Generalized Few-Shot Node Classification

Zhe Xu*, Kaize Ding†, Yu-Xiong Wang*, Huan Liu†, Hanghang Tong*
*University of Illinois Urbana-Champaign, {zhexu3, yxw, htong}@illinois.edu
†Arizona State University, {kding9, huanliu}@asu.edu

***Abstract*—For real-world graph data, the node class distribution is inherently imbalanced and long-tailed, which naturally leads to a few-shot learning scenario with limited nodes labeled for newly emerging classes. Existing efforts are carefully designed to solve such a few-shot learning problem via data augmentation, learning transferable initialization, to name a few. However, most, if not all, of them are based on a strong assumption that all the test nodes must exclusively come from novel classes, which is impractical in real-world applications. In this paper, we study a broader and more realistic problem named *generalized few-shot node classification*, where the test samples can be from both novel classes and base classes. Compared with the standard few-shot node classification, this new problem imposes several unique challenges, including *asymmetric classification* and *inconsistent preference*. To counter those challenges, we propose a shot-aware graph neural network (STAGER) equipped with an uncertainty-based weight assigner module for adaptive propagation. To formulate this problem from the meta-learning perspective, we propose a new training paradigm named imbalanced episodic training to ensure the label distribution is consistent between the training and test scenarios. Experiment results on four real-world datasets demonstrate the efficacy of our model, with up to 14% accuracy improvement over baselines.**

***Index Terms*—graph mining, node classification, meta-learning**

I. Introduction

The task of node classification aims to classify nodes into categories, which has been extensively studied [1], [3], [8], [23], [24], [36], [39]. It often requires sufficient labelled nodes from *all* the classes. However, due to the ever-growing new data and high annotation cost, the number of labeled nodes from various classes tends to follow a long-tailed distribution [10]. Consequentially, some classes might not have sufficient labeled nodes, which in turn degrades the performance of node classifiers dramatically. This problem with limited labelled nodes per class (i.e., shots) is known as few-shot node classification [10], [21], [26], [47].

Formally, few-shot node classification separates the classes of interest into the base (e.g., the classic research areas) and novel classes (e.g., the emerging research areas), where the former are provided with many shots and the latter are provided with only few shots (usually less than 10). At the test phase, the classifier aims to accurately classify test nodes *only* into the novel classes. Directly training advanced node classifiers (e.g., graph neural networks (GNNs)) on few labeled nodes from novel classes is prone to overfitting. Recently, increasing research efforts have been made to address the few-shot node classification problem. Representative works include MetaGNN [47], graph prototypical networks (GPN) [10], G-META [21], and so on. Most of the existing solutions are under the umbrella of *meta-learning* [38]. Concretely, they first construct episodes [10], [32], [40], [41], [47] from the base classes, every of which includes the training and evaluation of a (base-) learner (e.g., a node classifier). Then a meta-learner is updated by supervising the training and evaluation process of the learner, and 'learns to X' (e.g., learns to train a node classifier) through the iterative episodes. With such high-level knowledge extracted by the episodic training, fitting a classifier on few-shot novel classes under the supervision of the meta-learner often obtains strong performance.

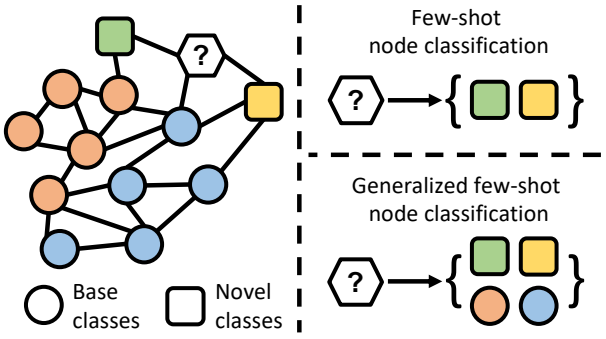

Fig. 1: An illustrative example to show the difference between the few-shot and generalized few-shot node classification tasks. For the generalized one, the test node is expected to be classified into the joint set of the novel and base classes.

Despite the great progress, most, if not all, of them follow a strong assumption that all the test nodes must be exclusively sampled from the novel classes, which is hardly realistic in real-world applications. A motivating example is a bibliography literature classification system. More often than not, a newly-published paper could fall into a classic domain (i.e., a base class) instead of a new domain (i.e., a novel class). To bridge the gap between the setting of the few-shot node classification problem and real-world scenarios, in this paper, we study a broader and more practical problem named *generalized few-shot node classification*. Given base classes with many shots and novel classes with few shots, in this new problem, a node classifier is expected to perform classification on the *joint label set* of both base and novel classes, instead of the label set of the novel classes alone. An illustrative example is provided in Figure 1. This subtle difference in the target label set leads to a significantly more challenging problem from the following two perspectives.

