as *cryptocurrency traders*, with only a handful of labelled users? In this few-shot learning (FSL) scenario, the model must classify users into this category using only 2-5 labelled examples. Traditional GNNs struggle in such settings, as they cannot effectively learn discriminative features from so few samples.

Meta-learning addresses this challenge by training models to "learn how to learn" from limited data [1]. Instead of learning a single classification task, meta-learning trains on many related tasks, each with few labelled examples. The model learns general strategies

[*] rdividino@brocku.ca

Figure 1. Graph Meta-Learning Setup with Data Augmentation: meta-training and meta-test tasks are designed based on the training and test classes, respectively. In each task, $\mathcal{K}$ nodes are sampled from $\mathcal{N}$ classes to form the support and query sets. To enhance generalization, data augmentation is applied, such as k-hop subgraph extraction (e.g., for $k = 1$) for each node in the support and query set, which captures local structural information and enriches node representations.



for adapting to new tasks quickly. In the episodic training framework, the model repeatedly encounters training episodes that simulate few-shot scenarios. Each episode presents a mini-task: given a small support set of labelled nodes (e.g., 2 examples per class), classify unlabelled query nodes from the same classes. By training across many such episodes with different class combinations, the model learns to generalize from few examples. For graph-based FSL, existing meta-learning approaches [2–7] construct episodes from graph data. Figure 1 illustrates a 2-way 2-shot episode, i.e., 2 classes with 2 labelled examples per class. First, two classes are randomly sampled. For each class, 2 labelled nodes form the support set (4 nodes total), while additional nodes from the same classes form the query set (1 node per class). For each node, a subgraph around that node is extracted. The model trains on support set subgraphs and is evaluated on query set subgraphs. This process repeats across many episodes with varying graph structures, enabling the model to learn robust features that transfer to unseen classes. Recent methods [8–10] have improved graph-based FSL performance by designing sophisticated loss functions and self-supervised objectives. COSMIC [11] jointly optimizes node embeddings and class assignments at multiple granularities. COSMETA [12] combines data augmentation with contrastive learning (CL). Task-aware CL [13] models relationships between node representations and downstream tasks. Despite these advances, performance gains remain modest.

Meanwhile, higher-order graph neural networks (HOGNNs) [14] have emerged as a powerful approach for enhancing GNN expressiveness. Unlike standard GNNs that operate on pairwise edges, HOGNNs capture polyadic relationships involving multiple nodes simultaneously. This enables them to model complex multi-node interactions and address common GNN limitations such as over-smoothing and over-squashing. Subgraph-based HOGNNs [15–17] are particularly effective, as they perform message passing over collections of distinguished vertices and edges, capturing richer structural patterns.

We hypothesize that the limited performance of existing graph-based FSL methods stems not from meta-learning design limitations, but from the restricted expressiveness of traditional GNN architectures. While prior work has focused on improving meta-learning strategies, we address the problem at its foundation by enhancing the encoder itself. Our

approach employs HOGNNs to generate more expressive node representations during both training and testing. Specifically, we leverage two subgraph selection strategies that capture richer structural patterns in node neighbourhoods. This enhanced encoder is agnostic to the meta-learning framework, meaning it can complement any existing meta-learning approach. Experiments on three benchmark datasets (Cora, CiteSeer, and CoraFull) demonstrate consistent improvements over state-of-the-art methods.

2. Foundations

In this work a graph is represented as $G = (V, E, X, A, Y)$ with a set of nodes $v_1, \ldots, v_N \in V$ and edges $(v_i, v_j) \in E$. $X \in \mathbb{R}^{N \times n}$ is the feature matrix where X_i corresponds to the n-dimensional feature vector of node $v_i \in V$, and $A \in \{0, 1\}^{N \times N}$ is the adjacency matrix in which $A_{ij} = 1$ if there exists an edge between nodes v_i and v_j, and $A_{ij} = 0$ otherwise. $Y = \{y_1, \ldots, y_M\}$ denotes a set of M distinct node labels.

2.1. Graph Neural Networks

Graph Neural Networks (GNNs) learn low-dimensional node representations (i.e., embeddings) from graph-structured data for downstream tasks such as node classification, link prediction, and graph classification. Various architectures have been proposed [18–21] following the message-passing scheme. GNNs learn a multi-step mapping $f : G \to Z \in \mathbb{R}^{N \times n'}$, where Z is an updated feature matrix of graph G, to represent nodes and edges as low-dimensional vectors. Each step in the message-passing scheme consists of two phases: (aggregation) information about the neighbourhood of each node is gathered, and (update): the features of each node are updated based on its current and the aggregated neighbourhood information.

These steps are applied iteratively for each layer. The representation $\mathbf{h}_v^{(l)}$ of node v at the l-th layer is updated as: $\mathbf{h}_v^{(l)} = \sigma\left(\mathbf{B}_l \mathbf{h}_v^{(l-1)} + \mathbf{W}_l \sum_{u \in \mathcal{N}(v)} g(\mathbf{h}_u^{(l-1)})\right)$, where $\mathbf{h}_v^{(0)} = \mathbf{x}_v$, $\mathcal{N}(v)$ is the set of neighbours of v, and g is a differentiable, permutation-invariant neighbourhood aggregation function such as mean, sum, or max pooling. The features of node v and its neighbours are combined by the update function, which is a differentiable function that takes the aggregated embeddings as input and produces a new embedding for each node. The output of the final layer $\mathbf{Z}^{(k)}$ can then be used for downstream tasks.

Although GNNs achieve state-of-the-art performance on many tasks, prior work [21] shows that they are limited by the 1-WL (Weisfeiler–Lehman) test and cannot distinguish certain structurally distinct graphs with similar local neighbourhoods.

