Robust Graph Meta-learning for Weakly-supervised Few-shot Node Classification

KAIZE DING, Northwestern University, USA
JIANLING WANG, Texas A&M University, USA
JUNDONG LI, University of Virginia, USA
JAMES CAVERLEE, Texas A&M University, USA
HUAN LIU, Arizona State University, USA

Graph machine learning (Graph ML) models typically require abundant labeled instances to provide sufficient supervision signals, which is commonly infeasible in real-world scenarios since labeled data for newly emerged concepts (e.g., new categorizations of nodes) on graphs is rather limited. In order to efficiently learn with a small amount of data on graphs, meta-learning has been investigated in graph ML. By transferring the knowledge learned from previous experiences to new tasks, graph meta-learning approaches have demonstrated promising performance on few-shot graph learning problems. However, most existing efforts predominately assume that all the data from the seen classes is gold-labeled, while those methods may lose their efficacy when the seen data is weakly-labeled with severe label noise. As such, we aim to investigate a novel problem of weakly-supervised graph meta-learning for improving the model robustness in terms of knowledge transfer. To achieve this goal, we propose a new graph meta-learning framework – Graph Interpolation Networks (Meta-GIN). Based on a new robustness-enhanced episodic training paradigm, Meta-GIN is meta-learned to interpolate node representations from weakly-labeled data and extracts highly transferable meta-knowledge, which enables the model to quickly adapt to unseen tasks with few labeled instances. Extensive experiments demonstrate the superiority of Meta-GIN over existing graph meta-learning studies on the task of weakly-supervised few-shot node classification.

CCS Concepts: • **Computing methodologies** → **Neural networks**; • **Information systems** → **Web mining**.

Additional Key Words and Phrases: Graph neural networks, few-shot learning, weak supervision, noisy labels

1 INTRODUCTION

Graphs serve as a common language for modeling a plethora of structured and relational systems, such as social networks [31], knowledge graphs [43], and academic graphs [16, 38]. Many central tasks in graph machine learning (ML), such as node classification [18, 45], link prediction [10, 12], and community detection [9, 52], have received much research attention due to their significant impacts in addressing real-world problems [13, 20, 22, 30, 41]. To harness the inherent structure among data, significant methodological advances have been made in Graph ML, which have produced promising results in applications from diverse domains [11, 18, 39].

Many prevailing Graph ML methods typically rely upon the availability of sufficient labeled data [5, 6, 54]. In contrast, the long-tail property of real-world graphs makes those methods less effective for learning new concepts

Authors' addresses: Kaize Ding, Northwestern University, Evanston, Illinois, USA, kaize.ding@northwestern.edu; Jianling Wang, Texas A&M University, College Station, Texas, USA, jlwang@tamu.edu; Jundong Li, University of Virginia, Charlottesville, Virginia, USA, jundong@virginia.edu; James Caverlee, Texas A&M University, College Station, Texas, USA, caverlee@tamu.edu; Huan Liu, Arizona State University, Tempe, Arizona, USA, huan.liu@asu.edu.

1556-4681/2023/11-ART $15.00
https://doi.org/10.1145/3630260

when only limited labeled data is available [1, 5, 6, 16]. A powerful Graph ML model should be able to quickly learn never-before-seen class labels using only a handful of labeled data. Dealing with such *few-shot* unseen concepts[1] is important and corresponds to many practical applications. For example, many social networking platforms such as Facebook and Twitter need to consistently promote new features or new social media groups to users. Based on the limited user interactions, the deployed model is required to provide high-quality recommendations for other users regarding these new concepts. Inspired by the recent success of meta-learning in image domain [36], increasing research efforts have been made in graph meta-learning [16, 19, 25, 54] for solving the problem of few-shot learning on graph-structured data. In essence, a meta-learning model learns across diverse *meta-training* tasks sampled from those *seen classes* with a large quantity of labeled data, and can be naturally generalized to a new task (i.e., *meta-test* task) with *unseen classes* during training [36]. Such a *learning-to-learn* procedure enables the model to accumulate knowledge from previous experiences, and has led to significant progress in few-shot learning (FSL) problems. Following this learning paradigm, researchers have proposed to use graph neural networks (GNNs) as the backbone to extrapolate meta-knowledge on graphs, which demonstrates promising results [5, 16, 42, 54] on graph few-shot learning problems.

Despite some exciting progress, the research of graph meta-learning overall remains in its infancy. In general, existing efforts such as Meta-GNN [54] and GPN [5] predominately focus on the *supervised* setting, where abundant gold-labeled nodes can be accessed from the seen classes during the meta-training process. This assumption is often infeasible since collecting such auxiliary knowledge is laborious and requires intensive domain-knowledge, especially when considering the heterogeneity of graph-structured data. An alternative solution is to adopt automatic labeling tools based on heuristics, crowd-sourcing, or weak-learners [53]. Though using such *weakly-labeled* data is more practical, one companion issue is that such data usually contains a significant amount of *label noise*. As shown in the previous research studies [33, 49], most of the existing few-shot learning (FSL) models are highly vulnerable to noise or outliers, thus the model performance on unseen tasks would be largely degraded if the model is meta-learned on those weakly-labeled data. Therefore, it is imperative to investigate the problem of *weakly-supervised* few-shot node classification, in order to push forward the frontiers of robust graph meta-learning.

As a research problem has been little explored, weakly-supervised few-shot node classification is challenging to solve, mainly due to the difficulty of *extracting highly transferable meta-knowledge from weakly-labeled base node classes*: **(i)** on the one hand, the existing literature of learning with noisy labels is tailored for independent and identically distributed (*i.i.d.*) data such as image and text. The inability of modeling relational data like graphs that lie in non-Euclidean space could largely jeopardize their effectiveness. Hence, it is necessary to develop new frameworks that can consider the dependencies among nodes and mitigate the inaccurate supervision signals; **(ii)** on the other hand, in order to extract meta-knowledge during the meta-training process, graph meta-learning models will be trained with diverse node classification tasks from disjoint label spaces. It requires the model to quickly adapt to never-before-seen labels, which poses great challenges to existing denoising algorithms since they are optimized in a static learning environment (i.e., a single task). Hence, how to bridge the gap between the two learning paradigms and design a graph meta-learning model that can effectively denoise and efficiently adapt to new task (i.e., *meta-test* task) with *unseen classes* is vital to be explored.

To address the aforementioned challenges, we present a new robust graph meta-learning framework – Meta Graph Interpolation Network (Meta-GIN) in this paper. To mitigate the label noise issue during meta-training, we first propose a simple yet effective *robustness-enhanced episodic training* paradigm based on the idea of task interpolation. Instead of directly learning the model from the noisy meta-training tasks, Meta-GIN meta-learns from a pool of *noisy-reduced tasks*, each of which is interpolated from a set of weakly-labeled meta-training tasks. Specifically, Meta-GIN first randomly samples a set of weakly-labeled tasks sharing the same label space from

[1]In this paper, we primarily focus on the task of few-shot node classification on attributed graphs.

base classes and meanwhile learn expressive node representations capturing both node attributes and topological structure information. Afterwards, the Graph Interpolation Network compares the node representations with the same index in different tasks and estimates a confidence score for each node. Based on the computed confidence scores, Meta-GIN further performs interpolation to get a noise-reduced representation for the corresponding class. With learning across a pool of those interpolated tasks, Meta-GIN model is able to not only denoise from weakly-labeled training data, but also extrapolate the knowledge from seen to unseen node classes. This way the learned Meta-GIN can quickly adapt to new tasks using few gold-labeled samples. In summary, our main contributions are:

- ***Problem***: We investigate a new problem – weakly-supervised node classification, which has not been addressed by existing graph meta-learning methods and our work pushs forward the frontiers of few-shot learning on graphs.
- ***Algorithm***: We propose a principled framework Meta-GIN that is capable of extracting highly transferable meta-knowledge from weakly-labeled data, in order to solve unseen node classification tasks with few labeled nodes.
- ***Evaluation***: We perform extensive experiments on various real-world datasets to corroborate the effectiveness of our approach. The experimental results demonstrate the superior performance of Meta-GIN over existing efforts.

2 RELATED WORK

Meta-Learning. Meta-learning, also known as learning-to-learn, has been widely used in various domains to address few-shot learning problems. Generally, existing meta-learning methods follow the episodic training paradigm, and fall into three broad categories: *metric-based* [23, 33, 36, 37, 40], *optimization-based* [8, 21, 29, 32], and *memory-based* [35, 55]. Metric-based approaches try to learn generalizable matching metrics between query and support set across different tasks. For example, Prototypical Networks [36] learn a matching metric by taking the mean of support sets as prototypes, and classify query instances by calculating their Euclidean distances to the computed prototypes. Relation Network [37] trains an auxiliary network to learn a non-linear metric between each query and the support set; Optimization-based approaches focus on learning an initialization of model parameters that can quickly adapt to new tasks by fine-tuning with few labeled examples. For example, in [32], Meta-Learner LSTM interprets stochastic gradient descent update rules as a gated recursive model with trainable parameters, in order to learn the update rules of model parameters. MAML [8] seeks a proper parameter initialization by second-order gradient descent method so that the model can achieve better generalization performance after a few steps of gradient descent. Meta-SGD [21] goes further in meta-learning by arguing to learn the weights initialization, gradient update direction and learning rate within a single step. Memory-based approaches use the memory mechanism to extract valuable knowledge acquired in the meta-training phase to assist meta-testing. For instance, MANN [35] learns new tasks by reminiscence mechanism in virtue of physical memory, CMN [55] uses the key-value memory network paradigm to obtain an optimal video representation in a larger space.

