Few-shot Node Classification with Extremely Weak Supervision

Song Wang
University of Virginia
sw3wv@virginia.edu

Yushun Dong
University of Virginia
yd6eb@virginia.edu

Kaize Ding
Arizona State University
kding9@asu.edu

Chen Chen
University of Virginia
zrh6du@virginia.edu

Jundong Li
University of Virginia
jundong@virginia.edu

ABSTRACT

Few-shot node classification aims at classifying nodes with limited labeled nodes as references. Recent few-shot node classification methods typically learn from classes with abundant labeled nodes (i.e., meta-training classes) and then generalize to classes with limited labeled nodes (i.e., meta-test classes). Nevertheless, on real-world graphs, it is usually difficult to obtain abundant labeled nodes for many classes. In practice, each meta-training class can only consist of several labeled nodes, known as the *extremely weak supervision* problem. In few-shot node classification, with extremely limited labeled nodes for meta-training, the generalization gap between meta-training and meta-test will become larger and thus lead to suboptimal performance. To tackle this issue, we study a novel problem of few-shot node classification with extremely weak supervision and propose a principled framework X-FNC under the prevalent meta-learning framework. Specifically, our goal is to accumulate meta-knowledge across different meta-training tasks with extremely weak supervision and generalize such knowledge to meta-test tasks. To address the challenges resulting from extremely scarce labeled nodes, we propose two essential modules to obtain pseudo-labeled nodes as extra references and effectively learn from extremely limited supervision information. We further conduct extensive experiments on four node classification datasets with extremely weak supervision to validate the superiority of our framework compared to the state-of-the-art baselines.

CCS CONCEPTS

• **Information systems** → **Data mining**; • **Computing methodologies** → **Transfer learning**.

KEYWORDS

Graph Neural Networks; Few-shot Learning; Weak Supervision

ACM Reference Format:
Song Wang, Yushun Dong, Kaize Ding, Chen Chen, and Jundong Li. 2023. Few-shot Node Classification with Extremely Weak Supervision. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining (WSDM '23), February 27-March 3, 2023, Singapore, Singapore.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3539597.3570435

WSDM '23, February 27-March 3, 2023, Singapore, Singapore

ACM ISBN 978-1-4503-9407-9/23/02.
https://doi.org/10.1145/3539597.3570435

1 INTRODUCTION

Node classification focuses on learning a model that can assign labels for unlabeled nodes on a graph [4, 14, 15]. Many real-world analytical tasks can be formulated as the node classification problem [7, 19]. For example, in disease diagnosis, the types of diseases are regarded as class labels while patients are represented by nodes on a patient similarity graph [20]. Recent studies mainly leverage Graph Neural Networks (GNNs) [34, 41] to learn node representations and classify unlabeled nodes based on the learned presentations. However, GNNs typically require a considerable number of labeled nodes for all classes to learn effective node representations [6, 43]. In practice, it is often difficult to obtain sufficient labeled nodes for each class as the labeling process requires a lot of human efforts [5, 38]. Hence, there is a surge of research interests aiming at performing node classification with limited labeled nodes as references, known as *few-shot node classification*.

To effectively solve the few-shot node classification problem, many recent works adopt a meta-learning strategy [4, 14, 19]. In specific, these works learn transferable knowledge from classes with abundant labeled nodes (i.e., *meta-training classes*) and then generalize such knowledge to other classes with limited labeled nodes (i.e., *meta-test classes*). The overall process is conducted on a series of meta-tasks (i.e., meta-training tasks and meta-test tasks), where each meta-task contains a small number of *support nodes* as references and several *query nodes* to be classified. Despite their empirical success, existing approaches [6, 14, 43] simply assume that meta-training classes consist of abundant labeled nodes, i.e., all the nodes in the meta-training classes are gold-labeled. However, such an assumption is generally unrealistic in practice, since each class may only consist of an extremely limited number of labeled nodes on real-world graphs. For example, in molecular property prediction, certain chemical properties (i.e., classes) only consist of extremely limited labeled molecules due to the expensive cost of the labeling process [11]. With extremely inadequate labeled nodes for each meta-training class (i.e., extremely weak supervision [8]), the effectiveness of the meta-learning paradigm for learning transferable meta-knowledge will be severely impacted. Thus, solving the problem of few-shot node classification with extremely limited labeled nodes for each meta-training class requires urgent research efforts. In this regard, we investigate a novel problem of *few-shot node classification with extremely weak supervision* in this paper. Specifically, the goal is to perform few-shot node classification after learning from extremely limited labeled nodes for each class.

However, it remains a challenging task to achieve this goal due to two major reasons. First, with extremely weak supervision, the

model performance will be deteriorated by the *under-generalizing* problem due to extremely inadequate support nodes. Specifically, given extremely limited labeled nodes for each meta-training class, the support nodes in each meta-training task are only sampled from a small set of labeled nodes. Therefore, the effectiveness of meta-learning in extracting meta-knowledge from different classes will be greatly weakened. As a result, the generalizability of the model to meta-test classes will drop significantly (i.e., under-generalizing). Second, with extremely weak supervision, the meta-training efficacy will be severely impacted by the *over-fitting* problem due to extremely inadequate query nodes. Recent few-shot node classification studies [4, 6, 19, 43] generally require a large number of query nodes during meta-training for model optimization. Nevertheless, with extremely weak supervision from the meta-training classes, the number of query nodes for optimization during meta-training is significantly reduced. As a result, the model will be easily over-fitted and result in suboptimal performance.

To tackle the aforementioned challenges, we propose a novel framework for few-shot node classification with extremely weak supervision from the meta-training classes, named as *X-FNC*. Essentially, our framework consists of two innovative modules to handle the *under-generalizing* and *over-fitting* issues, respectively. First, to compensate for the insufficient support nodes during meta-training, we perform label propagation to obtain abundant pseudo-labeled nodes based on Poisson Learning [3]. With the pseudo-labeled nodes, we can expand the support set in each meta-task to better extract discriminative meta-knowledge for each class. Second, to alleviate the negative impact of over-fitting caused by inadequate query nodes, we propose to optimize the model by both classifying nodes and filtering out irrelevant information (e.g., decisive classification information for classes not used in a meta-task) based on Information Bottleneck (IB) [33]. As a result, in addition to learning with supervision information, the model will also learn to ignore irrelevant information during the meta-learning process, which relieves over-fitting caused by insufficient supervision information. In summary, our main contributions are as follows:

- We investigate a novel research problem of few-shot node classification with extremely weak supervision.
- We develop a novel few-shot node classification framework under the extremely weak supervision scenario with two essential modules: (1) a label propagation module based on Poisson Learning to expand the support set in each meta-task by obtaining pseudo-labeled nodes; (2) an optimization strategy based on Information Bottleneck to learn from classifying query nodes while reducing irrelevant information.
- We conduct extensive experiments on four node classification datasets with extremely weak supervision. Experimental results demonstrate the superiority of our framework.

2 RELATED WORK

2.1 Few-shot Node Classification

Few-shot learning aims to achieve considerable classification performance using limited labeled samples as references. The general approach is to accumulate transferable knowledge from tasks with abundant labeled samples and then generalize such knowledge to novel tasks with few labeled samples. Generally, there are two main categories of approaches for few-shot learning: (1) *Metric-based* approaches focus on learning a metric function to match the query set with the support set for classification [17, 28]. For example, Prototypical Networks [27] learn prototypes for classes and classify query samples based on the Euclidean distances between the query set and the prototypes. (2) *Optimization-based* approaches aim to optimize model parameters based on gradients on support samples in each meta-task [22, 23, 25]. As a classic example, MAML [9] learns the parameter initialization for different meta-tasks with the proposed meta-optimization strategy. On graph data, many research efforts have been devoted to studying few-shot learning on graphs with limited labeled nodes [29, 30, 36, 39]. For example, Meta-GNN [43] combines meta-learning [9] with GNNs to reduce the requirement of labeled nodes. GPN [6] estimates node importance and leverages Prototypical Networks [27] for few-shot node classification. TENT [37] proposes to reduce the variance among tasks for generalization performance.