2.2. Higher-Order Graph Neural Networks

Higher-Order Graph Neural Networks (HOGNNs) [14] extend standard GNNs by incorporating higher-order structural information, enabling finer-grained discrimination beyond conventional message-passing architectures. In this work, we adopt the subgraph-based HOGNN paradigm, exemplified by models such as ESAN [15], SAGNN [17], and SUN [16], which construct node representations from selected graph substructures and aggregate them into unified embeddings. This approach contrasts with alternative higher-order methods, such as hypergraph neural networks, that rely on higher-dimensional topological constructs. The mathematical framework of subgraph-based HOGNNs extends traditional GNNs through three operations: subgraph generation, message passing, and aggregation.

Subgraph Generation: Given a graph $G = (V, E)$, the framework generates a collection of subgraphs $\mathcal{S} = \{S_i = (V_i, E_i) \mid i = 1, \ldots, k\}$, where each $S_i \subseteq G$. The subgraph selection process follows a policy $\mathcal{P}$ that can be implemented through various strategies: ego networks that include all vertices within k-hops of a center node, edge deletion approaches

that randomly remove edges to create sparser subgraphs, node deletion methods that sample node subsets and their induced subgraphs, motif-based selection focusing on specific structural patterns, or random walk approaches using stochastic traversal. Different policies capture complementary structural information, enabling the model to build multi-view representations. **Subgraph Message Passing:** Each subgraph S_i is embedded through a mapping $f : S_i \to Z_i \in \mathbb{R}^{N \times n'}$, where message passing operations aggregate information within the subgraph (see Section 2.1 for more details on the message passing mechanism). **Subgraph Aggregation:** The final node representation combines information from all subgraphs through an aggregation function $Z_G = \Phi(Z_1, \ldots, Z_k)$. This function Φ can be implemented through pooling operations, attention mechanisms that weight subgraph contributions, learnable neural networks, or concatenation followed by transformation. This architecture enables HOGNNs to construct sophisticated representations from local substructures, making them effective for tasks requiring fine-grained structural understanding.

2.3. **Few-shot Node Classification**

Given a graph G and the set of node labels Y, Y can be divided into two disjoint sets: the training classes Y_{tr} and the test classes Y_{ts} (such that $Y_{tr} \cap Y_{ts} = \emptyset$). Here, *classes* refers to the distinct categorical labels assigned to nodes (e.g., research topics in citation networks). It is to be assumed that there exist abundant labelled nodes in G for each class in Y_{tr} and extremely few labelled nodes for each class in Y_{ts}. The goal is to develop a classifier that can predict the categories of unlabelled nodes from node classes in Y_{ts} by transferring the knowledge obtained from Y_{tr} during the training stage, thus enabling the model to adapt to new classes (Y_{ts}) with limited labelled examples.

The meta-learning framework "learns how to learn" from few examples through a two-stage process: meta-training and meta-testing. During the meta-training phase, multiple learning episodes (or tasks) are constructed from the training classes Y_{tr}. In each episode, at first $\mathcal{N}$ classes from Y_{tr} are randomly sampled, then for each selected class, $\mathcal{K}$ labelled nodes are sampled to form the support set and additional nodes from the same classes to construct the query set. This episodic training paradigm ensures that the model repeatedly practices the exact scenario, which it will encounter during meta-testing: learning to adapt to new classes with limited examples. The support set in each episode serves as the few-shot training data, while the query set provides a way to evaluate, and optimize the model's adaptation capability. Through multiple episodes, the model learns generalizable strategies for feature extraction, and adaptation that can be transferred to novel classes in Y_{ts}.

Formally, let $\mathcal{T}_{tr} = \{T_1^{tr}, T_2^{tr}, ..., T_n^{tr}\}$ be a set of meta-training tasks, where each task T_i consists of a support set S_i^{tr} and a query set Q_i^{tr}. The meta-learning objective aims to find optimal meta model parameters θ that minimize the loss across all meta-training tasks $T_i \in \mathcal{T}_{tr}$, where each task contributes through the loss function $\mathcal{L}(f_{\theta_i}(Q_i); \theta)$. During meta-testing, the model's ability is evaluated to learn from few examples on entirely new classes from Y_{ts}. Let $\mathcal{T}_{ts} = \{T_1^{ts}, T_2^{ts}, ..., T_m^{ts}\}$ be a set of meta-test tasks, where each task T_i^{ts} consists of a support set S_i^{ts} and a query set Q_i^{ts}. Each support set S_i^{ts} contains $\mathcal{K}$ labelled examples per class for $\mathcal{N}$ test classes, and the model must leverage its meta-learned knowledge to quickly adapt, and make predictions on the unlabelled nodes in the query set.