Graph Few-shot Learning. Graph few-shot learning is a challenging research problem targeting a setting where the unseen concepts during the testing phase only have a few labeled instances [5, 54]. Traditional machine learning approaches [50, 51] cannot be directly applied to solve this problem due to the non-i.i.d. property of graph-structured data. Meanwhile, the effectiveness of powerful Graph ML methods such as GNNs largely relies on sufficient labeled instances and they also fail to address graph few-shot learning problems. Among recent advances of few-shot learning on graphs, a major line of work aims to solve the task of node classification [5, 16, 19, 25, 26, 42, 44, 54]. Among them, Meta-GNN, GPN and G-Meta are three representative ones. Specifically, Meta-GNN [54] directly applies the idea of gradient-based meta-learning to optimize a GNN

Symbols	Definitions
$\mathbf{G} = (\mathbf{A}, \mathbf{X})$	the input attributed graph with node, edge, and
v_i	the node with index i
$\mathbf{A}$	the adjacency matrix of graph G
$\tilde{\mathbf{A}}$	normalized adjacency matrix
$\mathbf{X}$	the feature matrix of graph G
$\mathbf{x}_i$	the node feature vector of node v_i
$\mathcal{Y}$	the node label set
$\mathbf{W}$	the weight matrix
$\mathbf{Z}$	the l-th layer embedding matrix
$\mathbf{h}_i^l$	the l-th layer embedding of node v_i
$\mathcal{T}_t = \{\mathcal{S}_t, \mathcal{Q}_t\}$	the t-th task with the corresponding support set and query set
$v_{t,k}$	the k-th sampled node in task $\mathcal{T}_t$
$\mathcal{V}_k$	The k-th weakly-labeled node set from M tasks
$\mathbf{p}_k$	prototype representation of the nodes in $\mathcal{V}_k$
$\mathbf{c}_k$	interpolated representation learned from $\mathcal{V}_k$
$\mathcal{T}_i' = \{\mathcal{S}', \mathcal{Q}'\}$	the i-th interpolated task with the corresponding support set and query set

Table 1. Table of main symbols.

model for few-shot node classification. GPN [5] extends prototypical networks to graph-structured data and learns expressive node class prototypes by considering the importance of each node. G-META [16] uses local subgraphs to transfer subgraph-specific information for enabling graph few-shot learning. Note that other methods such as GFL [47] and MetaTNE [19] are different from our scenario since they are focusing multiple networks and plain networks, respectively. In addition to few-shot node classification, other Graph ML tasks including graph classification [2, 27] and link prediction [1, 48] have also been studied under the few-shot learning setting. Unlike previous works, we propose a weakly-supervised graph meta-learning framework, which eliminates the dependency on gold-labeled data during the meta-training phase.

3 PRELIMINARIES

Following the commonly used notations, in this paper, we use calligraphic fonts, bold lowercase letters, and bold uppercase letters to denote sets (e.g., $\mathcal{V}$), vectors(e.g., $\mathbf{x}$), and matrices (e.g., $\mathbf{X}$), respectively. The i^{th} row of a matrix $\mathbf{X}$ is denoted by $\mathbf{x}_i$, and the transpose of a matrix $\mathbf{X}$ is represented as $\mathbf{X}^{\mathrm{T}}$. For other special notations, we will illustrate them in the corresponding sections.

Formally, consider an attributed graph $\mathbf{G} = (\mathcal{V}, \mathcal{E}, \mathbf{X})$, where $\mathcal{V}$ denotes the set of nodes $\{v_1, v_2, \dots, v_n\}$ and $\mathcal{E}$ denotes the set of edges $\{e_1, e_2, \dots, e_m\}$. Each node is associated with an attribute vector $\mathbf{x}_i \in \mathbb{R}^d$ and $\mathbf{X} = [\mathbf{x}_1^{\mathrm{T}}; \mathbf{x}_2^{\mathrm{T}}; \dots; \mathbf{x}_n^{\mathrm{T}}] \in \mathbb{R}^{n \times d}$ denotes all the node features. Concretely, we represent the attributed graph as $\mathbf{G} = (\mathbf{A}, \mathbf{X})$, where $\mathbf{A} = \{0, 1\}^{n \times n}$ is an adjacency matrix representing the network structure. Accordingly, we formulate the studied problem as follows:

Problem Definition 1. ***Weakly-supervised Few-shot Node Classification***: *Given an attributed graph* $\mathbf{G} = (\mathbf{A}, \mathbf{X})$*, and the node label set* $\mathcal{Y} = \{y_1, y_2, ..., y_c\}$ *that can be divided two subsets: the seen labels* $\mathcal{Y}_{train}$*, and the unseen labels* $\mathcal{Y}_{test}$*. Specifically, we have substantial weakly-labeled nodes with label noise for* $\mathcal{Y}_{train}$*, and few-shot clean-labeled nodes (i.e., support set* $\mathcal{S}$*) for each class in* $\mathcal{Y}_{test}$*. The problem aims to study how to predict labels*

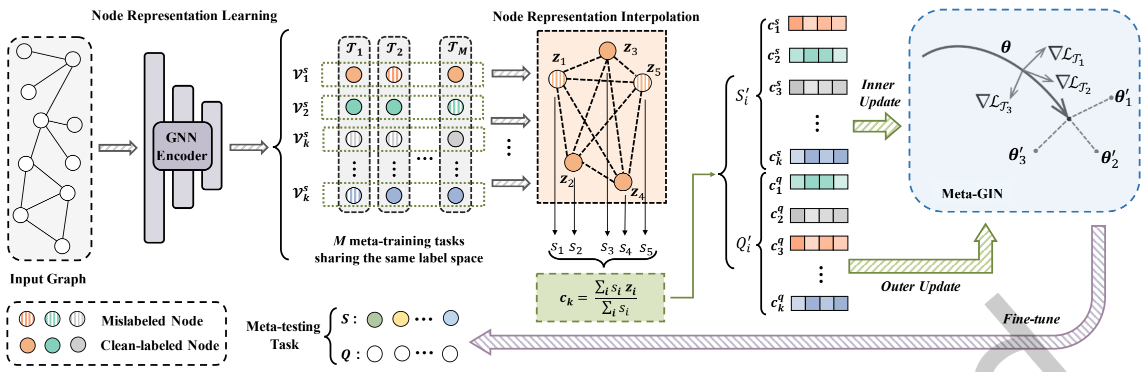

Node Representation Learning
Input Graph
GNN Encoder
M meta-training tasks sharing the same label space
Node Representation Interpolation
$c_k = \frac{\sum_i s_i z_i}{\sum_i s_i}$
Inner Update
Outer Update
Meta-GIN
Fine-tune
Mislabeled Node
Clean-labeled Node
Meta-testing Task

Fig. 1. Overview of the proposed framework. In each training episode, Meta-GIN interpolates weakly-labeled nodes from multiple meta-training tasks to obtain the noise-reduced support and query set and further extracts highly transferable meta-knowledge. During meta-testing, Meta-GIN can quickly adapt to unseen tasks with few-shot labeled instances.

for the unlabeled nodes (i.e., query set $\mathcal{Q}$) from those few-shot node classes $\mathcal{Y}_{test}$, by leveraging the knowledge of weakly-labeled nodes from $\mathcal{Y}_{train}$.

Following previous research [54], if $\mathcal{Y}_{test}$ consists of N classes and the support set $\mathcal{S}$ includes K labeled nodes per class, this problem is named N-way K-shot node classification problem. In essence, the objective of this problem is to learn a meta-classifier that can be adapted to new classes with only a few labeled nodes. Therefore, how to extract highly transferable meta-knowledge from weakly-labeled data from $\mathcal{Y}_{train}$ is the key for solving the studied problem.

4 METHODOLOGY

In this section, we introduce the details of the proposed Meta-GIN. To better explain how it works, we show its framework in Figure 1. Based on our robustness-enhanced episodic training, Meta-GIN facilitates graph meta-learning on weakly-labeled nodes by interpolated noise-reduced node representations. With the noise-reduced node representations, Meta-GIN further extracts highly transferable meta-knowledge and performs few-shot node classification on novel classes using optimization-based meta-learning. In the following subsections, we elaborate three key parts: robustness-enhanced episodic training, graph interpolation networks, and meta-optimization, respectively.

4.1 Robustness-enhanced Episodic Training

The effectiveness of few-shot learning algorithms largely benefits from the episodic training paradigm [40]. Briefly, the key idea of episodic training is to mimic the real test environment by sampling data from $\mathcal{Y}_{train}$ and the model learns over such *meta-training* tasks in a large number of episodes. Following this idea, graph FSL methods construct a pool of few-shot node classification tasks according to the seen labels. For each meta-training task $\mathcal{T}_t = \{\mathcal{S}_t, \mathcal{Q}_t\}$, the model is trained to minimize the loss of its predictions for the query set $\mathcal{Q}_t$, and goes episode by episode until convergence. In this way, the model gradually collects meta-knowledge across those meta-training tasks and then can be naturally generalized to the meta-testing task $\mathcal{T}_{test} = \{\mathcal{S}, \mathcal{Q}\}$ with unseen classes $\mathcal{Y}_{test}$.