2.2 Semi-supervised Few-shot Learning

Several recent approaches aim to combine semi-supervised or self-supervised learning with few-shot learning to improve the performance on few-shot classification tasks with unlabeled data. Ren et al. [26] extend Prototypical Networks with unlabeled data based on the Soft k-Means method. TPN [18] propagates labels of given data to unlabeled data, combined with a meta-learning strategy for optimization. On the other hand, the Information Bottleneck (IB) principle is also leveraged in self-supervised representation learning. DVIB [1] first utilizes IB in neural networks for robust representation learning. Moreover, GIB [40] develops information-theoretic modeling of graph structures and node features on graph representation learning.

3 PRELIMINARIES

3.1 Few-shot Node Classification

We denote an attributed graph as $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{X})$, where $\mathcal{V}$ and $\mathcal{E}$ denote the set of nodes and edges, respectively. $\mathbf{X} \in \mathbb{R}^{|\mathcal{V}| \times d}$ is the node feature matrix, where d is the feature dimension. Moreover, we denote the set of node classes as C, which can be further divided into two sets: C_{train} and C_{test}. Note that $C = C_{train} \cup C_{test}$ and $C_{train} \cap C_{test} = \emptyset$, where C_{train} and C_{test} denote the set of meta-training and meta-test classes, respectively. General few-shot settings assume that labeled nodes in C_{train} are abundant, while labeled nodes in C_{test} are generally scarce. However, it is usually unrealistic in practice to obtain adequate labeled nodes for all classes in C_{train}. With extremely weak supervision, the number of labeled nodes in C_{train} is severely limited. Subsequently, our goal is to develop a learning model such that after meta-training on extremely limited labeled nodes, the model can accurately predict labels for the nodes in C_{test} with only K labeled nodes for each of N randomly sampled classes as the reference. In this way, the problem is called N-way K-shot node classification.

3.2 N-way K-shot Meta-learning

We follow the prevalent episodic meta-learning paradigm, which has demonstrated superior performance in few-shot learning [9, 27,

35]. Particularly, we employ C_{train} and C_{test} for meta-training and meta-test, respectively. During meta-training, the model learns from a series of *meta-training tasks*. Each meta-training task consists of a support set $\mathcal{S}$ as the reference and a query set $\mathcal{Q}$ to be classified. Here $\mathcal{S} = \{(v_1, y_1), (v_2, y_2), \ldots, (v_{N\times K}, y_{N\times K})\}$ contains N classes randomly sampled from C_{train} and K labeled nodes for each of these N classes (i.e., N-way K-shot). $v_i \in \mathcal{V}$ is a node in $\mathcal{G}$ and y_i is the class of v_i. The query set $\mathcal{Q} = \{(v_1^*, y_1^*), (v_2^*, y_2^*), \ldots, (v_Q^*, y_Q^*)\}$ consists of totally Q different nodes from these N classes. Note that during the classification process in each meta-task, all nodes on the graph other than nodes in this meta-task are considered unlabeled and can be leveraged to advance the classification performance. During meta-test, the model is evaluated on meta-test tasks, which share a similar structure with meta-training tasks, except that the classes are in C_{test}. Under the meta-learning [9, 14, 43] framework, we first fine-tune the model based on support nodes and then conduct classification on query nodes.

4 THE PROPOSED FRAMEWORK

We first present an overview of our proposed framework X-FNC. Specifically, we formulate the problem of *few-shot node classification with extremely weak supervision* under the prevalent N-way K-shot meta-learning framework. In practice, we conduct meta-training on a series of randomly sampled meta-tasks, where a meta-task contains K nodes for each of N classes as the support set and several query nodes to be classified. Due to the extremely limited labeled nodes during meta-training, the model performance will be severely deteriorated by two problems: *under-generalizing* (caused by inadequate support nodes) and *over-fitting* (caused by inadequate query nodes). Therefore, we propose two essential modules: *Poisson Label Propagation* and *Information Bottleneck Fine-tuning*, which solve these two problems by obtaining pseudo-labeled nodes and maximally learning decisive information, respectively.

4.1 Poisson Label Propagation

To alleviate the problem of under-generalizing caused by extremely limited support nodes during meta-training, we propose to obtain pseudo-labeled nodes based on Poisson Learning [3]. Specifically, Poisson Learning is recently proposed to propagate labels from relatively limited labeled samples to unlabeled samples, based on the assumption that samples that are close to each other can potentially share similar classes. By recursively aggregating label information from close samples, the unlabeled samples can be pseudo-labeled based on label information propagated from labeled samples. Thus, it is helpful for obtaining pseudo-labeled nodes under the extremely weak supervision setting. However, it remains non-trivial to perform Poisson Learning on few-shot node classification with extremely weak supervision due to the following two reasons. First, Poisson Learning cannot fully take advantage of structural information on graph data when leveraged to obtain pseudo-labeled nodes. Originally proposed for image classification, Poisson Learning constructs a graph solely based on the Euclidean distances between images. However, on graph data, the graph structures encode crucial information for node classification and thus cannot be ignored. Second, Poisson Learning cannot effectively handle a varying class set. Few-shot learning models are required to deal with various classes across different meta-tasks, which contradicts the fact that Poisson Learning is originally proposed to operate on a fixed class set. Nevertheless, the ability to handle various classes is crucial for few-shot node classification [6, 14].

To overcome these two difficulties, as illustrated in Fig. 1, we propose to construct a subgraph in each meta-task based on graph structures and node features, and such subgraph consists of support nodes and randomly sampled unlabeled nodes. In addition, we include the neighbors of these nodes and the corresponding edges in the subgraph to effectively leverage local structures of support nodes. Moreover, we utilize the constructed subgraph in each meta-task instead of the entire graph, so that we can perform label propagation regarding the varying classes in different meta-tasks.

Specifically, consider a meta-task $\mathcal{T} = (\mathcal{S}, \mathcal{Q})$. We first aim to sample a number of unlabeled nodes for label propagation based on Poisson Learning. Basically, the neighbors of nodes in $\mathcal{S}$ bear a higher chance of belonging to classes in $\mathcal{S}$ than other random nodes. In addition, only using these neighboring nodes can be insufficient when the average node degree is small. Therefore, we sample unlabeled nodes via two strategies: neighbor sampling and random sampling. For the neighbor sampling, we select 2-hop neighbors of the labeled nodes in $\mathcal{S}$, since neighboring nodes maintain explicit connections to these labeled nodes and are thus more likely to share the same classes. In particular, denoting the set of 2-hop neighbors of node v_i as $\mathcal{N}_i$, the node set obtained via neighbor sampling is $\mathcal{V}_n = \bigcup_{i=1}^{NK} \mathcal{N}_i$. For the random sampling, we randomly select R nodes from the remaining node set $\mathcal{V} \setminus (\mathcal{S} \cup \mathcal{V}_n)$ to form a random node set $\mathcal{V}_r$, where $|\mathcal{V}_r| = R$. Then similarly, we extract the 2-hop neighbors of nodes in $\mathcal{V}_r$ as $\mathcal{V}_{rn}$. In consequence, combining nodes sampled from the two sampling strategies, we can obtain the final node set $\mathcal{V}_s = \mathcal{S} \cup \mathcal{V}_n \cup \mathcal{V}_r \cup \mathcal{V}_{rn}$ for the subgraph.