Challenges. First (*asymmetric classification*), at the meta-test phase for meta-learning based methods (or the test phase for

Dataset	Method	$b \to b$	$b \to n$	$n \to b$	$n \to n$
Amazon Clothing	APPNP	100.0	0.0	69.6	30.4
	MetaGNN	100.0	0.0	61.2	38.8
Cora-Full	APPNP	99.2	0.8	66.8	33.2
	MetaGNN	99.0	1.0	72.4	27.6

TABLE I: Test nodes (%) classified from base classes to base classes (i.e., $b \to b$), from base classes to novel classes (i.e., $b \to n$), from novel classes to base classes (i.e., $n \to b$), and from novel classes to novel classes (i.e., $n \to n$), respectively.

standard learning methods), a classifier tends to show more confidence towards the base classes compared with the novel classes [44] due to the imbalanced shots. In Table I, we illustrate that by presenting classification results on Amazon-Clothing [28] and Cora-Full [4] datasets from a regular node classifier (APPNP [24]) and a few-shot node classifier (MetaGNN [47]). We observe that the majority of nodes from the novel classes are misclassified into the base classes. Similar results have been reported in the generalized zero-shot learning problem [6], [14], [45] from the computer vision domain.

Second (*inconsistent preference*), appropriately aggregating information from multiple receptive fields plays an essential role in the effectiveness of graph neural networks [8], [11], [24]. However, the optimal weight assignments among receptive fields could vary dramatically between the many-shot cases (i.e., from a base class) and the few-shot cases (i.e., from a novel class). For instance, on a sparsely-connected homophilic graph (where edges often connect same-class nodes), under the many-shot settings, classifiers prefer to pay more attention to the local information [8], [24] from small receptive fields; on the contrary, under the few-shot settings, more attention to the long-range propagation is necessary as the labeled nodes will be very sparse in the given graph [9], [24], [26], [43]. Clearly, there is tension between the above two scenarios about the weight assignments of receptive fields. Nonetheless, it is unknown how to kill two birds with one stone by designing an adaptive model for both base and novel classes. It is worth pointing out that the inconsistent preference challenge only pertains to the generalized few-shot node classification problem. In contrast, for the standard few-shot node classification problem, a classifier will not face the *inconsistent preference* challenge, and a consistent weight assignment is often sufficient. This is because, at the meta-test time, classifiers only need to make a prediction among novel classes whose numbers of shots are more or less balanced.

Our Contributions. First (*C1. New Models*), we propose a novel model named shot-aware graph neural network (STAGER). The key idea is to decompose the prediction probability into two parts, and instantiate them by two parametric models: a meta-learner and a learner, respectively. Our analysis reveals that such a pair of models derived from the decomposed probability is essential to tackle the *asymmetric classification* challenge. In particular, we find that the prediction uncertainty can accurately reflect the shots of the class to which the test node belongs. Based on that, the meta-learner is instantiated as a weight assigner whose input is the prediction uncertainty of given nodes and output is the assigned weights for multiple receptive fields to address the *inconsistent preference* challenge. Second (*C2. New Training Paradigm*), to fit the training of our models into the meta-learning framework, we propose a new training paradigm called *imbalanced episodic training* which alleviates the discrepancy between the meta-training and meta-test scenarios by mimicking the imbalanced shots settings. The key idea of imbalanced episodic training is to split the base classes into *pseudo-novel* classes and *pseudo-base* classes and downsample the labeled nodes from the pseudo-novel classes. Third (*C3. Extensive Evaluations*), comprehensive experiments on four real-world datasets demonstrate that the proposed model STAGER and the imbalanced episodic training paradigm significantly improve the prediction accuracy of novel classes under various settings and sometimes even improve the prediction accuracy of base classes as well.