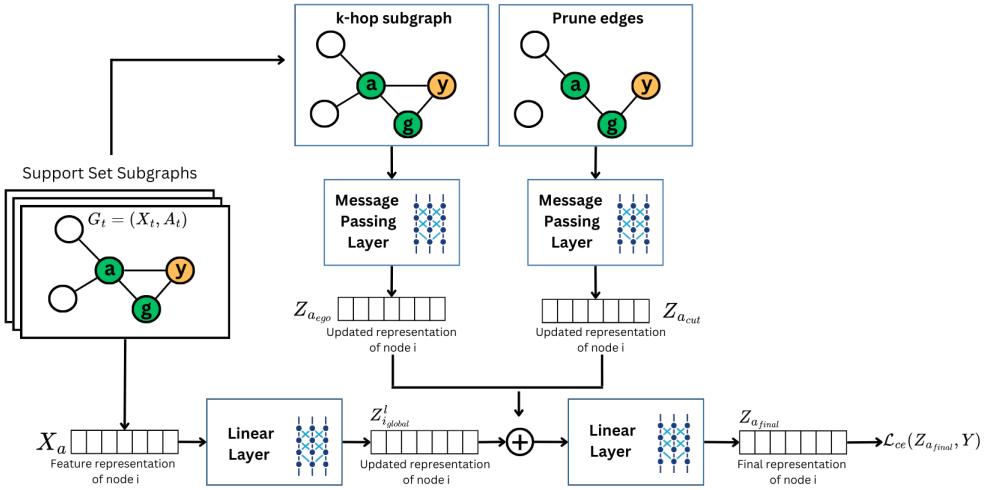
3. **Related Works**

Meta-GNN [22], the first framework to incorporate principles into GNNs for FSL node classification, leverages Model-Agnostic Meta-Learning (MAML) [1] to learn parameter initialization that can quickly adapt to new node classification tasks with only $\mathcal{K}$ labelled samples per class. An attribute-matching framework, AMM-GNN, based on MAML was

proposed in [7], using attribute-level attention to extract task-specific information on attributed networks. GPN [6], also designed for few-shot node classification on attributed networks, leverages prototypical learning. It implements an encoder that captures both node attributes and topological structure, and a node valuator that estimates the informativeness of each labelled node by considering its centrality and importance within the network. Similarly leveraging prototype learning, G-META's [3] core innovation is representing each node with its local subgraph. Prototypes computed from these embeddings guide adaptation via meta-gradients to improve cross-task knowledge transfer. Meta-GPS [5], proposed for heterophilic graphs, utilizes prototype-based initialization and scaling transformations. The framework adapts representations to varying task distributions using a network encoder and the S^2 transformation. X-FNC [23] generates pseudo-labelled nodes to address weak supervision in meta-training, using Poisson Label Propagation to enhance label propagation. The information bottleneck principle is applied to filter out irrelevant features, improving task adaptation. TENT [2] reduces distribution shifts between meta-training and meta-testing by incorporating node-level, class-level, and task-level adaptations. It reduces connectivity pattern variance by constructing subgraphs for each class using a virtual class node, ensuring structural coherence. TENT uses a task-adaptive matching strategy to preserve mutual information, enhancing task-level performance. GraphCEN [11] builds an affinity graph to model class relationships, leveraging both node- and class-level contrastive learning (CL) to jointly optimize node embeddings and class assignments. By combining class semantics with GNNs to create a joint matrix, it enables node-level CL in the row space for effective embeddings and class-level CL in the column space for class consistency, improving both representation learning and class assignment. Task-aware CL [13] enhance downstream task performance by maximizing mutual information between node representations and the downstream task (e.g., node classification). The core of Task-aware CL is the Task-Aware Contrastive Loss (XTCL), which uses the XGBoost sampler to select positive examples, optimizing XTCL for improved generalization. This method captures valuable graph signals in node representations, enhancing model performance. Similar, COSMETA [12] proposes a contrastive meta-learning framework which combines subgraph-based contrastive learning and a node-level cross-entropy objective, to learn robust and generalizable representations. This two-step optimization process encourages the learning of transferable features, enhancing the effectiveness of meta-learning when only a small amount of labelled data is available. I-GNN [24] was proposed to handle the observation that meta-learning approaches with graphs suffer significant performance degradation in the more realistic inductive setting. I-GNN freezes a pre-trained encoder and only fine-tunes a new linear classifier on FSL tasks. Finally, Pro-MC [4] uses proxy subgraphs to balance noisy and smooth subgraph biases, improving robustness and knowledge transfer. Pro-MC generates proxy subgraphs to calibrate manifold smoothness and enhance knowledge transfer.

Despite these advances, existing approaches have limitations. Gradient-based methods (Meta-GNN [22], AMM-GNN [7]) are sensitive to initialization and prone to overfitting; prototype-based methods (GPN [6], G-Meta [3]) use fixed aggregation schemes that may miss complex structures; and contrastive learning methods (COSMIC [8], COSMETA [12], Task-aware CL [13]) focus on loss design without improving the expressiveness of the GNN encoder, often introducing significant computational overhead. Our proposed model, HIGH-META, addresses these gaps by integrating HOGNNs into the meta-learning pipeline. Like related methods, it leverages local subgraphs around target nodes and follows the MAML framework, but instead of modifying subgraph elements, it encodes them through a HOGNN to produce richer, more expressive structural representations. This enhanced encoder is agnostic to the meta-learning strategy and can complement existing approaches by capturing more nuanced graph patterns.

Figure 2. HIGH-META overview. For each support and query node, two extraction policies generate complementary subgraph views: ego subgraphs preserve complete k-hop neighbourhoods, and cut subgraphs remove edges to create sparse structural views. Each subgraph undergoes parallel message passing to capture features at different structural granularities. A global encoder processes original features independently. The ego, cut, and global representations are concatenated and passed through a classification layer. This multi-view approach enhances encoder expressiveness, enabling effective learning from few labelled examples.



4. Methodology

We propose HIGH-META, a novel framework for FSL on graphs that leverages the enhanced expressiveness of subgraph HOGNNs. HIGH-META (see Figure 2) builds upon the substructure learning approach of SAGNN [17] and integrates it with the MAML protocol [1] through bi-level optimization.