Nevertheless, the sampled nodes in $\mathcal{V}_s$ can be distributed across the entire graph and potentially unconnected, which greatly hinders the process of label propagation. Therefore, we propose to construct a subgraph with these nodes based on both the structural and feature information. More specifically, we first extract the corresponding edge set $\mathcal{E}_s$ from $\mathcal{E}$ according to $\mathcal{V}_s$. Then we denote $\mathbf{A}' \in \mathbb{R}^{|\mathcal{V}_s|\times|\mathcal{V}_s|}$ as the adjacency matrix obtained from graph structures (i.e., $\mathcal{E}_s$), where $\mathbf{A}'_{ij} = 1$ if the i-th node in $\mathcal{V}_s$ connects to the j-th node in $\mathcal{V}_s$, and $\mathbf{A}'_{ij} = 0$, otherwise. In this way, we can construct edges without losing the original structural information. Furthermore, to incorporate feature information, we propose to compute another edge weight matrix based on Euclidean distances between node features [3] as follows:

$$\mathbf{A}''_{ij} = \exp\left(-\eta\|\mathbf{x}_i - \mathbf{x}_j\|\right), \tag{1}$$

where $\eta \in \mathbb{R}$ is a hyper-parameter to control the scale of $\mathbf{A}''$ and $\|\cdot\|$ is the ℓ_2-norm. In this way, all nodes in $\mathcal{V}_s$ are also connected according to their distances, which further advances the label propagation. Finally, we combine the two matrices to form the final adjacency matrix: $\mathbf{A} = \lambda\mathbf{A}' + (1-\lambda)\mathbf{A}''$ with a scaling hyper-parameter $\lambda \in [0, 1]$. As a result, the edges can absorb information from both graph structures and node features, which effectively promotes the label propagation process based on Poisson Learning on this subgraph. Then with the learned adjacency matrix $\mathbf{A}$, we can perform Poisson Learning on this subgraph to obtain pseudo-labeled nodes.

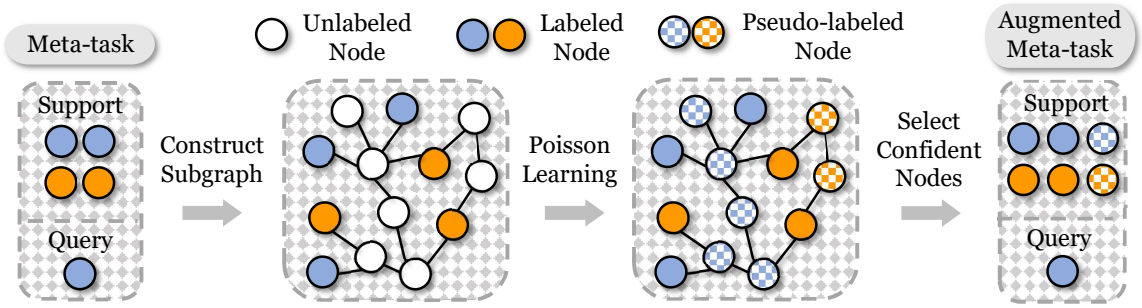

Figure 1: The pseudo-labeling process with our Poisson Label Propagation module. For each meta-task, we construct a subgraph based on support nodes and randomly sampled unlabeled nodes, including their neighboring nodes. Then we perform Poisson Label Propagation to obtain pseudo-labeled nodes. After that, we further select pseudo-labeled nodes with high confidence to form the augmented support set. The augmented support set will be used for fine-tuning in this meta-task.

Denote $\mathbf{u}_i \in \mathbb{R}^N$ as the label vector of the i-th node v_i in $\mathcal{V}_s$ to be learned, where the index of the largest element in $\mathbf{u}_i$ indicates that v_i belongs to this class. Intuitively, Poisson Learning [3] assumes that the label vector of an unlabeled node is the weighted average of its neighbors' label vectors, where the weight is from the corresponding entry in $\mathbf{A}$. Moreover, the label vectors of given labeled nodes are their corresponding classes minus the average label vector of all labeled nodes. In this way, the objective of Poisson Learning can be formulated as follows:

$$\begin{cases} \sum_{j=1}^{|\mathcal{V}_s|} \mathbf{A}_{ij} \left(\mathbf{u}_i - \mathbf{u}_j\right) = 0, & \text{if } NK+1 \le i \le |\mathcal{V}_s|, \\ \mathbf{u}_i = \mathbf{y}_i - \bar{\mathbf{y}}, & \text{if } 1 \le i \le NK, \end{cases}, \tag{2}$$

satisfying $\sum_{i=1}^{|\mathcal{V}_s|} d_i \mathbf{u}_i = 0$, where $d_i = \sum_{j=1}^{|\mathcal{V}_s|} \mathbf{A}_{ij}$. $\mathbf{y}_i \in \mathbb{R}^N$, where the j-th element is 1 if $\mathbf{x}_i$ belongs to the j-th class, and other elements are 0. $\bar{\mathbf{y}} = \sum_{i=1}^{NK} \mathbf{y}_i / NK$ is the average label vector. To solve Eq. (2), we iteratively update the prediction matrix $\mathbf{U} \in \mathbb{R}^{\mathcal{V}_s \times N}$ based on [3] as follows:

$$\mathbf{U}^{(t)} \leftarrow \mathbf{U}^{(t-1)} + \mathbf{D}^{-1}\left(\mathbf{B}^\top - \mathbf{L}\mathbf{U}^{(t-1)}\right), \tag{3}$$

where $t \in \{1, 2, \ldots, T_l\}$ and T_l is the number of label propagation steps. $\mathbf{D}$ is a diagonal matrix and $\mathbf{D}_{ii} = \sum_{j=1}^{|\mathcal{V}_s|} \mathbf{A}_{ij}$. $\mathbf{L} = \mathbf{D} - \mathbf{A}$ is the unnormalized Laplacian matrix, and $\mathbf{B} = [\mathbf{F} - \bar{\mathbf{y}}, \mathbf{O}]$, where $\mathbf{O} \in \mathbb{R}^{N \times (|\mathcal{V}_s| - NK)}$ is a zero matrix. $\mathbf{F} \in \mathbb{R}^{N \times NK}$ denotes the label matrix of the NK labeled nodes, whose i-th column is $\mathbf{y}_i$. The i-th row of the final result $\mathbf{U}^{(T_l)}$ is the obtained label vector of v_i. The iteration is achieved by replacing the label vector (i.e., $\mathbf{u}_i$) with the weighted average of label vectors from neighboring nodes of v_i.

In this way, we can obtain a considerable number of pseudo-labeled nodes to compensate for the lack of support nodes during meta-training. Nevertheless, some of the pseudo-labeled nodes could be incorrect and thus deteriorate the classification performance if all pseudo-labeled nodes are used to expand the support set. Therefore, we propose to select pseudo-labeled nodes with high prediction confidence for fine-tuning in each meta-task. Specifically, we compute the confidence score for each pseudo-labeled node according to the entropy of the prediction result as follows:

$$c_i = -\sum_{j=1}^{N} u_{ij} \log u_{ij}, \tag{4}$$

where u_{ij} is the j-th element of $\mathbf{u}_i$ after softmax. In this way, we can select top M pseudo-labeled nodes with the highest confidence scores. Note that $\mathcal{V}_s$ contains the NK support nodes, which will be ignored during the selection since they are already labeled. Then the support set can be augmented as $\widetilde{\mathcal{S}} = \mathcal{S} \cup \mathcal{S}_p$, where $\mathcal{S}_p$ is the set of selected pseudo-labeled nodes and $|\widetilde{\mathcal{S}}| = NK + M$.