Meta-training with Task Interpolation. However, existing graph FSL methods commonly assume that the labels of nodes from $\mathcal{Y}_{train}$ are clean, which is invalid under the weakly-supervised setting we target. To suppress the label noise during the meta-training process, we propose a robustness-enhanced episodic training paradigm by using the idea of *task interpolation*. Specifically, we sample M meta-training tasks that share the same label space and then perform interpolation among the M tasks to generate a *noise-reduced meta-training task*. Correspondingly, each node in the final N-way K-shot task is interpolated by M nodes from different tasks with the same class label.

During each episode training, we firstly sample a set of M meta-training tasks $\{\mathcal{T}_t\}_{t=1}^M$ sharing the same label space from $\mathcal{Y}_{train}$:

$$\begin{aligned} \mathcal{S}_t &= \{(v_{t,1}^s, y_{t,1}^s), (v_{t,2}^s, y_{t,2}^s), ..., (v_{t,N\times K}^s, y_{t,N\times K}^s)\}, \\ \mathcal{Q}_t &= \{(v_{t,1}^q, y_{t,1}^q), (v_{t,2}^q, y_{t,2}^q), ..., (v_{t,N\times K'}^q, y_{t,N\times K'}^q)\}, \\ \{\mathcal{T}_t\}_{t=1}^M &= \{\{\mathcal{S}_1, \mathcal{Q}_1\}, \{\mathcal{S}_2, \mathcal{Q}_2\}, ..., \{\mathcal{S}_M, \mathcal{Q}_M\}\}, \end{aligned} \tag{1}$$

where all the task in $\{\mathcal{T}_t\}_{t=1}^M$ are sampled from the same N classes. For each meta-training task $\mathcal{T}_t$, the support set $\mathcal{S}_t$ contains N classes and K weakly-labeled nodes per class, while $\mathcal{Q}_t$ containing K' query nodes sampled from the remainder of each of the N classes.

Furthermore, for each set of M label-sharing meta-training tasks with weakly-labeled nodes, the proposed framework Meta-GIN will try to generate a noise-reduced meta-training task via task interpolation to improve the effectiveness of episodic training under the weakly-supervised setting. Thus we can get the noise-reduced support and query sets:

$$\begin{aligned} \mathcal{S}' &= \{(\mathbf{c}_1^s, y_1^s), (\mathbf{c}_2^s, y_2^s), ..., (\mathbf{c}_{N\times K}^s, y_{N\times K}^s)\}, \\ \mathcal{Q}' &= \{(\mathbf{c}_1^q, y_1^q), (\mathbf{c}_2^q, y_2^q), ..., (\mathbf{c}_{N\times K'}^q, y_{N\times K'}^q)\}, \end{aligned} \tag{2}$$

where $\mathbf{c}_k$ denotes the noise-reduced node representation generated from $\mathcal{V}_k = \{v_{1,k}, v_{2,k}, ..., v_{M,k}\}$ across the M tasks and y_k denotes its corresponding shared target class. With the noise-reduced meta-training task, our model could further extract highly transferable meta-knowledge and solve the weakly-labeled graph meta-learning problems.

4.2 Graph Interpolation Networks

Moreover, we propose a new family of graph neural networks, called Graph Interpolation Networks (GIN) to facilitate graph meta-learning on weakly-labeled nodes. In essence, GIN is composed of two key building blocks, including (1) a *node representation learning* module that embeds each node; and (2) a *node interpolation* module for deriving noise-reduced node representations for the final noise-reduced meta-training task. The details are as follows:

Node Representation Learning. The initial step of conducting graph meta-learning is to learn expressive node representations that capture both graph structure and node features. To achieve this, we first design a GNN-based encoding module in GIN. Specifically, it is built with multiple GNN layers that encode each node to a low-dimensional latent representation. The core operation in GNNs is the message-passing scheme, in which information is propagated from each node to its neighborhoods with specific deterministic propagation rules. In general, a GNN layer can be defined as:

$$\begin{aligned} \mathbf{h}_i^l &= \textsc{Transform}^l\Big(\mathbf{h}_i^{l-1}, \mathbf{h}_{\mathcal{N}_i}^l\Big), \\ \mathbf{h}_{\mathcal{N}_i}^l &= \textsc{Aggregate}^l\Big(\{\mathbf{h}_j^{l-1} | \forall v_j \in \mathcal{N}_i \cup \{v_i\}\}\Big), \end{aligned} \tag{3}$$

where $\mathbf{h}_i^l$ is the node representation of node i at layer l and $\mathcal{N}_i$ is the set of neighboring nodes of v_i. Transform and Aggregate are two key functions of GNNs. Furthermore, by stacking multiple GNN layers, the model is able to capture the long-range node dependencies in the graph.

It is worth noting that the encoding module is compatible with arbitrary GNN-based architecture [11, 18, 39]. To improve the model efficiency on large graphs, we employ Simple Graph Convolution (SGC) [46] in this work. Specifically, SGC utilizes a simplified graph convolution pre-processing followed by standard multi-class logistic regression. Let $\tilde{\mathbf{A}} = \mathbf{A}+\mathbf{I}$ denotes the "normalized" adjacency matrix with added self-loops, $\tilde{\mathbf{D}}$ is the corresponding degree matrix of $\tilde{\mathbf{A}}$, the node representation with L layers propagation can be computed by:

$$Z = \underbrace{\tilde{\mathbf{A}}\ldots\tilde{\mathbf{A}}\tilde{\mathbf{A}}}_{L} X\Theta, \tag{4}$$

where $\mathbf{Z} \in \mathbb{R}^{n\times d'}$ denotes node representation matrix and the $\Theta \in \mathbb{R}^{d\times d'}$ is a learnable parameter matrix. This way the encoding model is linear, but still has the same increased "receptive field" of a L-layer GCN [18].

Node Representation Interpolation. However, the previously learned node representations from seen classes $\mathcal{Y}_{train}$ are unable to represent their corresponding classes since they are likely mislabeled. Hence, our *node representation interpolation* module aims at interpolating nodes in node set $\mathcal{V}_k$ to generate noise-reduced node representations and leverage those interpolated node representations to learn the concept of each class.

To generate a noise-reduced representation with the M labeled nodes from a set $\mathcal{V}_k$, one straightforward solution is taking the average of all the embedded nodes belonging to that set with $\mathbf{p}_k = \frac{1}{|\mathcal{V}_k|}\sum_{i\in\mathcal{V}_k}\mathbf{z}_i$, where $\mathbf{z}_i$ is the learned node representations from the *graph representation learning* module of node v_i. However, directly taking the mean vectors of the embedded nodes as noise-reduced node representation could be ineffective due to the existence of mislabeled nodes. Specifically, our *node representation interpolation* module is designed to estimate a confidence score α_i for each node and further perform fine-grained interpolation among the nodes from M tasks.

To identify the confidence score of each labeled node, GIN will compute the confidence score of each sample with message passing. As shown in Figure 1, we first build a full-connected interpolation graph using M sampled nodes from each set $\mathcal{V}_k$, then we develop a node re-weighting layer using graph attention mechanism to aggregate and compare the information among weakly-labeled nodes. The node re-weighting layer can be defined as follows:

$$s_i = \sigma\Bigg(\sum_{v_j\in\mathcal{V}_k}\alpha_{ij}\mathbf{w}^{\mathrm{T}}[\mathbf{z}_j||\Delta_j]\Bigg), \tag{5}$$

where $\mathbf{w} \in \mathbb{R}^{2d'}$ is the learnable parameter vector, σ is a nonlinear activation function, i.e., sigmoid function. s_i denote the confidence score of node v_i and $\Delta_j = \mathbf{z}_j - \mathbf{p}_k$ captures the difference between the embedding of node v_j and the prototype of $\mathcal{V}_k$. By incorporating the distance between each node and the prototype, GIN can better perceive the concept of the corresponding class and compute the final confidence score. Specifically, α_{ij} is the attention weight between nodes v_i and v_j, we compute it via attention:

$$\alpha_{ij} = \frac{\exp\big(\mathrm{LeakyReLU}\big(\mathbf{a}^{\mathrm{T}}[\mathbf{w}^{\mathrm{T}}\tilde{\mathbf{z}}_i||\mathbf{w}^{\mathrm{T}}\tilde{\mathbf{z}}_j]\big)\big)}{\sum_{m\in\mathcal{V}_k}\exp\big(\mathrm{LeakyReLU}\big(\mathbf{a}^{\mathrm{T}}[\mathbf{w}^{\mathrm{T}}\tilde{\mathbf{z}}_i||\mathbf{w}^{\mathrm{T}}\tilde{\mathbf{z}}_m]\big)\big)}, \tag{6}$$

where $\tilde{\mathbf{z}}_i = [\mathbf{z}_i||\mathbf{\Delta}_i]$ and the attention vector $\mathbf{a}$ is a trainable weight vector that assigns importance to different node during aggregation.

After obtaining the confidence scores of the weakly-labeled nodes in each set $\mathcal{V}_k$, GIN is able to generate a noise reduced node representation by interpolating these noisy labeled nodes with their weights. Based on the computed attentional weights from Eq. (5), we can obtain the interpolated representation with $\mathbf{c}_k = \frac{1}{\sum_i s_i}\sum_i s_i\mathbf{z_i}$.