4.1. HIGH-META Architecture

HIGH-META instantiates the subgraph-based HOGNN framework with specific subgraph generation policies and aggregation strategies designed for few-shot learning. For each meta-task $\mathcal{T} = \{S, Q\}$, the framework generates multiple subgraphs to provide multi-view representations for nodes in both support and query sets. Following the policy-based framework $\mathcal{P}$ described in Section 2, we employ two complementary subgraph extraction policies that capture different aspects of node neighborhoods:

Ego Subgraph Extraction Policy. This policy implements an ego network strategy where, for node i, the k-hop ego subgraph $G_i^{\text{ego}} = (V_i^{\text{ego}}, E_i^{\text{ego}}, X_i^{\text{ego}}, A_i^{\text{ego}}, Y_i^{\text{ego}})$ includes all nodes and edges within k hops (typically $k = 2$). This policy captures the complete local neighborhood structure around the target node, preserving all connectivity information within the specified radius. The aggregated ego feature is computed as:

$$\mathbf{x}_i^{\text{ego}} = \frac{1}{|V_i^{\text{ego}}|} \sum_{j \in V_i^{\text{ego}}} \mathbf{x}_j.$$

Cut Subgraph Extraction Policy. This policy implements an edge deletion strategy that creates a complementary structural view. Starting from the ego subgraph G_i^{ego}, we randomly remove a fraction p of edges (without replacement) to form a pruned graph $G_i^{\text{cut}} = (V_i^{\text{cut}}, E_i^{\text{cut}}, X_i^{\text{cut}}, A_i^{\text{cut}}, Y_i^{\text{cut}})$. This edge deletion policy serves two purposes: it

reduces connectivity to expose alternative structural paths, and it creates a more robust representation by simulating graph perturbations. Figure 2 illustrates the conceptual motivation for edge pruning. While various pruning strategies exist (weight-based, degree-based, random), we use uniform random edge dropping in our implementation. The cut subgraph feature is defined as:

$$\mathbf{x}_i^{\text{cut}} = \begin{cases} \dfrac{1}{|V_i^{\text{cut}}|} \displaystyle\sum_{j \in V_i^{\text{cut}}} \mathbf{x}_j, & \text{if } A_{i,j}^{\text{cut}} \neq \varnothing, \\ 0, & \text{otherwise.} \end{cases}$$

The combination of these two policies enables the model to capture complementary structural information: the ego policy provides a complete view of the local neighbourhood, while the cut policy reveals robust structural patterns that persist under edge perturbations. This multi-view approach, achieved through diverse policy selection, enhances the model's ability to learn discriminative features in few-shot scenarios where structural context is critical.

Subgraph Message Passing. Following the HOGNN framework, each subgraph representation is processed through a message passing layer that implements the mapping $f : S_i \to Z_i$:

$$\mathbf{Z}^{\text{ego}} = \text{MP}(\mathbf{Z}^{\text{ego}}, A), \quad \mathbf{Z}^{\text{cut}} = \text{MP}(\mathbf{Z}^{\text{cut}}, A).$$

Various message-passing architectures can be used, including GCN [19], GAT [20], GraphSage [18], and GIN [21].

Subgraph Aggregation. Following the aggregation framework $Z_G = \Phi(Z_1, \ldots, Z_k)$ from Section 2, we aggregate representations from multiple subgraphs. A global encoder applies a linear transformation to the original features: $\mathbf{Z}^{\text{global}} = \mathbf{W}_g \mathbf{X}^{\text{ego}} + \mathbf{b}_g$, where $\mathbf{W}_g \in \mathbb{R}^{h \times d^{\text{ego}}}$ and $\mathbf{b}_g \in \mathbb{R}^h$ are learnable parameters. The final node representation concatenates these encoded features:

$$\mathbf{Z}^{\text{concat}} = \left[\mathbf{Z}^{\text{ego}} \,\|\, \mathbf{Z}^{\text{cut}} \,\|\, \mathbf{Z}^{\text{global}}\right].$$

This concatenation implements the aggregation function Φ, combining information from different structural views generated by our two extraction policies.

The concatenated feature vector is passed through a fully connected layer to produce class logits: $\hat{\mathbf{Y}} = \mathbf{W}_f \mathbf{Z}^{\text{concat}} + \mathbf{b}_f \in \mathbb{R}^C$, where C is the number of classes. The final class probabilities are computed via softmax. The node embeddings are optimized for node classification via cross-entropy loss.

4.2. Meta-Learning Framework

Following the episodic training paradigm introduced in Section 2, HIGH-META is trained on a collection of meta-tasks $\mathcal{T}_{tr} = \{T_1^{tr}, \ldots, T_n^{tr}\}$ sampled from training classes Y_{tr}. Each task T_i^{tr} consists of a support set S_i^{tr} and a query set Q_i^{tr} under an $\mathcal{N}$-way $\mathcal{K}$-shot setting. Each meta-training episode is constructed by sampling $\mathcal{N}$ classes uniformly from Y_{tr}. For each class, $\mathcal{K}$ nodes are sampled for the support set and Q additional nodes for the query set. For every node in both sets, ego and cut subgraphs are extracted from the graph G.

We adopt the MAML [1] framework, which learns optimal initialization parameters θ^* via bi-level optimization: $\theta^* = \arg\min_\theta \sum_{T_i^{tr} \in \mathcal{T}_{tr}} \mathcal{L}_{query}(f_{\theta_i'}(Q_i^{tr}), y_{Q_i^{tr}})$, where f_θ denotes the HIGH-META model. The objective is to learn parameters that can rapidly adapt to new tasks after a small number of gradient updates on the support set.