II. PROBLEM DEFINITION

We use bold uppercase letters for matrices (e.g., $\mathbf{A}$), bold lowercase letters for vectors (e.g., $\mathbf{u}$), lowercase and uppercase letters in regular font for scalars (e.g., d, K), and calligraphic letters for sets (e.g., $\mathcal{T}$). $\mathbf{A}[i,j]$ denotes the entry of matrix $\mathbf{A}$ at the i-th row and the j-th column, $\mathbf{A}[i,:]$ denotes the i-th row of matrix $\mathbf{A}$, and $\mathbf{A}[:,j]$ denotes the j-th column of matrix $\mathbf{A}$. Similarly, $\mathbf{u}[i]$ denotes the i-th entry of vector $\mathbf{u}$. Superscript T denotes the transpose of matrices and vectors (e.g., $\mathbf{A}^T$ is the transpose of $\mathbf{A}$). An attributed graph can be represented as $\mathcal{G} = \{\mathbf{A}, \mathbf{X}\}$ which is composed by an adjacency matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$ and an attribute matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$, where n is the number of nodes and d is the node feature dimension. In total, nodes can be categorized into a set of classes C. The node set $\mathcal{V}$ and the class set $\mathcal{C}$ will be split and notated with appropriate subscripts. For example, $\mathcal{V}_{\texttt{test}}$ and $\mathcal{C}_{\texttt{novel}}$ refer to the test nodes and the novel classes, respectively. N denotes the number of novel classes and K denotes the number of training nodes per novel class. Note that the number of base classes and the number of training nodes per base class are much larger than N and K, and they are not fixed across different datasets. As mentioned in Section I, we do not exclude the base classes from the class membership of test nodes and study the *generalized few-shot node classification* problem which is defined as follows.

Problem 1: Generalized few-shot node classification

Given: (1) a graph $\mathcal{G} = \{\mathbf{A}, \mathbf{X}\}$, (2) labelled nodes $\mathcal{V}_{\texttt{base}}$ from base classes $\mathcal{C}_{\texttt{base}}$, (3) labelled nodes $\mathcal{V}_{\texttt{novel}}$ from novel classes $\mathcal{C}_{\texttt{novel}}$ ($\mathcal{C}_{\texttt{base}} \cap \mathcal{C}_{\texttt{novel}} = \emptyset$) where each novel class has very few (e.g., 1) labelled nodes, (4) unlabelled test nodes $\mathcal{V}_{\texttt{test}}$ from classes $\mathcal{C}_{\texttt{novel}} \cup \mathcal{C}_{\texttt{base}}$, where $\mathcal{V}_{\texttt{base}} \cap \mathcal{V}_{\texttt{test}} = \emptyset$ and $\mathcal{V}_{\texttt{novel}} \cap \mathcal{V}_{\texttt{test}} = \emptyset$.

Find: The predicted labels for the unlabelled test nodes $\mathcal{V}_{\texttt{test}}$.

We remark that the topology and attribute information of all the nodes are given and we study this problem under the semi-supervised (and transductive) setting. Based on the naming convention from the few-shot learning community, we name it as the *generalized N-way K-shot* node classification problem.

Note that this naming convention *only* describes the setting of novel classes. The number of base classes is at least $3N$, and each of the base classes is provided with at least $10K$ shots.

A. Preliminaries: Episodic Training for Meta-Learning

Our proposed method is inspired by the meta-learning-based few-shot learning solutions. Here, we introduce the classic episodic training [10], [12], [32], [40], [41], [47].

Meta-learning is also known as *learning-to-learn* which describes the interaction between a meta-learner (parameterized by ϕ) and a (base-) learner (e.g., a classifier parameterized by θ). According to the conventional N-way K-shot problem setting [10], [41], [47], at the *meta-test* phase, all the nodes exclusively come from the novel classes $\mathcal{C}_{\texttt{novel}}$ and $|\mathcal{C}_{\texttt{novel}}| = N$. The classifier θ will fit on the provided $N \times K$ labeled nodes (i.e., K labeled nodes per novel class) with the assistance of the meta-learner ϕ. The meta-test performance is measured by the fitted classier on the test nodes.

To align the meta-training and meta-test scenarios, episodic training [10], [32], [40], [41], [47] mimics the meta-test scenario and generates episodes $\{\mathcal{E}_i = \{\mathcal{S}_i, \mathcal{Q}_i\}\}$ from base classes. In every episode, N base classes are randomly selected. Then, for the selected base classes, K and I labeled nodes per base class are sampled to compose the support set $\mathcal{S}_i$ and the query set $\mathcal{Q}_i$ respectively. Here the configuration of the support set is to align with the N-way K-shot meta-test scenarios, and I is fixed (e.g., 30) as many existing works [10], [32] did. As the support set $\mathcal{S}_i$ and query set $\mathcal{Q}_i$ are both labeled but do not overlap with each other, we use v and v' with indices to distinguish nodes from the support set $\mathcal{S}_i$ with those from the query set $\mathcal{Q}_i$. The i-th episode can be represented as Eq. (1) and the training objective can be formulated as Eq. (2).