4.2 Information Bottleneck Fine-tuning

With the augmented support set $\widetilde{\mathcal{S}}$, we can conduct fine-tuning on $\widetilde{\mathcal{S}}$ for fast adaptations to the given meta-task $\mathcal{T}$ and then meta-optimize the model on the query set $\mathcal{Q}$. However, although the support set is augmented, the query set $\mathcal{Q}$ could still be inadequate for optimization with extremely weak supervision. In other words, the model can be easily influenced by irrelevant information (e.g., decisive classification information for classes not in $\mathcal{T}$) and thus leads to *over-fitting*. Therefore, we aim to fine-tune the model with extremely limited query nodes while ignoring the irrelevant information as much as possible. Particularly, the Information Bottleneck (IB) [33] provides an essential principle to extract classification information while maximally reducing the negative impact of irrelevant information. Moreover, the IB principle can also encourage the model to benefit from incorrect pseudo-labeled nodes by learning to neglect irrelevant information. Nevertheless, it remains non-trivial to utilize the IB principle on graph data, due to the fact that graph data does not follow the i.i.d. assumption used in previous IB-based models [40]. Thus, we further derive two variational bounds for IB to fine-tune the model in a more tractable manner.

Specifically, the objective of the IB principle can be formulated as follows:

$$\min \mathrm{IB}(D, Y; Z) \triangleq [-I(Y; Z) + \beta I(D; Z)], \tag{5}$$

where Y denotes the class set of nodes in $\widetilde{\mathcal{S}}$ and $|Y| = N$. Z denotes the node representations to be learned. D denotes the structural and feature information of nodes in $\widetilde{\mathcal{S}}$. β is a positive scalar to balance the trade-off between the desire to preserve classification information and being invariant to irrelevant graph structures and node features [42]. In particular, the IB aims to learn representations that are maximally informative for classification (i.e., maximizing $I(Y; Z)$) while reducing irrelevant information (i.e., minimizing $I(D; Z)$). Furthermore, it becomes more useful in few-shot learning

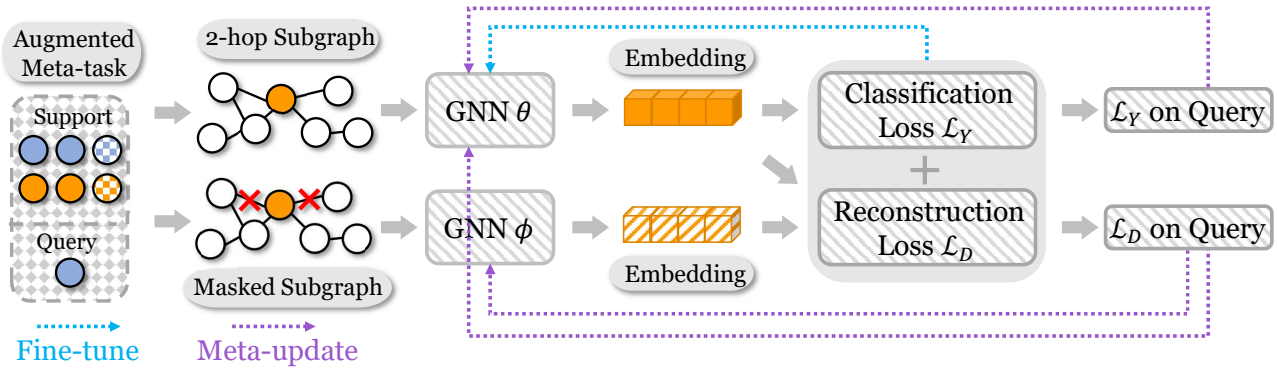

Figure 2: The optimization process with our IB fine-tuning module and meta-learning strategy. For each node in the augmented support set, we construct a 2-hop subgraph and a masked 2-hop subgraph. Then we utilize two GNN$_\theta$ and GNN$_\phi$ to perform IB fine-tuning via T steps. After that, we calculate the loss on the query set and meta-update model parameters.

since each meta-task is only conducted on N classes, and thus the irrelevant information D can be more redundant.

Nevertheless, it is difficult to directly optimize the objective in Eq. (5), since it is intractable [40]. Thus, we propose to derive an upper bound for each of the two terms in Eq. (5) for optimization. Specifically, the first term can be expressed using entropy as follows:

$$-I(Y;Z) = -\left[H(Y) - H(Y|Z)\right], \tag{6}$$

where H is the entropy. Since we aim to optimize the model for better Z, we can ignore the unrelated term $H(Y)$. Then we can obtain the explicit form of $H(Y|Z)$ based on the definition of entropy:

$$\begin{aligned} H(Y|Z) &= -\sum_{i=1}^{N}\sum_{j=1}^{|\widetilde{\mathcal{S}}|} p(y_i, z_j)\log\frac{p(y_i,z_j)}{p(z_j)} \\ &= -\sum_{i=1}^{N}\sum_{j=1}^{|\widetilde{\mathcal{S}}|} p(z_j|y_i)p(y_i)\log p(y_i|z_j), \end{aligned} \tag{7}$$

where y_i and z_i denote the label and the representation of the i-th node v_i in $\widetilde{\mathcal{S}}$, respectively. Since each meta-task contains K support nodes for each of N classes, we can assume that the prior distribution of Y is uniform, and thus $p(y_i)$ is a constant. To further estimate $p(z_j|y_i)$, we compute it via $p(z_j|y_i) = \mathbb{1}(z_j \in y_i)$, where $\mathbb{1}(z_j \in y_i) = 1$ if z_j belongs to y_i; otherwise $\mathbb{1}(z_j \in y_i) = 0$. In this way, the objective of $-I(Y;Z)$ is formulated as a cross-entropy loss:

$$-I(Y;Z) \to \mathcal{L}_Y = -\sum_{i=1}^{\widetilde{\mathcal{S}}} \log p(y_i'|z_i), \tag{8}$$

where y_i' denotes the specific label that the i-th node v_i belongs to. Then to estimate $p(y_i'|z_i)$, we further utilize a GNN$_\theta$ followed by an MLP classifier MLP$_\theta$. Specifically, for the i-th node v_i in $\widetilde{\mathcal{S}}$, we extract its 2-hop neighboring nodes to form a subgraph, represented by $(\mathbf{A}_i, \mathbf{X}_i)$. Here $\mathbf{A}_i$ and $\mathbf{X}_i$ denote the adjacency and feature matrix, respectively. Then we compute the output prediction score as

$$\mathbf{s}_i = \text{MLP}_\theta\left(\text{GNN}_\theta(\mathbf{A}_i, \mathbf{X}_i)\right), \tag{9}$$

where $\mathbf{s}_i \in \mathbb{R}^N$ is the unnormalized prediction score of v_i. With a softmax function, we can normalize $\mathbf{s}_i$ to finally obtain $p(y_j'|z_j)$. In this way, the model learns the crucial information for classification of classes in Y via maximizing $I(Y;Z)$.