The optimization consists of two nested loops: *Inner Loop (Task Adaptation):* For each task, task-specific parameters are obtained by gradient descent on the support set: $\theta_i' = \theta - \alpha \nabla_\theta \mathcal{L}_{support}(f_\theta(S_i^{tr}), y_{S_i^{tr}})$, where α is the inner-loop learning rate and $\mathcal{L}_{support}$ is the cross-entropy loss. *Outer Loop (Meta Update):* The meta-parameters are updated based on the query set performance after adaptation $\theta \leftarrow \theta - \beta \nabla_\theta \sum_{T_i^{tr} \in \mathcal{T}_{tr}} \mathcal{L}_{query}(f_{\theta_i'}(Q_i^{tr}), y_{Q_i^{tr}})$,

where β is the meta-learning rate. This bi-level structure enables learning initial parameters that facilitate fast adaptation across tasks.

During training, all extracted subgraphs are encoded using the HIGH-META architecture to obtain node representations. Inner-loop gradients are computed using the support sets, while the meta-loss is evaluated on the corresponding query sets.

At evaluation time, the learned meta-parameters θ^* are adapted to unseen tasks $\mathcal{T}_{ts}$ drawn from novel classes Y_{ts}, where $Y_{tr} \cap Y_{ts} = \emptyset$. For each test task T_j^{ts}, task-specific parameters θ_j' are obtained via inner-loop adaptation on S_j^{ts} and evaluated on Q_j^{ts}, demonstrating the model's ability to generalize to new classification tasks.

5. Evaluation

Datasets: In this study, we evaluate the proposed method on three benchmark graph datasets for node classification: Cora and CiteSeer [25], which are considered small- to medium-scale datasets, and CoraFull [26], a large-scale variant of Cora. The table below summarizes the key statistics of these datasets:

Dataset	Nodes	Edges	Classes	Features
Cora	2708	5429	7	1433
CiteSeer	3327	4552	6	3703
CoraFull	19793	126842	70	8710

Baselines: We compare the experimental results with those of the state-of-the- art FSL node classification methods, including MAML [1], ProtoNet [27], Meta-GNN [22], GPN [6], AMM-GNN [7], G-Meta [3], TENT [2], I-GNN [24].

Implementation Details: In our experiments, for the *Cut Subgraph Extraction Policy*, $p = 0.2$, meaning 20% of edges are randomly dropped. For episodic training, 2-hop subgraphs have been used along with $\mathcal{N}$-way $\mathcal{K}$-shot settings. In this regard, 2-way 1-shot, 2-way 3-shot, 2-way 5-shot, 5-way 1-shot & 5-way 5-shot settings have been used. The hyperparameters used for training and testing the GNN are provided in Table below:

Parameter	Value
Hidden Dimension	128
Number of Epochs	1000
Learning Rate	0.001
Weight Decay	1×10^{-5}
Number of Training Tasks	100 (Cora), 100 (CiteSeer), 1000 (CoraFull)
Number of Validation Tasks	20 (Cora), 20 (CiteSeer), 200 (CoraFull)
Number of Testing Tasks	40 (Cora), 40 (CiteSeer), 400 (CoraFull)
Batch Size	50
Negative Sampling (Training)	1–3

All experiments were conducted following configuration: Ubuntu 24.04.2 LTS, Linux kernel 6.11.0-19-generic, 13$^{\text{th}}$ Gen Intel Core i7-13700HX processor, 32 GiB RAM, and an NVIDIA GeForce RTX 4050 Laptop GPU (6 GB).

5.1. Results

Results are summarized in Table 1 across all datasets and FSL settings. In the 2-way 5-shot setting, HIGH-META outperforms prior models, achieving accuracies of 92.91% and 93.00% on Cora and CiteSeer, respectively. HIGH-META maintains robust performance across different graph structures. For instance, on the CoraFull dataset's 5-way settings, HIGH-META achieves 75.70% and 76.67% for 1-shot and 5-shot respectively, while the best baseline (G-Meta) reaches only 60.44% and 75.84%. TENT's relatively strong performance on CiteSeer (72.95% in 2-way 5-shot) compared to its weaker results on other datasets suggests that regularization or training strategies may be dataset-dependent. Similarly,

Table 1. Performance comparison (Mean Accuracy(%)) on the Cora, CiteSeer, and CoraFull datasets under various $\mathcal{N}$-way $\mathcal{K}$-shot FSL settings, as reported in [24, 28]. The best score is in bold and the second best is underlined. For CoraFull (70 classes), 5-way results are reported as this setting provides a more meaningful evaluation of discriminative capability compared to 2-way on a dataset with many classes. The 2-way results for CoraFull are: 1-shot: 98.45 ± 0.50%, 5-shot: 89.25 ± 2.83% (see Table 2 for the full breakdown).

Model	**Cora**		**CiteSeer**		**CoraFull**	
FSL Settings	2-way 1-shot	2-way 5-shot	2-way 1-shot	2-way 5-shot	5-way 1-shot	5-way 5-shot
MAML [1]	53.13 ± 2.26	57.39 ± 2.23	52.39 ± 2.20	54.13 ± 2.18	22.63 ± 1.19	27.21 ± 1.32
ProtoNet [27]	53.04 ± 2.36	57.92 ± 2.34	52.51 ± 2.44	55.69 ± 2.27	32.43 ± 1.61	51.54 ± 1.68
Meta-GNN [22]	65.27 ± 2.93	72.51 ± 1.91	56.14 ± 2.62	67.34 ± 2.10	55.33 ± 2.43	70.50 ± 2.02
GPN [6]	62.61 ± 2.71	76.39 ± 2.33	53.10 ± 2.39	63.09 ± 2.50	52.75 ± 2.32	72.82 ± 1.88
AMM-GNN [7]	65.23 ± 2.67	82.30 ± 2.07	54.53 ± 2.51	62.93 ± 2.42	58.77 ± 2.49	75.61 ± 1.78
G-Meta [3]	67.03 ± 3.22	80.05 ± 1.98	55.15 ± 2.68	64.53 ± 2.35	60.44 ± 2.48	75.84 ± 1.70
TENT [2]	53.05 ± 2.78	62.15 ± 2.13	62.75 ± 3.23	72.95 ± 2.13	55.44 ± 2.08	70.10 ± 1.73
I-GNN [24]	54.45 ± 3.13	65.18 ± 2.21	58.70 ± 3.17	65.60 ± 2.59	42.70 ± 1.92	51.46 ± 1.69
HIGH-META	**88.75 ± 4.08**	**92.91 ± 3.09**	**94.17 ± 1.17**	**93.00 ± 3.74**	**75.70 ± 1.33**	**76.67 ± 1.67**