$$\mathcal{S}_i = \{v_1, \ldots, v_{N\times K}\},\ \mathcal{Q}_i = \{v'_1, \ldots, v'_{N\times I}\}. \tag{1}$$

$$\begin{aligned} \phi^* &= \arg\min_{\phi}\ \mathbb{E}_{v_i\in\mathcal{Q}}\ L_{\texttt{cla}}(z(\mathcal{G}, \theta^*, \phi, v_i), y_i), \\ s.t.\ \theta^* &= \arg\min_{\theta} \mathbb{E}_{v_j\in\mathcal{S}}\ L_{\texttt{cla}}(z(\mathcal{G}, \theta, \phi, v_j), y_j), \end{aligned} \tag{2}$$

where $z(\mathcal{G}, \theta, \phi, v_i)$ is the classification results on node v_i by the classifier θ (with the assistance from the meta-learner ϕ), y_i is the label of v_i, and $L_{\texttt{cla}}()$ is the classification loss. The classifier θ is trained from scratch with the meta-learner ϕ on the support set $\mathcal{S}_i$ (i.e., the lower-level objective) and its loss on the query set $\mathcal{Q}_i$ serves as the supervision to update the meta-learner ϕ (i.e., the upper-level objective).

III. Proposed Method

In this section, we first introduce the overall motivation of our model design. Then we present the concrete instantiation of every model component. After that, a tailored imbalanced episodic training for our model is proposed.

A. Design Motivation

From the statistical learning perspective, the goal of the generalized few-shot node classification is to infer the probability $P(y_i|\mathcal{G}, v_i)$, where y_i is the label of the node v_i:

$$P(y_i|\mathcal{G}, v_i) = \sum_{\tilde{C}\in\{novel, base\}} P(y_i|\tilde{C}, \mathcal{G}, v_i)P(\tilde{C}|\mathcal{G}, v_i), \tag{3}$$

where $\tilde{C}$ is a variable to indicate whether the given node v_i belongs to novel classes or base classes. When $\tilde{C} = base$, inferring $P(y_i|\tilde{C}, \mathcal{G}, v_i)$ is equivalent to the standard node classification problem [1], [3], [8], [23], [24], [36], [39] with many shots. On the contrary, if $\tilde{C} = novel$, inferring $P(y_i|\tilde{C}, \mathcal{G}, v_i)$ is equivalent to the few-shot node classification problem [10], [41], [47]. In a nutshell, there exist rich approaches to estimate $P(y_i|\tilde{C}, \mathcal{G}, v_i)$ in both cases.

However, resolving the problem in one model as Eq. (3) is a great challenge. The empirical evidence in Table I illustrates that the effect of *asymmetric classification* deteriorates the classification performance significantly. In other words, when applied to the generalized few-shot node classification problem, existing classic and few-shot node classifiers tend to over-estimate $P(\tilde{C} = base|\mathcal{G}, v_i)$ yet under-estimate $P(\tilde{C} = novel|\mathcal{G}, v_i)$, i.e., $P(\tilde{C} = base|\mathcal{G}, v_i) \gg P(\tilde{C} = novel|\mathcal{G}, v_i)$ even if v_i is from the novel classes. An intuitive explanation is that, for existing methods, most of them directly estimate $P(y_i|\mathcal{G}, v_i)$ by one parametric model whose training is overwhelmed by the labeled nodes from the base classes. Thus, the implicit component $P(\tilde{C}|\mathcal{G}, v_i)$ of the estimated probability $P(y_i|\mathcal{G}, v_i)$ is heavily biased. Based on the above analysis, our overall solution for the *asymmetric classification* is estimating $P(y_i|\tilde{C}, \mathcal{G}, v_i)$ and $P(\tilde{C}|\mathcal{G}, v_i)$ separately.

Unfortunately, exactly inferring of $P(y_i|\tilde{C}, \mathcal{G}, v_i)$ or $P(\tilde{C}|\mathcal{G}, v_i)$ is not feasible. This is because, on graph data, the labels of a node v_i (both $\tilde{C}$ and y_i) are determined by its attributed ego net and exact inferring them requires enumerating all the possible attributed ego nets. Therefore, we approximate the above two distributions by tractable models, including a classifier $f(\theta)$ and a weight assigner $g(\phi)$. In the following subsections, we will introduce the instantiations of $f(\theta)$ and $g(\phi)$; after that, a novel generalized episodic training paradigm is presented, which is tailored for the generalized few-shot node classification problem and can work hand-in-hand with our models.

B. Models

The overall framework of our proposed model STAGER is presented in Figure 2. It is composed by a classifier $f(\theta)$ and a weight assigner $g(\phi)$ which are introduced below.