Node Classification. With the noise-reduced support set $\mathcal{S}'$ that contains K interpolated node representations for each of the N classes, GIN will try to classify each instance to its corresponding class label. This can be done with a feed-forward layer $\mathbf{y}_k = \mathrm{softmax}(\mathbf{W}_c^{\mathrm{T}}\mathbf{c}_k + \mathbf{b}_c)$, where $\mathbf{W}_c \in \mathbb{R}^{d'\times N}$ and $\mathbf{b}_c \in \mathbb{R}^N$ are learnable weight matrix and bias, respectively. Under the episodic training framework, the objective of each meta-training task is to minimize the cross-entropy loss function for performing node classification. Specifically, the training loss for each interpolated instance $\mathbf{c}_k$ is computed by:

$$\mathcal{L} = -\log p(y_k^*|\mathbf{c}_k), \tag{7}$$

where y_k^* is the shared label of set $\mathcal{V}_k$. As the training instances are computed by the *node interpolation* module, GIN is able to reduce the negative impacts of mislabeled nodes during the meta-learning process. By minimizing the above loss function, GIN is able to learn a generic classifier for a specific N-way K-set meta-training task and further extract highly transferrable meta-knowledge from weakly-labeled data.

4.3 Meta-optimization

Having the proposed Graph Interpolation Networks, we are able to obtain noise-reduced support set $\mathcal{S}'$ and query set $\mathcal{Q}'$ via interpolating multiple meta-training tasks. Upon that, we train the model via meta-learning, such that the meta-learned GIN model (Meta-GIN) is capable of effectively adapting to new tasks with few labeled instances. Specifically, we follow model-agnostic meta-learning [8] to learn Meta-GIN in an optimization-based fashion, in order to better exploit the clean-labeled support nodes during meta-testing and make fast and effective adaptation to a new task through a small number of gradient steps.

Meta-training. In the meta-training stage, we expect to obtain a good initialization of GIN, which is inherently generalizable to unseen tasks, and explicitly encourage the initialization parameters to perform well after a small number of gradient descent updates on a new learning task. When learning a specific interpolated task $\mathcal{T}_i'$, we begin with feeding the nodes from the noise-reduced support set $\mathcal{S}_i'$ to GIN, and calculate the cross-entropy loss $\mathcal{L}_{\mathcal{T}_i'}$ as formulated in Eq. (7). We consider a GIN model represented by a parameterized function $f_{\boldsymbol{\theta}}$ with parameters $\boldsymbol{\theta}$, the optimization algorithm first adapts the initial model parameters $\boldsymbol{\theta}$ to $\boldsymbol{\theta}_i'$ for each interpolated learning task $\mathcal{T}_i'$ independently. Specifically, the updated parameter $\boldsymbol{\theta}_i'$ is computed using $\mathcal{L}_{\mathcal{T}_i'}$ on the interpolated node representation and the corresponding class label. Formally, the parameter update with one gradient step can be expressed as:

$$\boldsymbol{\theta}_i' = \boldsymbol{\theta} - \alpha \nabla_{\boldsymbol{\theta}} \mathcal{L}_{\mathcal{T}_i'}(f_{\boldsymbol{\theta}}), \tag{8}$$

where α controls the learning rate for each task. Note that Eq. (8) only includes one-step gradient update, while it is straightforward to extend to multiple gradient updates [8].

Our model Meta-GIN is trained by optimizing for the best performance of $f_{\boldsymbol{\theta}}$ with respect to $\boldsymbol{\theta}$ across all interpolated meta-training tasks. More concretely, the meta-objective function is defined as follows:

$$\min_{\boldsymbol{\theta}} \sum_{\mathcal{T}_i' \sim p(\mathcal{T})} \mathcal{L}_{\mathcal{T}_i'}(f_{\boldsymbol{\theta}_i'}) = \min_{\boldsymbol{\theta}} \sum_{\mathcal{T}_i' \sim p(\mathcal{T})} \mathcal{L}_{\mathcal{T}_i'}(f_{\boldsymbol{\theta} - \alpha \nabla_{\boldsymbol{\theta}} \mathcal{L}_{\mathcal{T}_i'}(f_{\boldsymbol{\theta}})}), \tag{9}$$

where $p(\mathcal{T})$ is the distribution of interpolated meta-training tasks. Since the meta-optimization is performed over parameters $\boldsymbol{\theta}$ with the objective computed using the updated parameters (i.e., $\boldsymbol{\theta}_i'$) for all tasks, correspondingly, the model parameters are optimized such that one or a small number of gradient steps on the target task will produce maximal effectiveness.

Formally, we leverage stochastic gradient descent (SGD) to update the model parameters $\boldsymbol{\theta}$ with the instances from the interpolated query set, such that the model parameters $\boldsymbol{\theta}$ are updated as follows:

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \beta \nabla_{\boldsymbol{\theta}} \sum_{\mathcal{T}_i' \sim p(\mathcal{T})} \mathcal{L}_{\mathcal{T}_i'}(f_{\boldsymbol{\theta}_i'}), \tag{10}$$

where β is the meta step size. The detailed learning process of Meta-GIN is presented in Algorithm 1.

Meta-testing. After training on a considerable number of meta-training tasks, we expect that the Meta-GIN model has been gradually meta-learned well for handling unseen few-shot node classification tasks. Its generalization performance will be measured on the test episodes, which contain clean-labeled nodes sampled from $\mathcal{Y}_{test}$ instead

Algorithm 1: The learning algorithm of Meta-GIN.

```
Input: Task distribution $p(\mathcal{T})$ over the input graph $\mathbf{G}$
Output: The well-trained model Meta-GIN
1  Randomly initialize the parameters $\boldsymbol{\theta}$ of GIN
2  while not converge do
3      Randomly sample a batch of task sets with all the tasks in a set $\{\mathcal{T}_t\}_{t=1}^{M}$ sharing the same label space.
4      for each task set $\{\mathcal{T}_t\}_{t=1}^{M}$ do
5          // Node Interpolation
6          for each $\mathcal{V}_k \in \{\{\mathcal{S}_t\}_{t=1}^{M}, \{\mathcal{Q}_t\}_{t=1}^{M}\}$ do
7              Compute the representations for nodes in $\mathcal{V}_k$
8              Interpolate and obtain the noise-reduced node representation $\mathbf{c}_k$
9          Obtain the noise-reduced $\mathcal{S}'_i$ and $\mathcal{Q}'_i$
10         Evaluate $\nabla_{\boldsymbol{\theta}}\mathcal{L}_{\mathcal{T}'_i}(f_{\boldsymbol{\theta}})$ using $\mathcal{S}'_i$ and $\mathcal{L}_{\mathcal{T}'_i}$ in Eq. (7)
11         Compute adapted parameters $\boldsymbol{\theta}'$ by Eq. (8)
12     Update $\boldsymbol{\theta}$ with the interpolated query set by Eq. (9)
13 return Meta-learned graph meta-learning model Meta-GIN
```

of $\mathcal{Y}_{train}$. For each meta-testing episode, we we will remove the Node Interpolation module, and fine-tune the meta-learned classifier Meta-GIN with the provided clean-labeled support set $\mathcal{S}$ and classify each query node in $\mathcal{Q}$ into the most likely class.

5 EXPERIMENTS

In this section, we will start with the experimental setup and then present our experiment results to answer three research questions: **RQ1**: Is Meta-GIN effective in solving the weakly-supervised few-shot node classification? **RQ2**: Can Meta-GIN achieve satisfying performance on a variety of noise levels? **RQ3**: How does each module and the hyper-parameters in Meta-GIN contribute to the final performance?

5.1 Experiment Settings

Evaluation Datasets. In our experiments, we adopt four datasets used in previous research [4, 16] for few-shot node classification. The summary of dataset statistics is presented in Table 2.

- **Electronics** [28] is built with the products in "Electronics" on Amazon[2]. In this network, products are considered as nodes and their attributes are constructed with the corresponding product description. We use low-level product categories to define the class label and the complementary relationship ("bought together") between products to connect the nodes. Those classes containing 100 to 1000 nodes are selected and the isolated products is deleted.
- **Clothing** [28] is an additional product network consisting of products categorized under "Electronics" on Amazon. Similar to Electronics, each node in this network represents a product, and its attributes provide a description of the respective product while the its label is the fine-grained product category. We establish edges between products based on the complementary relationship of being "bought together".
- **DBLP** [38] is a citation network[3] constructed with data extracted from DBLP in which nodes denoting papers and node attributes are from paper abstracts. The citation relations among papers are used to create links

[2]http://snap.stanford.edu/data/amazon/productGraph/
[3]https://lfs.aminer.cn/misc/dblp.v11.zip

Table 2. Statistics of the evaluation datasets.