For another term $I(D;Z)$, we first express it via the expectation:

$$I(D;Z) = \mathbb{E}\left(\log\frac{p(Z|D)}{p(Z)}\right), \tag{10}$$

where D denotes the structural and feature information of nodes in $\widetilde{\mathcal{S}}$. It is noteworthy that nodes on graphs do not follow the i.i.d. assumption and thus are inherently correlated. Hence, although the fine-tuning is conducted on a specific meta-task, D should incorporate the information from the entire graph due to the correlations among nodes (i.e., $D = (\mathcal{E}, \mathbf{X})$). However, it is difficult to estimate $p(Z|D)$, since the only D is represented by the entire graph. Therefore, we introduce another distribution $q(Z)$ to approximate the true posterior $p(Z|D)$. In this way, we can further derive an upper bound of $I(D;Z)$ for optimization:

$$\begin{aligned} \mathbb{E}\left(\log\left(\frac{p(Z|D)}{q(Z)}\frac{q(Z)}{p(Z)}\right)\right) &= \mathbb{E}\left(\log\frac{p(Z|D)}{q(Z)}\right) - \text{KL}\left(p(Z)||q(Z)\right) \\ &\le \mathbb{E}\left(\log\frac{p(Z|D)}{q(Z)}\right), \end{aligned} \tag{11}$$

where KL$(\cdot||\cdot)$ denotes the KL-divergence of distributions. In this way, the final objective is to minimize the KL-divergence between $p(Z|D)$ and $q(Z)$. In practice, to estimate $p(Z|D)$, we utilize GNN$_\theta$ to instantiate $p(Z|D)$. However, incorporating structural and feature information from the entire graph can be inefficient and redundant for classification in a meta-task. Thus, we leverage the local-dependence assumption [40] of graph data to define D as the specific structural and feature information of each node in $\widetilde{\mathcal{S}}$. In this way, GNN$_\theta$ can learn to provide a comprehensive estimation for $p(Z|D)$ based on the specific D in each meta-task, since D changes with $\widetilde{\mathcal{S}}$ in different meta-tasks. On the other hand, $q(Z)$ is a prior distribution for Z and is thus difficult to estimate. Therefore, we propose to instantiate $q(Z)$ with another GNN parameterized by ϕ (i.e., GNN$_\phi$). Meanwhile, since $q(Z)$ is not conditioned on D, it is necessary to alleviate the inevitable influence of D. Therefore, we propose to randomly mask the corresponding graph structures and node features in the subgraph $(\mathbf{A}_i, \mathbf{X}_i)$ of each node in $\widetilde{\mathcal{S}}$. Specifically, for a subgraph represented by $(\mathbf{A}_i, \mathbf{X}_i)$, each entry in $\mathbf{A}_i$ and $\mathbf{X}_i$ has a probability of γ to be masked (i.e., becomes zero), and the masked matrices are denoted as $(\widetilde{\mathbf{A}}_i, \widetilde{\mathbf{X}}_i)$. As a result, the model can

learn to extract the decisive information for classification while maximally ignoring irrelevant information in D. Then for the i-th node v_i in $\widetilde{\mathcal{S}}$, as illustrated in Fig. 2, we can achieve the two representations obtained by GNN_θ and GNN_ϕ as follows:

$$\mathbf{h}_i = \text{GNN}_\theta(\mathbf{A}_i, \mathbf{X}_i), \quad \widetilde{\mathbf{h}}_i = \text{GNN}_\phi(\widetilde{\mathbf{A}}_i, \widetilde{\mathbf{X}}_i), \tag{12}$$

where $\mathbf{h}_i$ and $\widetilde{\mathbf{h}}_i$ denote the representations of v_i from the two GNNs, respectively. To minimize the KL-divergence between $p(Z|D)$ and $q(Z)$, we utilize a predictor [10, 32] p_θ (a two-layer MLP) that uses $\mathbf{h}_i$ to produce a prediction $p_\theta(\mathbf{h}_i)$ for $\widetilde{\mathbf{h}}_i$. After normalizing both $p_\theta(\mathbf{h}_i)$ and $\widetilde{\mathbf{h}}_i$, the mean squared error can be defined as follows:

$$\text{MSE}(p_\theta(\mathbf{h}_i), \widetilde{\mathbf{h}}_i) = \left\| \frac{p_\theta(\mathbf{h}_i)}{\|p_\theta(\mathbf{h}_i)\|} - \frac{\widetilde{\mathbf{h}}_i}{\|\widetilde{\mathbf{h}}_i\|} \right\|^2 = 2 - 2 \cdot \frac{p_\theta(\mathbf{h}_i) \cdot \widetilde{\mathbf{h}}_i}{\|p_\theta(\mathbf{h}_i)\| \|\widetilde{\mathbf{h}}_i\|}, \tag{13}$$

where $\|\cdot\|$ denotes the ℓ_2-norm. In this way, the loss becomes:

$$I(D; Z) \rightarrow \mathcal{L}_D = -\sum_{i=1}^{\widetilde{\mathcal{S}}} \frac{p_\theta(\mathbf{h}_i) \cdot \widetilde{\mathbf{h}}_i}{\|p_\theta(\mathbf{h}_i)\| \|\widetilde{\mathbf{h}}_i\|}, \tag{14}$$

which is the cosine similarity between $p_\theta(\mathbf{h}_i)$ and $\widetilde{\mathbf{h}}_i$. Then the final fine-tuning loss can be defined as $\mathcal{L} = \mathcal{L}_Y + \beta \mathcal{L}_D$, where β is the hyper-parameter in the IB principle to trade off the two mutual information terms.

4.3 Meta Learning-based Optimization

In this part, we elaborate on the optimization process of X-FNC. As illustrated in Fig. 2, our optimization process consists of two main stages: fine-tuning and meta-optimization. Given a specific meta-task $\mathcal{T}$, we first obtain the augmented support set $\widetilde{\mathcal{S}}$ via the proposed Poisson Label Propagation module introduced in Sec. 4.1. Then we fine-tune our framework on $\widetilde{\mathcal{S}}$ for a fast adaptation to this meta-task. Furthermore, to ensure adaptations to each meta-task during evaluation, we utilize the prevalent strategy [9, 16], which meta-optimizes model parameters according to loss on the query set. The original strategy is proposed to optimize an entire model with one meta-learning rate. However, X-FNC consists of multiple modules with various purposes. Therefore, we propose to separately optimize modules in X-FNC based on different losses.

Specifically, let θ denote the total parameters of GNN_θ, MLP_θ, and the predictor p_θ. For the fine-tuning process, we first initialize the parameters for fine-tuning as $\theta_0 \leftarrow \theta$. Then we conduct T steps of fine-tuning based on the loss $\mathcal{L}$ calculated on $\widetilde{\mathcal{S}}$ as follows:

$$\theta_t \leftarrow \theta_{t-1} - \alpha \nabla_{\theta_{t-1}} \mathcal{L}\left(\widetilde{\mathcal{S}}; \theta_{t-1}\right), \tag{15}$$

where $t \in \{1, 2, \ldots, T\}$ and $\mathcal{L}(\widetilde{\mathcal{S}}; \theta_{t-1})$ denotes that the loss is calculated based on $\widetilde{\mathcal{S}}$ with the parameters θ_{t-1}. α is the learning rate in each fine-tuning step.