A - Classifier $f(\theta)$. The classifier is instantiated as follows which is based on the *predict-then-propagate* design [24],

$$\mathbf{H}^{(0)} = \texttt{MLP}(\mathbf{X}, \theta), \tag{4a}$$

$$\mathbf{H}^{(j+1)} = \tilde{\mathbf{A}}\mathbf{H}^{(j)}, \tag{4b}$$

$$\mathbf{Z} = \texttt{softmax}\Big(\sum_{j=0}^{p}(\mathbf{W}[:, j]\mathbf{1}^T) \odot \mathbf{H}^{(j)}\Big). \tag{4c}$$

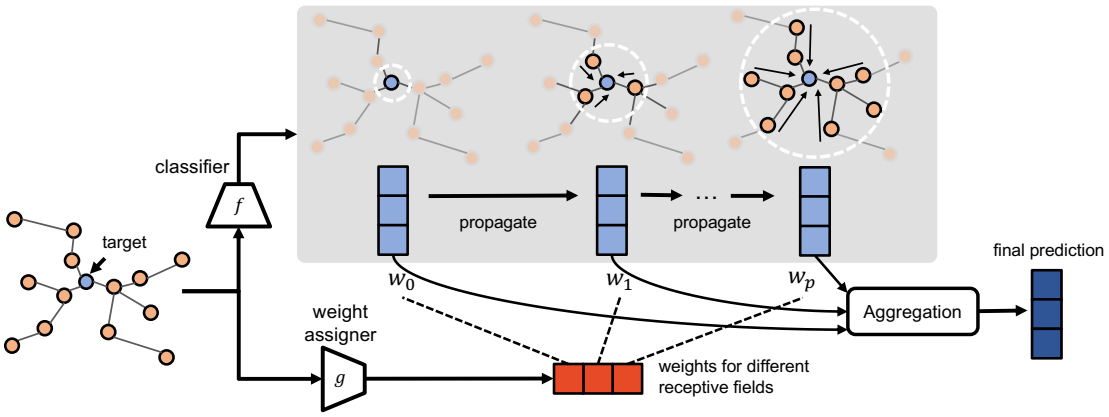

Fig. 2: The framework of the proposed STAGER. For clarity, we set the propagation step $p = 2$ in this figure.

We first obtain the prediction $\mathbf{H}^{(0)} \in \mathbb{R}^{n\times C}$ based on the node attributes $\mathbf{X}$ from a multi-layer perceptron (MLP) parameterized by θ (i.e., Eq. (4a)), where n is the number of nodes and C is the total number of node classes (including both $\mathcal{C}_{\texttt{base}}$ and $\mathcal{C}_{\texttt{novel}}$). Then, the prediction matrix $\mathbf{H}^{(0)}$ is propagated p steps to obtain a group of prediction matrices $\{\mathbf{H}^{(0)}, \ldots, \mathbf{H}^{(p)}\}$ by power iterations with $\tilde{\mathbf{A}}$ (i.e., Eq. (4b)). Here $\tilde{\mathbf{A}} \in \mathbb{R}^{n\times n}$ is the symmetrically normalized adjacency matrix with self-loops. Notice that these prediction matrices also denote the predictions based on different receptive fields. Finally, all these prediction matrices are aggregated by a weight assignment matrix $\mathbf{W} \in \mathbb{R}^{n\times(p+1)}$ whose entry $\mathbf{W}[i,j]$ represents the importance of the j-th propagated prediction matrix (i.e., $\mathbf{H}^{(j)}$) for the i-th node (i.e., $\mathbf{Z}[i,:]$). By broadcasting the weight vector for the j-th propagation (i.e., $\mathbf{W}[:,j]$) with an all-one vector $\mathbf{1} \in \mathbb{R}^{C\times 1}$, we assign the weight to the j-th propagated prediction matrix through the Hadamard product $\odot$. The $\texttt{softmax}$ is row-wise.

Existing models APPNP [24] and GPRGNN [8] set every column of weight matrix $\mathbf{W}$ as a constant vector, which might suffer from the *inconsistent preference* problem as the optimal weight assignments among receptive fields could vary dramatically between the many-shot cases and the few-shot cases (see the detailed analysis in Section I). One possible solution is to set $\mathbf{W}$ as a free learnable parameter of the classifier, which is prone to overfitting as the number of parameters is linear w.r.t. the number of nodes. More importantly, as we have analyzed in Section III-A, explicitly estimating $P(\tilde{C}|\mathcal{G}, v_i)$ is necessary. Hence, we propose to encode $P(\tilde{C}|\mathcal{G}, v_i)$ into $\mathbf{W}$ as the output of a well-designed *weight assigner* as introduced below.