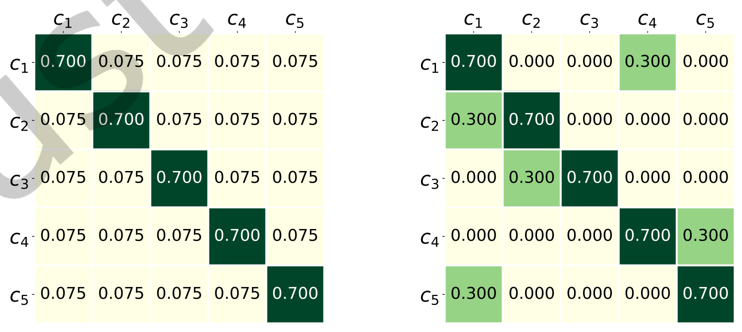
Datasets	# nodes	# edges	# attributes	# Train/Valid/Test
Electronics	42,318	43,556	8,669	90/37/40
Clothing	24,919	91,680	9,037	40/17/20
DBLP	40,672	288,270	7,202	80/27/30
ogbn-arxiv	169,343	1,166,243	128	16/12/12

between nodes and the paper venue is utilized to define the class label. For constructing the graph, each paper in these venues is treated as a node and the citation relations are regarded as links between them. We apply bag-of-words model on the paper abstract to obtain node attributes.

- **ogbn-arxiv** [15] is a benchmark dataset from the Open Graph Benchmark (OGB)[4], in which each node denotes a paper with its subject area as the class label. The links between nodes indicate the citation relation. Specifically, ogbn-arxiv is built with all Computer Science Arxiv papers indexed by MAG. For each paper, its attributes are obtained by averaging the 128-dimension word2vec embeddings of words in its title and abstract.

We follow the same train/validation/test splits and data preprocess procedure as in [5] for Electronics, Clothing and DBLP datasets. For ogbn-arxiv dataset, we retrieve the network with the public OGB package and split it for few-shot learning node classification scenario.

Label Corruption. To explore the performance of different methods for graph meta-learning on weakly-labeled data, we follow previous work [3, 14, 17, 34] and inject two representative types of label noise (i.e., symmetric and asymmetric noise) to the datasets. Specifically, the noise injection is done by flipping the labels following the transition probabilities defined in a corruption matrix T, in which T_{ij} denotes the probability of flipping class c_i to class c_j. For injecting noise of ratio ϵ to a dataset with P classes: **Symmetric noise** flips a label uniformly to all the other classes, s.t. $T_{ii} = 1 - \epsilon$ and $T_{ij} = \epsilon/(P-1)$ if $i \neq j$; **Asymmetric noise** flips a label to a different class with probability ϵ, s.t. $T_{ii} = 1 - \epsilon$ and $\exists i \neq j$, $T_{ij} = \epsilon$. In Figure 2, we visualize the examples of corruption matrix for the dataset containing 5 label classes. To make the evaluation more realistic, *both training and validation data will be perturbed.*



(a) **Symmetric Noise** ($\epsilon = 0.3$) (b) **Asymmetric Noise** ($\epsilon = 0.3$)

Fig. 2. Example of the noise corruption matrix.

[4]https://ogb.stanford.edu/

Compared Methods. In the experiments, we compare the proposed model GPN with two different categories of methods: (1) *GNN-based methods* covering three representative semi-supervised node classification methods GCN, SGC, GraphSAGE and PTA which are adapted for few-shot learning scenarios as in [5, 54]. (2) *Graph meta-learning methods* Meta-GNN, GPN, G-Meta and Meta-GPS, which are the state-of-the-arts for graph few-shot node classification:

- **GCN** [18]: It learns node representations with stacked layers of first-order approximation of spectral graph convolutions.
- **SGC** [46]: This linear model reduces the unnecessary complexity of GCN by successively collapsing the convolution functions between consecutive layers into a linear transformation.
- **GraphSAGE** [11]: It can efficiently generate node embeddings by uniformly sampling a fixed size of neighbors and then aggregating the feature information from the neighbors.
- **PTA** [7]: It is a decoupled GNN which is robust to label noise using adaptive weighting strategy.
- **Meta-GNN** [54]: It applies MAML [8] to Simple Graph Convolution (SGC) for few-shot node classification in graphs.
- **GPN** [5]: This Graph Prototypical Network can learn highly representative class prototypes with a GNN-based network encoder and node valuator. It predicts node labels by measuring their similarity with prototypes.
- **G-Meta** [16]: It constructs a local subgraph for each node and regards the centroid embedding of subgraphs as the prototypes. It is optimized with both the prototypical loss and MAML.
- **Meta-GPS** [24] extends GPN with Prototype and Scaling & shifting transformation.

It is worth noting that existing methods for learning with noisy labels cannot be applied to few-shot node classification without principled modifications and only achieve very poor performance in our preliminary experiments, due to the space limit, we do not include those methods in this paper.

Model Implementation. The proposed model is implemented in PyTorch. Specifically, we employ a 2-layer propagation SGC for the node representation learning module. As for the clean node hallucination module, we use one aggregation layer and the negative slope for the LeakyReLU in it is set to be 0.2. We grid search for task numbers in {1, 5, 10, 15, 20, 25}, meta learning rate α and meta step size β in {0.0001, 0.0005, 0.001, 0.005, 0.01, 0.05, 0.1, 0.5}. The optimal values are selected when the model achieve the best performance for validation set. We select the meta-learning rate α to be 0.1 and the meta step size β to be 0.001. For model training, the task number in each batch is 5 and the query size K' is 5. For constructing the meta-training episodes, unless otherwise notice, we let the set size M to be 5 for both the support and query set. We train the model with 20,000 episodes or stop earlier when the performance on validation set converges. To make a fair comparison, we test all the baseline methods with the publicly released implementations, and we fine-tune the hyperparameters to report their best performance in the experiments.

5.2 General Comparisons (RQ1)

We evaluate the proposed Meta-GHN and all the baseline models on different weakly-supervised node classification tasks including 5-way 1-shot, 5-way 3-shot, 10-way 1-shot, and 10-way 3-shot. For each of the datasets, we inject either the symmetric noise or the Asymmetric noise with noise ratio $\epsilon = 30\%$. Following previous work of few-shot node classification, we adopt *Accuracy* (ACC) as the evaluation metric for evaluating performance. We randomly sampled 100 *meta-test* tasks from the test node classes and evaluate each method on these tasks. The process is repeated for 10 times and the resulted the mean ± standard deviation is reported in Table 3. In the following, we elaborate our in-depth observations and analysis based on the results:

- In general, the proposed Meta-GIN can significantly outperform all the baseline methods on weakly-supervised node classification tasks for different datasets corrupted by either symmetric or Asymmetric label noise. Take the benchmark ogbn-arxiv dataset as an example, we find that Meta-GIN can achieve 8.0% and 3.1%

Table 3. Weakly-supervised few-shot node classification accuracy (30% label noise).

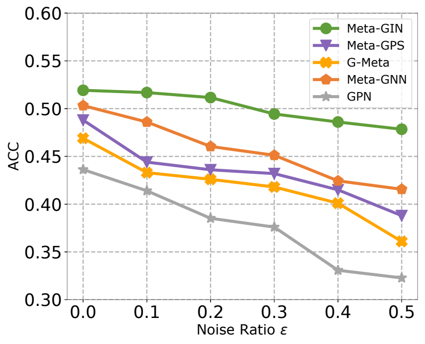
Electronics	5-way 1-shot		5-way 3-shot		10-way 1-shot		10-way 3-shot	
	Symmetric	Asymmetric	Symmetric	Asymmetric	Symmetric	Asymmetric	Symmetric	Asymmetric
GCN	0.386 ± 0.025	0.381 ± 0.023	0.534 ± 0.027	0.529 ± 0.023	0.211 ± 0.019	0.197 ± 0.021	0.376 ± 0.027	0.372 ± 0.022
SGC	0.390 ± 0.019	0.385 ± 0.021	0.535 ± 0.018	0.527 ± 0.017	0.248 ± 0.016	0.232 ± 0.012	0.383 ± 0.013	0.380 ± 0.015
GraphSAGE	0.338 ± 0.018	0.406 ± 0.023	0.529 ± 0.021	0.568 ± 0.016	0.211 ± 0.026	0.224 ± 0.018	0.362 ± 0.026	0.373 ± 0.014
PTA	0.405 ± 0.023	0.437 ± 0.018	0.553 ± 0.012	0.581 ± 0.022	0.252 ± 0.016	0.255 ± 0.015	0.394 ± 0.023	0.409 ± 0.018
Meta-GNN	0.403 ± 0.021	0.480 ± 0.018	0.601 ± 0.028	0.650 ± 0.024	0.279 ± 0.034	0.325 ± 0.031	0.555 ± 0.023	0.563 ± 0.029
GPN	0.408 ± 0.015	0.463 ± 0.023	0.629 ± 0.024	0.651 ± 0.027	0.263 ± 0.011	0.314 ± 0.026	0.535 ± 0.011	0.579 ± 0.016
G-Meta	0.488 ± 0.025	0.496 ± 0.024	0.614 ± 0.018	0.658 ± 0.022	0.336 ± 0.025	0.376 ± 0.021	0.463 ± 0.021	0.504 ± 0.019
Mata-GPS	0.495 ± 0.021	0.512 ± 0.019	0.631 ± 0.019	0.662 ± 0.025	0.359 ± 0.018	0.397 ± 0.019	0.541 ± 0.024	0.561 ± 0.021
Meta-GIN	**0.694 ± 0.033**	**0.652 ± 0.010**	**0.750 ± 0.021**	**0.725 ± 0.021**	**0.567 ± 0.027**	**0.559 ± 0.025**	**0.627 ± 0.028**	**0.615 ± 0.022**