It is noteworthy that during fine-tuning, other parameters of our framework (i.e., GNN_ϕ parameterized by ϕ) are kept unchanged, since simultaneously optimizing GNN_θ and GNN_ϕ can result in collapse (e.g., a constant representation) [10, 32]. After T steps of fine-tuning, we will meta-optimize GNN_ϕ with the loss calculated on the query set $\boldsymbol{Q}$. Meanwhile, since different modules bear various purposes, we optimize them with two meta-learning rates and losses. More specifically, on the query set $\boldsymbol{Q}$, we meta-optimize θ and ϕ with the following update functions:

$$\theta =: \theta - \beta_1 \nabla_\theta \mathcal{L}(\boldsymbol{Q}; \theta_T), \quad \phi =: \phi - \beta_2 \nabla_\phi \mathcal{L}_D(\boldsymbol{Q}; \theta_T), \tag{16}$$

where β_1 and β_2 are meta-learning rates for θ and ϕ, respectively. Note that GNN_ϕ is only used to calculate $\mathcal{L}_D$. Thus, GNN_ϕ will be meta-optimized regarding $\mathcal{L}_D$ instead of $\mathcal{L}$, while θ (i.e., parameters of GNN_θ, MLP_θ, and p_θ) is meta-optimized based on $\mathcal{L}$.

Moreover, it is noteworthy that our framework does not explicitly prevent collapse with extra operations (e.g., the negative samples used in contrastive learning [12, 24]) when minimizing $\mathcal{L}_D$. Nevertheless, the loss design of our framework naturally avoids converging to a minimum regarding both θ and ϕ (e.g., a trivial constant representation). Different from BYOL [10] and BGRL [32], which utilize a momentum strategy, we propose two different losses (i.e., $\mathcal{L}_Y$ and $\mathcal{L}_D$) for meta-optimization regarding θ and ϕ. As a result, the meta-optimization targets are different for θ and ϕ and thus will not cause collapse during meta-optimization. In addition, the collapse will also not occur during fine-tuning since only θ is updated while ϕ remains unchanged in this step.

After meta-training on a specific number of meta-training tasks, we evaluate the performance of our framework X-FNC on the meta-test tasks, which are sampled from $\mathcal{C}_{test}$.

5 EXPERIMENTAL EVALUATIONS

5.1 Datasets

To evaluate the performance of X-FNC on few-shot node classification with extremely weak supervision, we conduct experiments on four prevalent real-world graph datasets: `Amazon-E` [21], `DBLP` [31], `Cora-full` [2], and `ogbn-arxiv` [13]. Each dataset is a graph and consists of a considerable number of node classes to ensure that the meta-test tasks contain a variety of classes for a more comprehensive evaluation. Specifically, we obtain `Amazon-E` and `DBLP` datasets from [6]. `Cora-full` and `ogbn-arxiv` are from the corresponding source. Then we conduct experiments on these datasets under the extremely weak supervision setting. In particular, we choose three different settings: 5/10/20 labels per class. In other words, each meta-training class only consists of 5/10/20 labeled nodes (50/100/200 for `ogbn-arxiv` due to the large size of the graph), where the total labeled nodes are approximately 1%/2%/4% of nodes on the graph. It is noteworthy that we randomly select these labeled nodes from the training classes in the original datasets. The detailed statistics of these datasets are summarized in Table 2, where the class split setting denotes the number of classes used for training/validation/test.

5.2 Experimental Settings

To achieve a comparison of X-FNC with competitive baselines, we conduct experiments with the state-of-the-art few-shot node classification methods. **Prototypical Networks (PN)** [27] and **MAML** [9] are conventional few-shot methods, and we apply them on graph data. **G-Meta** [14], **GPN** [6], and **RALE** [19] are recently proposed studies on few-shot node classification.

During meta-training, we randomly sample $\mathcal{T}_{train}$ meta-training tasks from meta-training classes for model optimization. Here the support set and the query set in each meta-task will only be sampled from labeled nodes (i.e., 5/10/20 labeled nodes in each class). Then

Table 1: The overall few-shot node classification results (accuracy in %) of X-FNC and baselines under different settings.

Dataset	DBLP						Amazon-E					
Setting	5-way 3-shot			10-way 3-shot			5-way 3-shot			10-way 3-shot		
# Labels per Class	5	10	20	5	10	20	5	10	20	5	10	20
PN	49.4±3.2	51.9±3.1	53.3±3.9	36.3±3.8	38.5±2.8	40.2±3.9	51.6±2.3	52.2±2.3	53.8±2.3	36.7±3.0	38.2±2.0	41.3±3.9
MAML	50.9±3.1	51.8±1.8	56.1±2.1	39.4±2.3	44.3±2.0	45.4±3.1	48.8±2.4	49.4±3.3	53.9±2.7	39.0±3.2	40.3±3.2	41.5±3.2
G-Meta	59.8±3.3	61.8±3.5	63.3±4.1	44.9±2.9	51.0±3.4	52.9±3.6	53.4±2.2	55.7±3.6	56.6±3.2	39.6±4.1	41.9±3.0	45.6±4.3
GPN	58.6±3.8	62.5±2.8	66.9±4.3	50.6±3.9	52.7±2.4	54.6±3.4	56.0±4.1	60.7±4.7	63.0±2.3	42.1±4.8	45.8±3.3	52.1±4.8
RALE	64.7±4.1	66.9±4.7	67.9±4.0	51.3±4.2	55.0±3.2	56.9±4.0	60.4±4.5	64.0±4.8	66.1±4.5	47.8±4.4	48.6±4.8	52.4±3.3
X-FNC	**70.1**±4.0	**75.5**±3.5	**76.8**±3.3	**57.2**±3.4	**63.6**±3.3	**65.8**±3.1	**69.9**±3.9	**72.8**±3.4	**76.0**±4.8	**49.2**±4.1	**51.5**±2.8	**56.3**±3.4

Dataset	Cora-full						ogbn-arxiv					
Setting	5-way 3-shot			10-way 3-shot			5-way 3-shot			10-way 3-shot		
# Labels per Class	5	10	20	5	10	20	50	100	200	50	100	200
PN	45.5±2.7	48.1±3.6	48.9±3.8	28.2±3.8	31.6±3.3	34.4±2.5	39.1±2.5	40.8±3.7	42.6±3.1	23.1±3.6	24.4±3.1	27.7±3.5
MAML	46.9±2.6	48.6±3.0	49.2±2.7	32.7±2.5	33.2±2.3	35.8±1.9	41.0±2.4	41.9±1.9	43.1±3.4	23.2±2.1	25.4±3.1	28.0±3.2
G-Meta	57.7±3.9	58.7±3.6	59.8±2.6	41.7±3.3	42.0±3.0	43.8±2.7	43.5±3.6	44.7±2.9	46.5±4.4	27.4±4.4	29.0±2.8	29.9±2.5
GPN	54.6±2.8	55.2±3.6	57.7±4.2	38.4±2.8	40.2±2.9	42.0±4.5	46.6±3.4	47.1±3.9	48.4±2.9	26.1±2.6	30.9±3.6	33.5±3.5
RALE	58.2±2.8	59.3±4.1	63.1±3.9	38.1±4.2	43.4±2.8	44.0±4.5	49.3±3.0	51.4±3.9	52.5±4.6	30.4±2.5	31.7±3.3	33.9±4.8
X-FNC	**62.9**±4.5	**68.0**±3.7	**69.2**±4.6	**43.7**±4.8	**45.6**±4.4	**47.7**±4.5	**54.6**±2.6	**56.7**±4.0	**58.7**±4.1	**33.3**±3.9	**35.7**±4.4	**39.8**±2.4

Table 2: Statistics of four node classification datasets.