B - Weight assigner $g(\phi)$. Based on the design motivation laid out in Section III-A, on estimating $P(\tilde{C}|\mathcal{G}, v_i)$ and working closely with the classifier $f(\theta)$, our design of the weight assigner $g(\phi)$ bears the following rationales. First, essentially, for generalized few-shot node classification problem, $P(\tilde{C}|\mathcal{G}, v_i)$ reflects the number of shots; e.g., $P(\tilde{C} = base|\mathcal{G}, v_i)$ indicates the probability of the class of v_i being provided with many shots. Second, a sub-module should explicitly extract the shot-aware representation from the input $\{\mathcal{G}, v_i\}$. Finally, if the input of a module is shot-aware, its output should also be shot-aware. Concrete instantiation of g is as follows whose two sub-modules are parameterized by $\phi = \{\phi_1, \phi_2\}$,

$$\tilde{\mathbf{Z}} = \texttt{rank}\Big(\texttt{softmax}\big(g_1(\mathbf{A}, \mathbf{X}, \phi_1)\big)\Big), \tag{5a}$$

$$\mathbf{W} = \texttt{MLP}\big(\tilde{\mathbf{Z}}, \phi_2\big). \tag{5b}$$

The first sub-module (Eq. (5a), motivated by the first and second rationales) is a preliminary node classifier g_1 (parameterized by ϕ_1, followed with $\texttt{softmax}$) whose *ranked* output $\tilde{\mathbf{Z}} \in \mathbb{R}^{n\times C}$ is the shot-aware representation. The second sub-module is an MLP (Eq. (5b), parameterized by ϕ_2, following the last rationale), whose output is the shot-aware weight assignment matrix. The $\texttt{rank}$ is row-wise.

Our key idea is to utilize the *epistemic uncertainty*.[1] Since the epistemic uncertainty can reflect the size of training data, i.e., shots for our problem, our first sub-module of the weight assigner (Eq. (5a)) conducts a preliminary prediction (i.e., $\texttt{softmax}(g_1(\cdot))$, and then measures the epistemic uncertainty (i.e., $\texttt{MLP}(\texttt{rank}(\cdot))$, where $\texttt{MLP}$ is merged into another $\texttt{MLP}$ from Eq. (5b) for brevity) to output the shot-aware representation $\tilde{\mathbf{Z}}$. We further elaborate on a few more points. First, for better extracting the epistemic uncertainty, we adopt a practical approach named dropout variational inference [18], [22] and rewrite the input of $\texttt{rank}(\cdot)$ from $\texttt{softmax}\big(g_1(\mathbf{A}, \mathbf{X}, \phi_1)\big)$ to $\frac{1}{T}\sum_{t=1}^{T}\texttt{softmax}\big(g_1(\mathbf{A}, \mathbf{X}, \phi_1^t)\big)$ where ϕ_1^t is the masked parameter of ϕ_1 through dropout layers [33]. Second, the design of the $\texttt{rank}$ function is based on the intuition that the uncertainty is closely related to the ordered prediction vector. For example, commonly-used uncertainty metrics of a prediction vector $\mathbf{u}$, $\texttt{max}(\mathbf{u})$ (the largest probability) and $\texttt{gap}(\mathbf{u})$ (the largest probability minus the second largest one) can be represented by $[1, 0, \ldots, 0] \cdot \texttt{rank}(\mathbf{u})$ and $[1, -1, \ldots, 0] \cdot \texttt{rank}(\mathbf{u})$, respectively. Finally, the proposed uncertainty measure: $\texttt{MLP}(\texttt{rank}(\cdot))$ is flexible, thanks to the *universal approximator* $\texttt{MLP}$ [20]. We will provide an interesting discussion about the selection of g_1 in Section IV-C.

[1]Epistemic uncertainty is defined to measure how well the model fits the data and is reducible as the size of training data increases [2], [22].

C. Imbalanced Episodic Training

As mentioned in Section III-A, we decompose the classification goal into two probability distributions ($P(y_i|\tilde{C}, \mathcal{G}, v_i)$ and $P(\tilde{C}|\mathcal{G}, v_i)$) and estimate them by the classifier $f(\theta)$ and weight assigner $g(\phi)$ respectively. The key idea is designing $g(\phi)$ to output a shot-aware weight assignment matrix for the $f(\theta)$. This is because, fundamentally, for the generalized few-shot node classification problem, $P(\tilde{C}|\mathcal{G}, v_i)$ reflects shots, and vice versa. Here, we further ask: *how can we train $g(\phi)$ to estimate $P(\tilde{C}|\mathcal{G}, v_i)$ in an even broader scope, beyond the scenario of base classes $\mathcal{C}_{base}$ vs. novel classes $\mathcal{C}_{novel}$?* In other words, whether there exist other base classes vs. novel classes scenarios from which the $g(\phi)$ can learn?

To answer this question, let us take a close look at a prevalent training paradigm for meta-learning problems named episodic training [10], [32], [40], [41], [47].