Table 2. Additional performance analysis (Mean Accuracy ± std (%)) of HIGH-META on the Cora, CiteSeer and CoraFull datasets for various $\mathcal{N}$-way $\mathcal{K}$-shot FSL settings.

Dataset	**2-Way**			**3-Way**		**5-Way**	
	1-shot	**3-Shot**	**5-Shot**	**3-Shot**	**5-Shot**	**1-Shot**	**5-shot**
Cora	88.75 ± 4.08	88.75 ± 6.30	92.91 ± 3.09	88.43 ± 2.18	70.66 ± 1.22	61.50 ± 2.70	73.25 ± 2.10
CiteSeer	94.17 ± 1.17	91.11 ± 3.82	93.00 ± 3.74	87.69 ± 5.96	82.66 ± 5.89	57.80 ± 3.10	70.40 ± 2.65
CoraFull	98.45 ± 0.50	88.50 ± 3.49	89.25 ± 2.83	89.44 ± 3.70	81.00 ± 2.56	75.70 ± 1.33	76.67 ± 1.67

the performance differences between ProtoNet and Meta-GNN across datasets indicate that prototype-based versus gradient-based meta-learning approaches have varying effectiveness depending on the underlying graph characteristics. HIGH-META outperforms baselines, addressing FSL challenges that previous methods struggled with. In challenging low-data regimes, the model achieved significant accuracy gains compared to the baselines.

In extended FSL configurations (Table 2), HIGH-META achieves strong performance, with scores of 88.43% on Cora and 82.66% on CiteSeer in 3-way 5-shot settings. In 1-shot scenarios, HIGH-META performs particularly well on CoraFull (98.45%) and CiteSeer (94.17%). On Cora, performance remains relatively stable across 2-way configurations (88.75%). CoraFull demonstrates the most robust performance profile, maintaining relatively high accuracy even in challenging 5-way scenarios (75.70% and 76.67%). This may reflect CoraFull's richer feature representations or more distinct class boundaries that facilitate multi-class discrimination. The consistently low standard deviations across CoraFull experiments also indicate more stable learning dynamics on this dataset.

5.2. **Ablation Studies**

To evaluate the contribution of each architectural component of HIGH-META, we compare the full model with reduced variants. Table 3 reports HIGH-META's performance across subgraph configurations. Using only ego subgraph extraction policy restricts the model to local neighbourhood information, limiting access to higher-order relationships. Global-only configurations capture long-range dependencies but sacrifice local discriminative features, particularly hurting performance in 1-shot scenarios on Cora and CiteSeer. Cut-only subgraphs show intermediate performance, falling between ego-only and global-only baselines. Combined configurations consistently outperform single-subgraph approaches, though the optimal combination varies by dataset characteristics. For smaller, denser graphs

Table 3. Additional performance analysis (Mean Accuracy ± std (%)) of HIGH-META under different subgraph extraction strategies across various $\mathcal{N}$-way $\mathcal{K}$-shot FSL settings.

Dataset	Subgraph	2-Way			3-Way		5-Way	
		1-shot	3-shot	5-shot	3-shot	5-shot	1-shot	5-shot
Cora	Ego Only	53.10 ± 3.0	59.40 ± 1.40	64.10 ± 2.00	50.80 ± 1.80	59.00 ± 1.50	46.40 ± 2.10	52.50 ± 2.10
	Cut Only	50.20 ± 2.50	55.80 ± 1.60	58.90 ± 1.80	47.60 ± 2.20	55.20 ± 1.70	43.10 ± 1.80	49.80 ± 1.80
	Global Only	48.40 ± 2.20	51.70 ± 1.20	53.70 ± 1.00	44.70 ± 3.50	52.40 ± 1.80	40.60 ± 1.40	47.80 ± 1.00
	Ego + Global	**61.60 ± 1.10**	**68.30 ± 1.80**	**70.60 ± 1.60**	**57.30 ± 1.90**	**65.50 ± 1.20**	**52.30 ± 1.00**	**60.90 ± 2.60**
	Cut + Global	58.40 ± 1.40	64.50 ± 1.90	67.20 ± 1.70	54.10 ± 2.10	62.30 ± 1.40	49.50 ± 1.30	57.60 ± 2.30
CiteSeer	Ego Only	51.90 ± 1.30	59.70 ± 0.80	59.60 ± 2.80	48.70 ± 1.10	55.80 ± 2.40	42.50 ± 1.40	51.60 ± 1.70
	Cut Only	47.80 ± 1.80	54.30 ± 1.20	55.10 ± 2.40	44.90 ± 1.40	52.10 ± 2.00	38.90 ± 1.60	46.80 ± 1.90
	Global Only	43.60 ± 2.20	49.10 ± 1.40	50.20 ± 1.30	41.10 ± 0.70	48.90 ± 1.20	35.60 ± 1.90	42.00 ± 1.90
	Ego + Global	**56.50 ± 1.10**	**65.10 ± 1.60**	**66.00 ± 1.60**	**54.60 ± 3.70**	**62.70 ± 1.30**	**50.20 ± 0.60**	**57.70 ± 0.90**
	Cut + Global	53.20 ± 1.30	61.40 ± 1.70	62.50 ± 1.80	51.30 ± 3.20	59.40 ± 1.50	46.80 ± 0.80	54.20 ± 1.10
CoraFull	Ego Only	43.10 ± 2.00	45.50 ± 1.90	49.50 ± 2.30	40.20 ± 2.60	46.50 ± 1.30	36.20 ± 2.40	40.40 ± 1.70
	Cut Only	58.40 ± 1.80	64.20 ± 2.30	68.50 ± 1.60	54.80 ± 2.50	64.30 ± 1.70	48.70 ± 1.90	56.80 ± 1.40
	Global Only	69.10 ± 1.40	75.60 ± 2.60	79.20 ± 1.20	65.20 ± 2.30	75.70 ± 1.50	59.10 ± 1.40	67.30 ± 1.10
	Ego + Global	68.90 ± 4.20	76.20 ± 2.30	76.30 ± 3.90	66.00 ± 1.60	75.10 ± 3.00	55.40 ± 2.30	68.30 ± 2.70
	Cut + Global	**70.50 ± 3.60**	**77.80 ± 2.50**	**80.10 ± 3.20**	**67.40 ± 1.90**	**76.90 ± 2.70**	**58.20 ± 2.50**	**69.80 ± 2.40**