Clothing	5-way 1-shot		5-way 3-shot		10-way 1-shot		10-way 3-shot	
	Symmetric	Asymmetric	Symmetric	Asymmetric	Symmetric	Asymmetric	Symmetric	Asymmetric
GCN	0.381 ± 0.011	0.367 ± 0.010	0.524 ± 0.030	0.536 ± 0.055	0.228 ± 0.055	0.214 ± 0.027	0.405 ± 0.057	0.400 ± 0.017
SGC	0.456 ± 0.021	0.459 ± 0.026	0.564 ± 0.033	0.560 ± 0.028	0.323 ± 0.018	0.317 ± 0.041	0.432 ± 0.025	0.428 ± 0.035
GraphSAGE	0.493 ± 0.028	0.473 ± 0.037	0.592 ± 0.057	0.588 ± 0.039	0.348 ± 0.037	0.345 ± 0.046	0.502 ± 0.036	0.461 ± 0.026
PTA	0.501 ± 0.030	0.528 ± 0.039	0.628 ± 0.034	0.639 ± 0.037	0.394 ± 0.029	0.359 ± 0.031	0.510 ± 0.025	0.512 ± 0.037
Meta-GNN	0.560 ± 0.034	0.582 ± 0.045	0.762 ± 0.033	0.760 ± 0.028	0.461 ± 0.032	0.514 ± 0.027	0.654 ± 0.031	0.623 ± 0.043
GPN	0.568 ± 0.015	0.610 ± 0.028	0.740 ± 0.027	0.766 ± 0.016	0.467 ± 0.029	0.507 ± 0.019	0.636 ± 0.038	0.649 ± 0.030
G-Meta	0.587 ± 0.039	0.595 ± 0.022	0.779 ± 0.031	0.788 ± 0.040	0.509 ± 0.033	0.524 ± 0.037	0.655 ± 0.023	0.660 ± 0.032
Mata-GPS	0.602 ± 0.027	0.617 ± 0.031	0.781 ± 0.029	0.790 ± 0.037	0.521 ± 0.030	0.539 ± 0.034	0.669 ± 0.031	0.652 ± 0.029
Meta-GIN	**0.769 ± 0.038**	**0.780 ± 0.025**	**0.846 ± 0.021**	**0.839 ± 0.028**	**0.664 ± 0.013**	**0.639 ± 0.033**	**0.692 ± 0.039**	**0.694 ± 0.026**

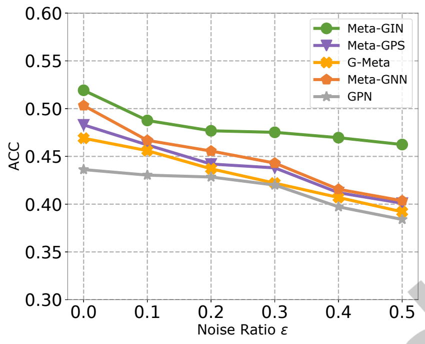
DBLP	5-way 1-shot		5-way 3-shot		10-way 1-shot		10-way 3-shot	
	Symmetric	Asymmetric	Symmetric	Asymmetric	Symmetric	Asymmetric	Symmetric	Asymmetric
GCN	0.369 ± 0.022	0.345 ± 0.031	0.470 ± 0.041	0.444 ± 0.031	0.212 ± 0.030	0.201 ± 0.014	0.345 ± 0.033	0.335 ± 0.042
SGC	0.376 ± 0.023	0.374 ± 0.016	0.487 ± 0.024	0.479 ± 0.023	0.224 ± 0.013	0.223 ± 0.014	0.359 ± 0.019	0.348 ± 0.025
GraphSAGE	0.345 ± 0.021	0.354 ± 0.028	0.536 ± 0.021	0.540 ± 0.024	0.262 ± 0.014	0.279 ± 0.015	0.350 ± 0.018	0.395 ± 0.023
PTA	0.388 ± 0.020	0.392 ± 0.013	0.550 ± 0.024	0.561 ± 0.025	0.280 ± 0.016	0.303 ± 0.019	0.366 ± 0.022	0.427 ± 0.025
Meta-GNN	0.581 ± 0.010	0.611 ± 0.009	0.684 ± 0.021	0.702 ± 0.025	0.498 ± 0.028	0.514 ± 0.025	0.573 ± 0.021	0.578 ± 0.024
GPN	0.566 ± 0.020	0.621 ± 0.014	0.758 ± 0.009	0.766 ± 0.014	0.464 ± 0.011	0.501 ± 0.018	0.649 ± 0.017	0.650 ± 0.015
G-Meta	0.618 ± 0.027	0.627 ± 0.022	0.697 ± 0.025	0.761 ± 0.010	0.497 ± 0.025	0.502 ± 0.014	0.536 ± 0.027	0.605 ± 0.012
Mata-GPS	0.631 ± 0.018	0.639 ± 0.020	0.749 ± 0.016	0.759 ± 0.013	0.504 ± 0.021	0.511 ± 0.017	0.613 ± 0.018	0.629 ± 0.020
Meta-GIN	**0.734 ± 0.012**	**0.739 ± 0.019**	**0.794 ± 0.010**	**0.772 ± 0.016**	**0.591 ± 0.023**	**0.593 ± 0.020**	**0.672 ± 0.020**	**0.695 ± 0.011**

ogbn-arxiv	5-way 1-shot		5-way 3-shot		10-way 1-shot		10-way 3-shot	
	Symmetric	Asymmetric	Symmetric	Asymmetric	Symmetric	Asymmetric	Symmetric	Asymmetric
GCN	0.259 ± 0.033	0.244 ± 0.018	0.296 ± 0.029	0.288 ± 0.021	0.142 ± 0.017	0.122 ± 0.015	0.171 ± 0.010	0.157 ± 0.011
SGC	0.277 ± 0.020	0.274 ± 0.011	0.334 ± 0.018	0.321 ± 0.009	0.157 ± 0.007	0.155 ± 0.011	0.217 ± 0.009	0.196 ± 0.012
GraphSAGE	0.280 ± 0.021	0.260 ± 0.019	0.317 ± 0.022	0.297 ± 0.025	0.143 ± 0.016	0.121 ± 0.012	0.170 ± 0.015	0.144 ± 0.019
PTA	0.311 ± 0.018	0.303 ± 0.020	0.352 ± 0.016	0.344 ± 0.022	0.187 ± 0.017	0.180 ± 0.017	0.235 ± 0.014	0.227 ± 0.016
Meta-GNN	0.451 ± 0.017	0.443 ± 0.009	0.481 ± 0.028	0.478 ± 0.026	0.230 ± 0.030	0.222 ± 0.018	0.327 ± 0.021	0.302 ± 0.011
GPN	0.376 ± 0.015	0.420 ± 0.019	0.492 ± 0.018	0.514 ± 0.019	0.255 ± 0.019	0.266 ± 0.017	0.266 ± 0.015	0.338 ± 0.009
G-Meta	0.418 ± 0.012	0.422 ± 0.014	0.453 ± 0.013	0.500 ± 0.015	0.272 ± 0.012	0.282 ± 0.018	0.355 ± 0.017	0.377 ± 0.011
Mata-GPS	0.432 ± 0.016	0.438 ± 0.015	0.469 ± 0.013	0.487 ± 0.011	0.263 ± 0.011	0.271 ± 0.014	0.335 ± 0.015	0.357 ± 0.014
Meta-GIN	**0.494 ± 0.021**	**0.475 ± 0.018**	**0.572 ± 0.013**	**0.545 ± 0.016**	**0.336 ± 0.021**	**0.325 ± 0.023**	**0.447 ± 0.013**	**0.390 ± 0.013**

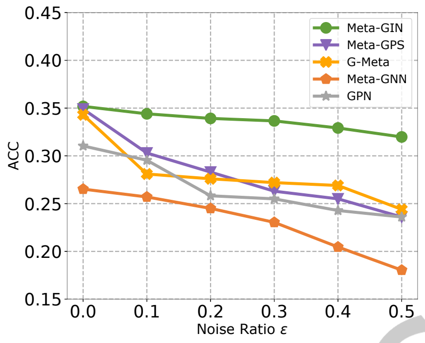
improvement for the 5-way-3-shot task with symmetric and Asymmetric noise if we compare it with the best performing baseline. On Electronics dataset, the improvements are more substantial. We can conclude that Meta-GIN is effective in solving the challenging problem of weakly-supervised graph meta-learning.



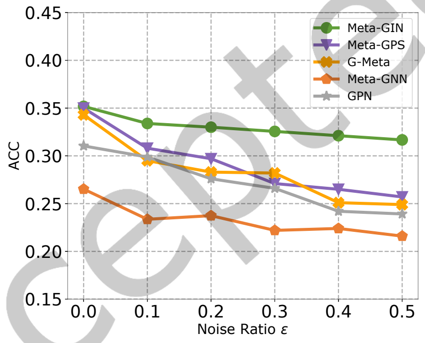
(a) 5-w 1-s (Sym)



(b) 5-w 1-s (Asym)



(c) 10-w 1-s (Sym)



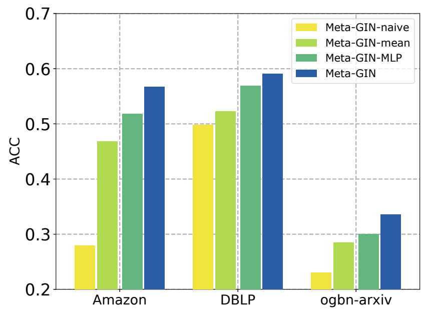
(d) 10-w 1-s (Asym)

Fig. 3. Weakly-Supervised few-shot node classification accuracy *w.r.t.* different noise ratios on ogbn-arxiv.