The core idea of episodic training is to leverage the abundant labeled base samples to generate sufficient few-shot episodes (all classes are few-shot). The few-shot episodes facilitate the learning of a meta-learner which can in turn assist the learning of learners on few-shot scenarios. However, directly grafting such a strategy on the training of $g(\phi)$ is pointless. That is because, the goal of the weight assigner ($g(\phi)$) for the generalized few-shot node classification problem is not *learning to learn a few-shot classifier* (the core idea of meta-learning based few-shot learning), but *learning to tell if a node is from novel classes or base classes*. Based on this key insight, we generalize the episodic training and propose a novel training strategy named *imbalanced episodic training*.

A - Imbalanced episodes. For the generalized few-shot node classification task, as we have mentioned before, the ideal training scenarios of the weight assigner $g(\phi)$ are composed of base classes vs. novel classes. Hence, we propose *imbalanced episodic training* to mimic such scenarios. Particularly, our first step is to sample *pseudo-base* and *pseudo-novel* classes ($\mathcal{C}_{\texttt{pseudo-novel}}$ and $\mathcal{C}_{\texttt{pseudo-base}}$) from the base classes such that $|\mathcal{C}_{\texttt{pseudo-novel}}| = N$ and $|\mathcal{C}_{\texttt{pseudo-base}}| = M$. Then, the *labelled nodes (from $\mathcal{C}_{base}$)* which belongs to $\mathcal{C}_{\texttt{pseudo-novel}}$ and $\mathcal{C}_{\texttt{pseudo-base}}$ are notated as $\mathcal{V}_{\texttt{pseudo-novel}}$ and $\mathcal{V}_{\texttt{pseudo-base}}$. After that, episodes $\{\mathcal{E}_i = \{\mathcal{S}_i, \mathcal{Q}_i\}\}$ are sampled from $\mathcal{V}_{\texttt{pseudo-novel}}$ and $\mathcal{V}_{\texttt{pseudo-base}}$ where the i-th episode can be represented as follows.

$$\begin{aligned}\mathcal{S}_i &= \{\underbrace{v_1, \ldots, v_{N\times K}}_{\text{from } \mathcal{V}_{\texttt{pseudo-novel}}}, \underbrace{v_{N\times K+1}, \ldots, v_{N\times K+M\times L}}_{\text{from } \mathcal{V}_{\texttt{pseudo-base}}}\} \\ \mathcal{Q}_i &= \{\underbrace{v'_1, \ldots, v'_{N\times I}}_{\text{from } \mathcal{V}_{\texttt{pseudo-novel}}}, \underbrace{v'_{N\times I+1}, \ldots, v'_{(N+M)\times I}}_{\text{from } \mathcal{V}_{\texttt{pseudo-base}}}\}\end{aligned} \tag{6}$$

where K nodes are sampled from $\mathcal{V}_{\texttt{pseudo-novel}}$ per class and L nodes are sampled from $\mathcal{V}_{\texttt{pseudo-base}}$ per class to form every support set $\mathcal{S}$; from both $\mathcal{V}_{\texttt{pseudo-base}}$ and $\mathcal{V}_{\texttt{pseudo-novel}}$, I nodes are sampled per class to form every query set $\mathcal{Q}$. About the selection of N, M, K, L, I, we follow three rules of thumb: (1) $N \ll M$ because it is common that $|\mathcal{C}_{\texttt{novel}}| \ll |\mathcal{C}_{\texttt{base}}|$, (2) $K \ll L$ because $\mathcal{C}_{\texttt{pseudo-base}}$ serves as the many-shot classes and $\mathcal{C}_{\texttt{pseudo-novel}}$ serves as the few-shot classes, and (3) I is fixed (e.g., 30) as many existing works [10], [32] did. Specific selections of the above values can be found in Section IV-A.

B - Training procedure for STAGER. We rewrite the final prediction of the proposed STAGER from Eq. (4c) as $z(\mathcal{G}, \theta, \phi_1, \phi_2, v_i)$ to represent the prediction results w.r.t. the i-th node. Following the same format, we rewrite the prediction from the preliminary predictor g_1 (i.e., $\texttt{softmax}\big(g_1(\mathbf{A}, \mathbf{X}, \phi_1)\big)$ from Eq. (5a)) as $\tilde{z}(\mathcal{G}, \phi_1, v_i)$. In addition, we use y_i to represent the label of the i-th node. Then the objective of imbalanced episodic training for STAGER is,

$$\begin{aligned}\phi_2^* &= \arg\min_{\phi_2} \ \mathbb{E}_{v_i\in\mathcal{Q}} \ L_{\texttt{cla}}(z(\mathcal{G}, \theta^*, \phi_1^*, \phi_2, v_i), y_i), \\ s.t.\ \theta^*, \phi_1^* &= \arg\min_{\theta,\phi_1} \ \mathbb{E}_{v_j\in\mathcal{S}} \ L_{\texttt{cla}}(z(\mathcal{G}, \theta, \phi_1, \phi_2, v_j), y_j) \\ &\qquad + \lambda L_{\texttt{cla}}(\tilde{z}(\mathcal{G}, \phi_1, v_j), y_j),\end{aligned} \tag{7}$$

where λ is a trade-off parameter and $L_{\texttt{cla}}$ denotes the classification loss.