Table 4. Time (minutes) required to train on the Cora, CiteSeer and CoraFull datasets for various $\mathcal{N}$-way $\mathcal{K}$-shot FSL settings.

Training Time(minutes) Analysis(Average time after at least three runs)							
Dataset	**2-Way**			**3-Way**		**5-Way**	
	1-shot	**3-Shot**	**5-Shot**	**3-Shot**	**5-Shot**	**1-Shot**	**5-shot**
Cora	12.96	32.40	51.84	17.35	70.00	32.14	32.17
CiteSeer	34.63	86.58	138.52	46.81	187.00	34.25	343.53
CoraFull	826.95	2067.25	3307.60	2790.79	4465.26	2050.71	8202.85

(Cora and CiteSeer), Ego + Global achieves the best results, delivering strong gains in challenging 1-shot and 5-way settings. For the larger, sparser CoraFull dataset, Cut + Global proves most effective, with substantial improvements across all FSL settings. These results validate HIGH-META's core design: combining complementary structural views, whether local ego networks or cut-based partitions, with global topology yields more discriminative embeddings for FSL.

5.3. **Theoretical Complexity and Empirical Runtime Analysis**

Table 4 provides a view of HIGH-META training time across a range of FSL settings: training times range from 13 minutes (Cora, 2-way 1-shot) to 136 hours (CoraFull, 5-way 5-shot). The computational complexity of HIGH-META can be analyzed as follows. For each node, the ego subgraph extraction requires $O(d^k)$ time where d is the average node degree and k is the hop count. The cut subgraph extraction adds $O(|E_{ego}|)$ for random edge removal. Given P extraction policies (P=2 in our case: ego and cut), the total encoding cost per node is $O(P \times L \times |E_{sub}| \times h)$, where L is the number of message-passing layers, $|E_{sub}|$ is the average number of edges per subgraph, and h is the hidden dimension of the feature matrix. Compared to standard GNN-based meta-learning methods that use a single subgraph view, HIGH-META incurs a factor of approximately $P \times overhead$.

6. **Limitations**

While HIGH-META demonstrates strong performance on citation networks, several limitations needs to be discussed. First, all three benchmark datasets (Cora, CiteSeer, and CoraFull) are citation networks with similar structural properties, such as relatively high homophily ratios and sparse connectivity patterns. Consequently, the generalizability as

well as performance of the proposed approach to structurally different domains such as biological interaction networks, where adjacent nodes may exhibit greater label heterogeneity, or social networks with denser and more heterogeneous connectivity remains an open question. Second, the computational overhead introduced by multiple subgraph extractions and parallel message-passing channels increases training time substantially, particularly on larger graphs (e.g., CoraFull requires over 136 hours for 5-way 5-shot). This limits practical applicability to very large-scale graphs without further optimization. Third, the proposed subgraph extraction policies (ego and cut) use fixed strategies; adaptive or learned extraction mechanisms could potentially yield better task-specific representations.

7. Conclusion

This paper presents HIGH-META, a novel few-shot node classification framework that leverages higher-order GNNs to address the limited expressiveness of traditional GNN architectures. HIGH-META employs two complementary subgraph extraction policies: ego subgraphs that capture complete local neighbourhoods and cut subgraphs that remove a fraction of edges to expose robust structural patterns. This multi-view approach enables the learning of transferable representations that generalize effectively under minimal labelled data. Extensive experiments on benchmark citation networks (Cora, CiteSeer, and CoraFull) demonstrate that HIGH-META outperforms existing baselines across multiple FSL configurations. The approach achieves particularly strong performance in challenging 2-way and 3-way settings by capturing both local structural patterns and global graph topology through its hierarchical subgraph extraction strategy. Future work will address scalability through sparse sampling and batch-parallel extraction, evaluate performance on diverse homogeneous and heterogeneous graph benchmarks, and develop learnable subgraph generation mechanisms that adaptively select informative structures for specific tasks with particular attention to low-homophily and heterogeneous settings.

References

[1] C. Finn, P. Abbeel, and S. Levine. "Model-agnostic meta-learning for fast adaptation of deep networks". In: *International conference on machine learning*. PMLR. 2017, pp. 1126–1135.