Dataset	# Nodes	# Edges	# Features	Class Split
Amazon-E	42,318	43,556	8,669	90/37/40
DBLP	40,672	288,270	7,202	80/27/30
Cora-full	19,793	65,311	8,710	25/20/25
ogbn-arxiv	169,343	1,166,243	128	15/5/20

during meta-test, we evaluate the model on a series of randomly sampled meta-test tasks from the entire node set of meta-test classes. The final averaged classification accuracy on meta-test tasks will be used as the evaluation metric. For the Poisson Label Propagation module, we set the number of label propagation steps T_l as 10. The number of randomly sampled nodes R to construct the subgraph for label propagation is set as 10. The scaling parameter of $\mathbf{A}''$ is set as 100, and the hyper-parameter λ is set as 0.5. The number of selected pseudo-labeled nodes M is 20. For IB fine-tuning, the number of fine-tuning steps T is 40. The mask rate γ is set as 0.1. The learning rate α during fine-tuning is 0.1. The meta-learning rates β_1 and β_2 during meta-optimization are set as 0.005. The trade-off hyper-parameter β is set as 1. The hidden sizes of GNN_θ and GNN_ϕ are both 64. The hidden size of MLP_θ is 64 while the hidden sizes of the two MLP layers in p_θ are 128 and 64, respectively. The number of training epochs T_{train} is 5,000, and the number of meta-test tasks T_{test} is 500. The dropout rate is 0.5. The query set size $|\boldsymbol{Q}|$ is 10. Our code can be found at https://github.com/SongW-SW/X-FNC.

5.3 Performance Comparison

Table 1 presents the performance comparison of our framework X-FNC and all baselines on few-shot node classification with extremely weak supervision. Specifically, we choose two different few-shot settings to obtain a more comprehensive comparison: 5-way 3-shot and 10-way 3-shot. We use the average classification accuracy over 10 repetitions as the evaluation metric. From Table 1, we can have the following observations: (1) Our framework X-FNC achieves the best results compared with all other baselines in all datasets. The performance also consistently outperforms other baselines under different settings, which validates the superiority of X-FNC on few-shot node classification with extremely weak supervision. (2) When the number of labels per class decreases from 20 to 5, X-FNC has the least performance drop compared with other baselines. The main reason is that X-FNC obtains pseudo-labeled nodes via Poisson Label Propagation to alleviate the under-generalizing problem with extremely weak supervision. (3) The performance improvement of X-FNC over other baselines is slightly larger on DBLP. This is due to the fact that DBLP has a larger average node degree, which helps improve the pseudo-labeling accuracy during meta-training for better performance. (4) When the value of N increases (i.e., more classes in each meta-task), all methods encounter a significant performance drop, since query nodes are classified from a larger class set in each meta-task. Nevertheless, under the extremely limited setting, X-FNC consistently outperforms other baselines. It is because X-FNC can better extract decisive information for classification with a larger value of N via IB fine-tuning.

5.4 Ablation Study

We conduct an ablation study on Amazon-E and Cora-full to evaluate the effectiveness of different components in our framework X-FNC (similar results on other datasets). Specifically, we compare X-FNC with three degenerate versions: (a) X-FNC without pseudo-labeling (X-FNC\P); (b) X-FNC without IB-based fine-tuning (X-FNC\I); (c) without both (X-FNC\PI). More specifically, X-FNC\P removes the pseudo-labeling process such that the support set only consists of the given labeled nodes. X-FNC\I replaces the IB fine-tuning process with a simple classifier during fine-tuning, while X-FNC\PI combines the two variants. From Fig. 3, we can obtain several observations. First, our framework outperforms all other variants, which further validates that each module plays an important role in few-shot node classification with extremely weak

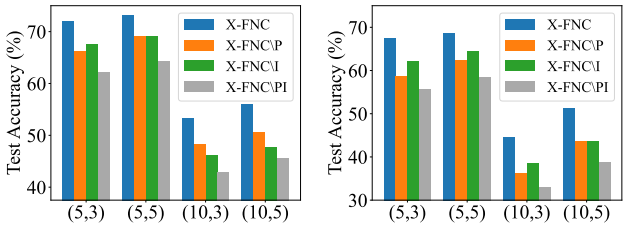

Figure 3: Ablation study on our framework on `Amazon-E` (left) and `Cora-full` (right) in the N-way K-shot setting (N, K).

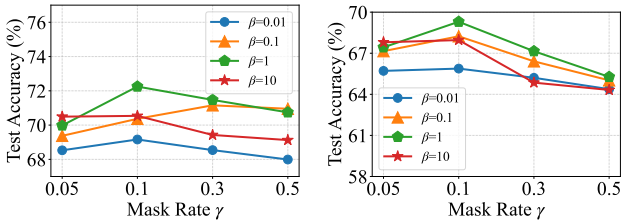

Figure 4: Results of our framework on `Amazon-E` (left) and `Cora-full` (right) with different mask rates.

supervision. Second, removing the Poisson Learning module deteriorates the performance on `Cora-full` more than that on `Amazon-E`. The reason is that `Cora-full` consists of significantly fewer meta-training classes than `Amazon-E`, and obtaining pseudo-labeled nodes becomes more crucial in this scenario. Third, without IB fine-tuning, the performance drops more significantly on 10-way settings than 5-way settings. The result further indicates that IB fine-tuning is critical for model generalization to meta-test classes, especially when each meta-task includes more classes.

5.5 Effect of Loss $\mathcal{L}_D$

In this part, we conduct experiments to study the effect of the loss $\mathcal{L}_D$ in IB fine-tuning, which is calculated according to Eq. (14) with two hyper-parameters (i.e., the loss weight β and the mask rate γ). Specifically, in X-FNC, β represents the level of attention the model pays to the irrelevant local structures for classification on a specific node. According to the IB principle, with a higher loss weight β, the model will focus more on filtering out irrelevant information for classification while less on extracting decisive classification information. On the other hand, the mask rate γ represents the approximate ratio of irrelevant information in the local structure of each node, which should be adjusted according to different datasets. To demonstrate the joint impact of these two hyper-parameters, we present the results with different values of β and γ on `Amazon-E` and `Cora-full`. From Fig. 4, we can observe that the mask rate of 0.1 generally provides better performance than other values. This is mainly because a small mask rate can be insufficient to filter out irrelevant structural information, while a larger mask rate can result in the loss of helpful information in local structures. Moreover, the performance drop on `Cora-full` is slightly larger than `Amazon-E`. The reason is that in `Cora-full`, the average node degree is significantly larger than `Amazon-E`. As a result, the graph structure encodes more decisive information for classification, which is more easily impacted by a large mask rate.

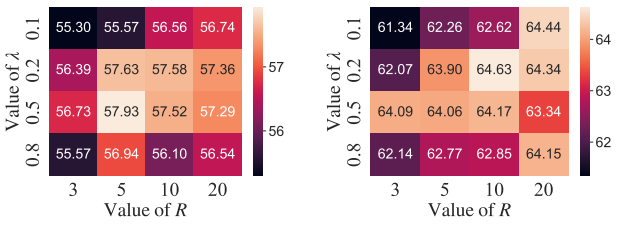

Figure 5: Results of pseudo-labeling accuracy (in %) on `Amazon-E` (left) and `Cora-full` (right) with different λ and R.