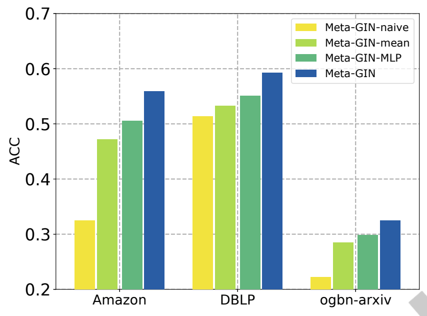
- GNN-based methods such as GCN, SGC, and GraphSAGE are originally proposed for semi-supervised node classification and require abundant clean labeled data to achieve satisfying classification accuracy. Thus, they obtain poor performance while adapted for weekly-supervised few-shot learning scenarios. Though PTA can mitigate label noise to some extent, it is unable to transfer the knowledge from seen classes to unseen classes. For the graph FSL methods, they still fall behind the proposed Meta-GIN on the weakly-labeled data. This observation verifies the assumption that most of the existing FSL models is vulnerable to label noise.
- Powered by the well-designed Graph Interpolation Networks under the robustness-enhanced episodic training framework, Meta-GIN is able to generate the noise-reduced node representation and achieve the best few-shot node classification performance on the noisy label data. It is also worth noting that noise usually have larger impact on meta-learning relying on fewer shot. However, compared with the graph meta-learning methods, the improvement of Meta-GIN on N-way-1-shot tasks is larger than that on N-way-3-shot tasks. This illustrates Meta-GIN's power on denoising for the practical few-shot learning scenarios.

5.3 Robustness Analysis (RQ2)

To examine the robustness of Meta-GIN on data with different noise levels, we show its performance in Figure 3 by varying the noise ratio. Firstly, on the data with no injected noise (i.e., $\epsilon = 0$), Meta-GIN can still outperform the state-of-the-art for graph few-shot learning, which shows it is powerful in extrapolating the knowledge from



(a) 10-w 1-s (Sym $\epsilon = 30\%$)



(b) 10-w 1-s (Asym $\epsilon = 30\%$)

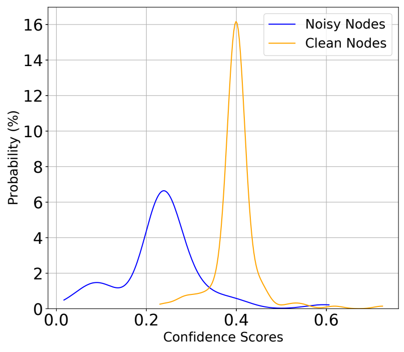
Fig. 4. Ablation results for different model variants.

seen to unseen node classes. Then if we inject the noise, the performance of all the baseline methods is degraded as the noise ratio increases, which is in accordance with our assumption. In addition, we also find that symmetric noise leads to larger decrease in the performance compared to asymmetric noise in both 5-way 1-way and 10-way 1-way tasks. The main reason is that corrupting a label to a wider range of node classes may lead to a more challenging weakly-supervised meta-learning task. However, when we increase the noise ratio for either the symmetric or asymmetric noise, the performance of Meta-GIN does not decrease very much. It can obtain larger improvement compared with the baselines in the data with higher noise level. This verifies the effectiveness of Meta-GIN in achieving robust performance on weakly-labeled data.

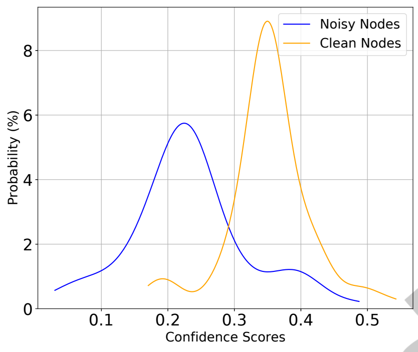
5.4 Model Design Analysis (RQ3)

Ablation Study. To investigate the contribution of each component in Meta-GIN, we compare it with its variants in Figure 4. Specifically, *Meta-GIN-naive* can be considered as a naive variant by excluding both robustness-enhanced episodic training and node interpolation. *Meta-GIN-MLP* and *Meta-GIN-mean* denote the variants that calculate the confidence score for each node using a fully connecting layer and taking the average, respectively. As shown in the reported results, *Meta-GIN-naive* is highly vulnerable to label noise and unable to obtain competitive results with other variants on weakly-labeled few-shot node classification. Based on the proposed robustness-enhanced episodic training, *Meta-GIN-mean* uses the simplest way to compute the noise-reduced node representations, but can significantly outperforms *Meta-GIN-naive*, which verifies the importance of using the new episodic training paradigm. Meanwhile, though *Meta-GIN-MLP* can improve *Meta-GIN-mean* by assigning weigthed confidence score to each node, it still fall behind our approach, which shows the node interpolation module can better estimate the confidence score of each weakly-labeled node via message passing.

Visualization. To further validate the effectiveness of the confidence scores we obtain from Eq. 6, we compare the distribution of the confidence scores for clean labels and noisy labels in Clothing datasets using a 10-way 3-shot task. Figure 5 provides a visual representation of the obtained confidence score distributions. As anticipated, the results demonstrate a clear distinction between the confidence scores assigned to clean nodes and noisy nodes. Specifically, we observed a significantly higher probability of obtaining higher confidence scores for clean nodes compared to the noisy nodes. Similar patterns are also observed on other tasks with different datasets. By assigning higher weights to the clean nodes and lower weights to the noisy nodes during the interpolation process, the Node Representation Interpolation module in GIN can effectively prioritize the information from the clean nodes while attenuating the influence of the noisy nodes, leading to more robust and accurate interpolated representations for each set.

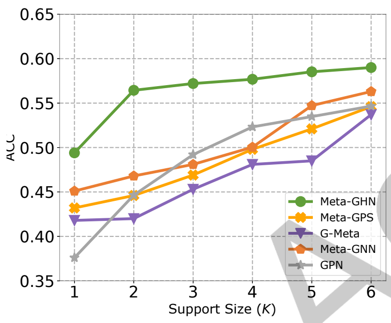


(a) 10-w 3-s (Sym $\epsilon = 30\%$)

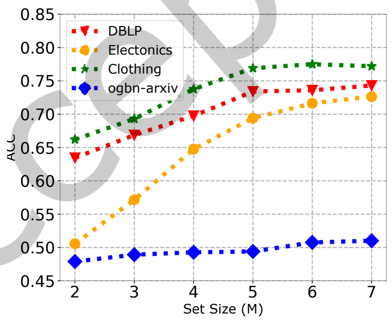


(b) 5-w 3-s (Sym $\epsilon = 30\%$)

Fig. 5. The distribution of confidence scores for different types of nodes.



(a) 5-w K-s (Sym $\epsilon = 30\%$)



(b) 5-w 1-s (Sym $\epsilon = 30\%$)

Fig. 6. Evaluation for different learning parameters in Meta-GIN.

Parameter Analysis. To further understand the model design, we analyze the sensitivity of Meta-GIN to the support size K and the task number M. Due to the space limit, here we show the results under the task of 5-way 1-shot with symmetric noise ($\epsilon = 0.3$), similar patterns can be observed for other cases. In Figure 6 (a), we summarize the performance of Meta-GIN with various support size K on ogbn-arxiv and we can observe that the proposed Meta-GIN can always achieve the best performance on different 5-way K-shot tasks. This demonstrates the superiority of Meta-GIN for solving weakly-supervised graph few-shot learning problems. Next, we investigate the performance of Meta-GIN by varying the task number M and show the results of 5-way 1-shot (Sym) on the four datasets. From Figure 6 (b), we find that by increasing M, the performance of Meta-GIN gradually improves, which indicates that interpolating more tasks is helpful for noise-reduced node representations. Also, the model performance become stable when $M \geq 5$, thus 5 is the appropriate value for M to obtain satisfying performance considering both efficiency and effectiveness.

6 CONCLUSION

In this paper, we introduce a novel graph meta-learning framework Graph Interpolation Networks (Meta-GIN) to solve few-shot learning problems under the weakly-supervised setting. Unlike existing methods, our approach does not require abundant golden labeled data from seen classes and can be meta-learned to denoise for extracting highly transferable meta-knowledge from weakly-labeled data. Essentially, Meta-GIN leverages robustness-enhanced episodic training to interpolate node representations by comparing and summarizing from weakly-labeled data in a meta-learning fashion. The empirical results over different datasets demonstrate that our proposed model can effectively generalize to unseen tasks. For future work, one potential research direction could be how to advance a graph meta-learner to incrementally learn new concepts from very few labelled samples, without forgetting the previously learned ones.

REFERENCES

[1] Jinheon Baek, Dong Bok Lee, and Sung Ju Hwang. 2020. Learning to Extrapolate Knowledge: Transductive Few-shot Out-of-Graph Link Prediction. *NeurIPS* (2020).

[2] Jatin Chauhan, Deepak Nathani, and Manohar Kaul. 2019. FEW-SHOT LEARNING ON GRAPHS VIA SUPER-CLASSES BASED ON GRAPH SPECTRAL MEASURES. In *ICLR*.