[2] S. Wang, K. Ding, C. Zhang, C. Chen, and J. Li. "Task-adaptive few-shot node classification". In: *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 2022, pp. 1910–1919.

[3] K. Huang and M. Zitnik. "Graph meta learning via local subgraphs". In: *Advances in neural information processing systems* 33 (2020), pp. 5862–5874.

[4] Z. Wang, L. Cao, W. Lin, M. Jiang, and K. C. Tan. "Robust graph meta-learning via manifold calibration with proxy subgraphs". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 37. 12. 2023, pp. 15224–15232.

[5] Y. Liu, M. Li, X. Li, F. Giunchiglia, X. Feng, and R. Guan. "Few-shot node classification on attributed networks with graph meta-learning". In: *Proceedings of the 45th international ACM SIGIR conference on research and development in information retrieval*. 2022, pp. 471–481.

[6] K. Ding, J. Wang, J. Li, K. Shu, C. Liu, and H. Liu. "Graph prototypical networks for few-shot learning on attributed networks". In: *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 2020, pp. 295–304.

[7] N. Wang, M. Luo, K. Ding, L. Zhang, J. Li, and Q. Zheng. "Graph few-shot learning with attribute matching". In: *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 2020, pp. 1545–1554.

[8] S. Wang, Z. Tan, H. Liu, and J. Li. "Contrastive meta-learning for few-shot node classification". In: *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 2023, pp. 2386–2397.

[9] H. Liu, J. Feng, L. Kong, D. Tao, Y. Chen, and M. Zhang. "Graph Contrastive Learning Meets Graph Meta Learning: A Unified Method for Few-shot Node Tasks". In: *Proceedings of the ACM on Web Conference 2024*. 2024, pp. 365–376.

[10] F. Zhao and D. Wang. "Multimodal graph meta contrastive learning". In: *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. 2021, pp. 3657–3661.

[11] W. Ju, Y. Qin, S. Yi, Z. Mao, K. Zheng, L. Liu, X. Luo, and M. Zhang. "Zero-shot node classification with graph contrastive embedding network". In: *Transactions on Machine Learning Research* (2023).

[12] M. S. M. Prince and R. Dividino. "COS-META: Enhancing Few-shot Node Classification with Contrastive Meta-Learning". In: *Proceedings of the 17th International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2025)*. 2025, pp. 16–31.

[13] Y.-C. Lin and J. Neville. "Improving Node Representation by Boosting Target-Aware Contrastive Loss". In: *arXiv preprint arXiv:2410.03901* (2024).

[14] C. Morris, M. Ritzert, M. Fey, W. L. Hamilton, J. E. Lenssen, G. Rattan, and M. Grohe. "Weisfeiler and leman go neural: Higher-order graph neural networks". In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 33. 01. 2019, pp. 4602–4609.

[15] B. Bevilacqua, F. Frasca, D. Lim, B. Srinivasan, C. Cai, G. Balamurugan, M. M. Bronstein, and H. Maron. "Equivariant Subgraph Aggregation Networks". In: *International Conference on Learning Representations*. 2022.

[16] F. Frasca, B. Bevilacqua, M. Bronstein, and H. Maron. "Understanding and extending subgraph GNNs by rethinking their symmetries". In: *Advances in Neural Information Processing Systems* 35 (2022), pp. 31376–31390.

[17] D. Zeng, W. Liu, W. Chen, L. Zhou, M. Zhang, and H. Qu. "Substructure aware graph neural networks". In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 37. 9. 2023, pp. 11129–11137.

[18] W. Hamilton, R. Ying, and J. Leskovec. "Inductive representation learning on large graphs". In: *Advances in neural information processing systems* 30 (2017).

[19] T. N. Kipf and M. Welling. "Semi-Supervised Classification with Graph Convolutional Networks". In: *International Conference on Learning Representations*. 2017.

[20] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio. "Graph Attention Networks". In: *International Conference on Learning Representations*. 2018.

[21] K. Xu, W. Hu, J. Leskovec, and S. Jegelka. "How Powerful are Graph Neural Networks?" In: *International Conference on Learning Representations*. 2018.

[22] F. Zhou, C. Cao, K. Zhang, G. Trajcevski, T. Zhong, and J. Geng. "Meta-GNN: On few-shot node classification in graph meta-learning". In: *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*. 2019, pp. 2357–2360.

[23] S. Wang, Y. Dong, K. Ding, C. Chen, and J. Li. "Few-shot node classification with extremely weak supervision". In: *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining*. 2023, pp. 276–284.

[24] H. Mathavan, Z. Tan, N. Mudiam, and H. Liu. "Inductive linear probing for few-shot node classification". In: *International Conference on Social Computing, Behavioral-Cultural Modeling and Prediction and Behavior Representation in Modeling and Simulation*. Springer. 2023, pp. 274–284.

[25] Z. Yang, W. Cohen, and R. Salakhudinov. "Revisiting semi-supervised learning with graph embeddings". In: *International Conference on Machine Learning*. PMLR. 2016, pp. 40–48.

[26] A. Bojchevski and S. Günnemann. "Deep gaussian embedding of graphs: Unsupervised inductive learning via ranking". In: *arXiv preprint arXiv:1707.03815* (2017).

[27] J. Snell, K. Swersky, and R. Zemel. "Prototypical networks for few-shot learning". In: *Advances in neural information processing systems* 30 (2017).

[28] Z. Tan, S. Wang, K. Ding, J. Li, and H. Liu. "Transductive linear probing: A novel framework for few-shot node classification". In: *Learning on Graphs Conference*. PMLR. 2022, pp. 4–1.