5.6 Random Sampling in Pseudo-labeling

In this part, we study the impact of factors that affect the pseudo-labeling accuracy during label propagation. Specifically, the sample number R controls the ratio of random unlabeled nodes in the constructed subgraph during pseudo-labeling. On the other hand, the distance-based adjacency matrix $\mathbf{A}''$ acts as flexible connections between randomly sampled nodes and the limited labeled nodes (i.e., support nodes). Hence, we also adjust the values of R and the scaling hyper-parameter λ to evaluate their influence. From Fig. 5, we observe that the two parameters affect the pseudo-labeling accuracy differently. In particular, increasing the number of randomly sampled nodes R will first increase pseudo-labeling accuracy and then keep it stable. It is because a more complex structure of the constructed subgraph can help the label propagation process. In addition, a higher λ (i.e., the scaling hyper-parameter for $\mathbf{A}''$) first increases the pseudo-labeling accuracy while later deteriorating the accuracy. The reason is that with a higher value of λ, the model will focus more on label propagation to random nodes instead of neighbors of support nodes. As a result, a higher λ can help discover more nodes that share the same classes with support nodes.

6 CONCLUSION

In this paper, we study the problem of few-shot node classification with extremely weak supervision, which focuses on predicting labels for nodes in meta-test classes while utilizing extremely limited labeled nodes for meta-training. Furthermore, to tackle the challenges caused by extremely limited labeled nodes, we propose an innovative framework X-FNC to obtain pseudo-labeled nodes via Poisson Learning and conduct fine-tuning based on the IB principle. As a result, our framework can expand the support set in each meta-task to alleviate the problem of under-generalizing while filtering out irrelevant information for classification to avoid over-fitting. We conduct extensive experiments on four node classification datasets with extremely weak supervision, and the results validate the superiority of our framework over other state-of-the-art baselines.

ACKNOWLEDGEMENTS

This work is supported by the National Science Foundation under grants IIS-2006844, IIS-2144209, IIS-2223769, CNS-2154962, and BCS-2228534, the JP Morgan Chase Faculty Research Award, the Cisco Faculty Research Award, the Jefferson Lab subcontracts JSA-22-D0311 and JSA-23-D0163, the Commonwealth Cyber Initiative awards HV-2Q23-003 and VV-1Q23-007, the 4-VA Collaborative Research grant, and the UVA 3Cavaliers Seed Research Grant.

REFERENCES

[1] Alexander A Alemi, Ian Fischer, Joshua V Dillon, and Kevin Murphy. 2017. Deep variational information bottleneck. In *ICLR*.

[2] Aleksandar Bojchevski and Stephan Günnemann. 2018. Deep Gaussian Embedding of Graphs: Unsupervised Inductive Learning via Ranking. In *ICLR*.

[3] Jeff Calder, Brendan Cook, Matthew Thorpe, and Dejan Slepcev. 2020. Poisson Learning: Graph Based semi-supervised learning at very low label rates. In *ICML*.

[4] Kaize Ding, Jundong Li, Nitin Agarwal, and Huan Liu. 2020. Inductive anomaly detection on attributed networks. In *IJCAI*.

[5] Kaize Ding, Jianling Wang, Jundong Li, James Caverlee, and Huan Liu. 2021. Weakly-supervised Graph Meta-learning for Few-shot Node Classification. *arXiv:2106.06873* (2021).

[6] 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*.

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

[8] Kaize Ding, Chuxu Zhang, Jie Tang, Nitesh Chawla, and Huan Liu. 2022. Toward Graph Minimally-Supervised Learning. In *SIGKDD*.

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

[10] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Pires, Zhaohan Guo, Mohammad Azar, et al. 2020. Bootstrap Your Own Latent: A new approach to self-supervised learning. In *NeurIPS*.

[11] Zhichun Guo, Chuxu Zhang, Wenhao Yu, John Herr, Olaf Wiest, Meng Jiang, and Nitesh V Chawla. 2021. Few-Shot Graph Learning for Molecular Property Prediction. In *The Web Conference*.

[12] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. 2020. Momentum contrast for unsupervised visual representation learning. In *CVPR*.

[13] 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 *NeurIPS*.

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

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

[16] Xinzhe Li, Qianru Sun, Yaoyao Liu, Qin Zhou, Shibao Zheng, Tat-Seng Chua, and Bernt Schiele. 2019. Learning to self-train for semi-supervised few-shot classification. In *NeurIPS*.

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

[18] Yanbin Liu, Juho Lee, Minseop Park, Saehoon Kim, and Yi Yang. 2018. Transductive propagation network for few-shot learning.

[19] 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*.

[20] Zheng Liu, Xiaohan Li, Hao Peng, Lifang He, and S Yu Philip. 2020. Heterogeneous similarity graph neural network on electronic health records. In *IEEE Big Data*.

[21] Julian McAuley, Rahul Pandey, and Jure Leskovec. 2015. Inferring networks of substitutable and complementary products. In *SIGKDD*.

[22] Nikhil Mishra, Mostafa Rohaninejad, Xi Chen, and Pieter Abbeel. 2018. A Simple Neural Attentive Meta-Learner. In *ICLR*.

[23] Alex Nichol, Joshua Achiam, and John Schulman. 2018. On first-order meta-learning algorithms. In *arXiv:1803.02999*.

[24] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. In *arXiv:1807.03748*.

[25] Sachin Ravi and Hugo Larochelle. 2016. Optimization as a model for few-shot learning. In *International Conference on Learning Representations*.

[26] 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*.

[27] Jake Snell, Kevin Swersky, and Richard Zemel. 2017. Prototypical networks for few-shot learning. In *NeurIPS*.

[28] 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*.

[29] Zhen Tan, Kaize Ding, Ruocheng Guo, and Huan Liu. 2022. Graph few-shot class-incremental learning. In *WSDM*.

[30] Zhen Tan, Song Wang, Kaize Ding, Jundong Li, and Huan Liu. 2022. Transductive Linear Probing: A Novel Framework for Few-Shot Node Classification. In *LoG*.

[31] Jie Tang, Jing Zhang, Limin Yao, Juanzi Li, Li Zhang, and Zhong Su. 2008. Arnetminer: extraction and mining of academic social networks. In *SIGKDD*.

[32] Shantanu Thakoor, Corentin Tallec, Mohammad Gheshlaghi Azar, Remi Munos, Petar Veličković, and Michal Valko. 2021. Bootstrapped Representation Learning on Graphs. In *ICLR Workshop on Geometrical and Topological Representation Learning*.

[33] Naftali Tishby and Noga Zaslavsky. 2015. Deep learning and the information bottleneck principle. In *2015 IEEE ITW*.

[34] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2018. Graph attention networks. *ICLR*.

[35] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. 2016. Matching networks for one shot learning. In *NeurIPS*.

[36] Song Wang, Chen Chen, and Jundong Li. 2022. Graph Few-shot Learning with Task-specific Structures. In *NeurIPS*.

[37] Song Wang, Kaize Ding, Chuxu Zhang, Chen Chen, and Jundong Li. 2022. Task-Adaptive Few-shot Node Classification. In *SIGKDD*.

[38] Song Wang, Yushun Dong, Xiao Huang, Chen Chen, and Jundong Li. 2022. FAITH: Few-Shot Graph Classification with Hierarchical Task Graphs. In *IJCAI*.

[39] Song Wang, Xiao Huang, Chen Chen, Liang Wu, and Jundong Li. 2021. REFORM: Error-Aware Few-Shot Knowledge Graph Completion. In *CIKM*.

[40] Tailin Wu, Hongyu Ren, Pan Li, and Jure Leskovec. 2020. Graph Information Bottleneck. In *NeurIPS*.

[41] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. 2020. A comprehensive survey on graph neural networks. In *IEEE TNNLS*.

[42] Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny. 2021. Barlow twins: Self-supervised learning via redundancy reduction. In *ICML*.

[43] 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*.