[3] Pengfei Chen, Ben Ben Liao, Guangyong Chen, and Shengyu Zhang. 2019. Understanding and utilizing deep neural networks trained with noisy labels. In *ICML*.

[4] Kaize Ding, Jundong Li, Rohit Bhanushali, and Huan Liu. 2019. Deep anomaly detection on attributed networks. In *SDM*.

[5] Kaize Ding, Jianling Wang, Jundong Li, Kai Shu, Chenghao Liu, and Huan Liu. 2020. Graph prototypical networks for few-shot learning on attributed networks. In *CIKM*.

[6] Kaize Ding, Zhe Xu, Hanghang Tong, and Huan Liu. 2022. Data augmentation for deep graph learning: A survey. *SIGKDD Explorations* (2022).

[7] Hande Dong, Jiawei Chen, Fuli Feng, Xiangnan He, Shuxian Bi, Zhaolin Ding, and Peng Cui. 2021. On the Equivalence of Decoupled Graph Convolution Network and Label Propagation. In *TheWebConf*.

[8] Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *ICML*.

[9] Zheng Gao, Hongsong Li, Zhuoren Jiang, and Xiaozhong Liu. 2020. Detecting User Community in Sparse Domain via Cross-Graph Pairwise Learning. In *SIGIR*.

[10] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *KDD*.

[11] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *NeurIPS*.

[12] Xiaorong Hao, Tao Lian, and Li Wang. 2020. Dynamic Link Prediction by Integrating Node Vector Evolution and Local Neighborhood Representation. In *SIGIR*.

[13] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang. 2020. Lightgcn: Simplifying and powering graph convolution network for recommendation. In *SIGIR*.

[14] Dan Hendrycks, Mantas Mazeika, Duncan Wilson, and Kevin Gimpel. 2018. Using trusted data to train deep networks on labels corrupted by severe noise. In *NeuIPS*.

[15] Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. 2020. Open graph benchmark: Datasets for machine learning on graphs. In *NuerIPS*.

[16] Kexin Huang and Marinka Zitnik. 2020. Graph meta learning via local subgraphs. In *NeurIPS*.

[17] Lu Jiang, Zhengyuan Zhou, Thomas Leung, Li-Jia Li, and Li Fei-Fei. 2018. Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels. In *ICML*.

[18] Thomas N Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *NeurIPS*.

[19] Lin Lan, Pinghui Wang, Xuefeng Du, Kaikai Song, Jing Tao, and Xiaohong Guan. 2020. Node Classification on Graphs with Few-Shot Novel Labels via Meta Transformed Network Embedding. In *NeurIPS*.

[20] Feng Li, Zhenrui Chen, Pengjie Wang, Yi Ren, Di Zhang, and Xiaoyu Zhu. 2019. Graph intention network for click-through rate prediction in sponsored search. In *SIGIR*.

[21] Zhenguo Li, Fengwei Zhou, Fei Chen, and Hang Li. 2017. Meta-sgd: Learning to learn quickly for few-shot learning. *arXiv preprint arXiv:1707.09835* (2017).

[22] Jianghao Lin, Weiwen Liu, Xinyi Dai, Weinan Zhang, Shuai Li, Ruiming Tang, Xiuqiang He, Jianye Hao, and Yong Yu. 2021. A Graph-Enhanced Click Model for Web Search. In *SIGIR*. 1259–1268.

[23] Lu Liu, Tianyi Zhou, Guodong Long, Jing Jiang, and Chengqi Zhang. 2019. Learning to propagate for graph meta-learning. In *NeurIPS*.

[24] Yonghao Liu, Mengyu Li, Ximing Li, Fausto Giunchiglia, Xiaoyue Feng, and Renchu Guan. 2022. Few-shot Node Classification on Attributed Networks with Graph Meta-learning. In *SIGIR*.
[25] Zemin Liu, Yuan Fang, Chenghao Liu, and Steven CH Hoi. 2021. Relative and Absolute Location Embedding for Few-Shot Node Classification on Graph. In *AAAI*.
[26] Zemin Liu, Wentao Zhang, Yuan Fang, Xinming Zhang, and Steven CH Hoi. 2020. Towards locality-aware meta-learning of tail node embeddings on networks. In *CIKM*.
[27] Ning Ma, Jiajun Bu, Jieyu Yang, Zhen Zhang, Chengwei Yao, Zhi Yu, Sheng Zhou, and Xifeng Yan. 2020. Adaptive-Step Graph Meta-Learner for Few-Shot Graph Classification. In *CIKM*.
[28] Julian McAuley, Rahul Pandey, and Jure Leskovec. 2015. Inferring networks of substitutable and complementary products. In *KDD*.
[29] Nikhil Mishra, Mostafa Rohaninejad, Xi Chen, and Pieter Abbeel. 2018. A Simple Neural Attentive Meta-Learner. In *ICLR*.
[30] Wentao Ouyang, Xiuwu Zhang, Shukui Ren, Li Li, Kun Zhang, Jinmei Luo, Zhaojie Liu, and Yanlong Du. 2021. Learning Graph Meta Embeddings for Cold-Start Ads in Click-Through Rate Prediction. In *SIGIR*.
[31] Guo-Jun Qi, Charu Aggarwal, Qi Tian, Heng Ji, and Thomas Huang. 2011. Exploring context and content links in social media: A latent space method. *TPAMI* (2011).
[32] Sachin Ravi and Hugo Larochelle. 2017. Optimization as a model for few-shot learning. In *ICLR*.
[33] Mengye Ren, Eleni Triantafillou, Sachin Ravi, Jake Snell, Kevin Swersky, Joshua B Tenenbaum, Hugo Larochelle, and Richard S Zemel. 2018. Meta-learning for semi-supervised few-shot classification. In *ICLR*.
[34] Mengye Ren, Wenyuan Zeng, Bin Yang, and Raquel Urtasun. 2018. Learning to reweight examples for robust deep learning. In *ICML*.
[35] Adam Santoro, Sergey Bartunov, Matthew Botvinick, Daan Wierstra, and Timothy Lillicrap. 2016. Meta-learning with memory-augmented neural networks. In *ICML*.
[36] Jake Snell, Kevin Swersky, and Richard Zemel. 2017. Prototypical networks for few-shot learning. In *NeurIPS*.
[37] Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip HS Torr, and Timothy M Hospedales. 2018. Learning to compare: Relation network for few-shot learning. In *CVPR*.
[38] Jie Tang, Jing Zhang, Limin Yao, Juanzi Li, Li Zhang, and Zhong Su. 2008. Arnetminer: extraction and mining of academic social networks. In *KDD*.
[39] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2018. Graph attention networks. In *ICLR*.
[40] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. 2016. Matching networks for one shot learning. In *NeurIPS*.
[41] Jianling Wang, Kaize Ding, Liangjie Hong, Huan Liu, and James Caverlee. 2020. Next-item recommendation with sequential hypergraphs. In *SIGIR*.
[42] Ning Wang, Minnan Luo, Kaize Ding, Lingling Zhang, Jundong Li, and Qinghua Zheng. 2020. Graph few-shot learning with attribute matching. In *CIKM*.
[43] Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. 2017. Knowledge graph embedding: A survey of approaches and applications. In *TKDE*.
[44] Song Wang, Kaize Ding, Chuxu Zhang, Chen Chen, and Jundong Li. 2022. Task-adaptive few-shot node classification. In *KDD*.
[45] Zhihao Wen, Yuan Fang, and Zemin Liu. 2021. Meta-Inductive Node Classification across Graphs. In *SIGIR*.
[46] Felix Wu, Tianyi Zhang, Amauri Holanda de Souza Jr, Christopher Fifty, Tao Yu, and Kilian Q Weinberger. 2019. Simplifying graph convolutional networks. *ICML* (2019).
[47] Huaxiu Yao, Chuxu Zhang, Ying Wei, Meng Jiang, Suhang Wang, Junzhou Huang, Nitesh V Chawla, and Zhenhui Li. 2020. Graph few-shot learning via knowledge transfer. In *AAAI*.
[48] Chuxu Zhang, Huaxiu Yao, Chao Huang, Meng Jiang, Zhenhui Li, and Nitesh V Chawla. 2020. Few-shot knowledge graph completion. In *AAAI*.
[49] Jian Zhang, Chenglong Zhao, Bingbing Ni, Minghao Xu, and Xiaokang Yang. 2019. Variational few-shot learning. In *ICCV*.
[50] Shichao Zhang and Jiaye Li. 2021. Knn classification with one-step computation. *TKDE* (2021).
[51] Shichao Zhang, Jiaye Li, and Yangding Li. 2022. Reachable distance function for KNN classification. *TKDE* (2022).
[52] Tiantian Zhang and Bin Wu. 2012. A method for local community detection by finding core nodes. In *ASONAM*.
[53] Weihe Zhang, Yali Wang, and Yu Qiao. 2019. Metacleaner: Learning to hallucinate clean representations for noisy-labeled visual recognition. In *CVPR*.
[54] Fan Zhou, Chengtai Cao, Kunpeng Zhang, Goce Trajcevski, Ting Zhong, and Ji Geng. 2019. Meta-GNN: On Few-shot Node Classification in Graph Meta-learning. In *CIKM*.
[55] Linchao Zhu and Yi Yang. 2018. Compound memory networks for few-shot video classification. In *ECCV*.