Remarks. First, in Eq. (7), the episodes are imbalanced according to Eq. (6). Second, the preliminary classifier $g_1(\phi_1)$ should be trained from scratch to be shot-aware in every episode; hence, ϕ_1 is optimized in the lower-level objective and only ϕ_2 is updated across episodes. Finally, in implementation, for every episode, we pretrain the preliminary classifier $g_1(\phi_1)$ based on $L_{\texttt{cla}}(\tilde{z}(\mathcal{G}, \phi_1, v_j), y_j)$ (which does not contain θ or ϕ_2) to converge and keep it fixed when solving the bilevel optimization problem in Eq. (2) (i.e., remove the term $\lambda L_{\texttt{cla}}(\tilde{z}(\mathcal{G}, \phi_1, v_j), y_j)$ from the lower-level objective). Such a training strategy shows great efficacy empirically. In principle, we can use any gradient descent-based optimization. To compute the gradient of the upper-level problem, many choices such as the first-order approximation [15], [29] and iterative differentiation methods [16], [17] are available. At the meta-test phase, we train the classifier $f(\theta)$ and preliminary classifier $g_1(\phi_1)$ from scratch and fine-tune the second module of weight assigner parameterized with ϕ_2 on all the labeled nodes with sample re-weighting.

C - Discussion. If the nodes notated as 'from $\mathcal{V}_{\texttt{pseudo-base}}$' are removed in Eq. (6), the composition of both the support set $\mathcal{S}_i$ and the query set $\mathcal{Q}_i$ is the same as Eq. (1). From another perspective, the conventional episodic training mimics the uniform distribution of the number of nodes from the selected N (pseudo-novel) classes under the few-shot settings. Our imbalanced episodic training takes one step further and mimics a more complex mixed distribution from two uniform distributions (i.e., many shots for $\mathcal{C}_{\texttt{pseudo-base}}$ and few shots for $\mathcal{C}_{\texttt{pseudo-novel}}$).

Importantly, this training paradigm is independent of our model STAGER, and it is even independent of the node classification task. In fact, it is a new *general* meta-training paradigm which can be applied to most, if not all, of the generalized few-shot learning problem from the meta-learning perspective. In addition, the idea behind the proposed imbalanced episode training can be naturally generalized to other more realistic

Dataset	Nodes	Edges	Features	Labels
Amazon Clothing	24919	91680	9034	77
Amazon Electronics	42318	43556	8669	167
Aminer	40672	288270	7202	137
Cora-Full	18800	62685	8710	56

TABLE II: Statistics of Datasets

distributions such as the long-tailed power-law distribution, and we leave those interesting topics as future works.

For the model complexity, compared with other models following the *predict-then-propagate* design [24] (e.g., APPNP [24] and GPRGNN [8]), we only add an extra MLP whose number of parameters (if only 1 hidden layer) is $O(d \times d_{\mathrm{h}} \times (p+1))$ where d, d_{h}, and p are the node feature dimension, hidden feature dimension and number of propagation steps, respectively.

IV. Experiments

We design experiments to answer the following questions:

- How effective are the model STAGER and the imbalanced episodic training?
- How to select the preliminary classifier g_1 in the weight assigner module?

A. Experiment Setup

Datasets. We use two E-commerce datasets: Amazon Clothing[2] [28], Amazon Electronics[2] [28], and two citation datasets: Aminer[3] [35], and Cora-Full[4] [4]. Detailed statistics of the datasets is in Table II. All the datasets used in this paper are publicly accessible. They are all anonymized, numerized, and do not contain personally identifiable information or offensive content. For all the datasets, we only select classes whose number of nodes is larger or equal to 100, so that we can have sufficient test nodes for every class. For the **novel classes**, we follow the N-way K shot setup where $N \in \{5, 10\}$ and $K \in \{1, 3\}$. Also, we select **many base classes** (i.e. $N \ll |\mathcal{C}_{\mathtt{base}}|$) with 50 shots and select a part of base classes as the validation classes. Our code and data are accessible [5] with detailed dataset split and hyperparameter settings.

Metrics. The performance of models is evaluated by the accuracy (ACC) on test nodes. To be specific, we report the accuracy on the base classes, novel classes, and all the classes, respectively. We report the average result together with the standard deviation in 10 runs.

Baseline methods. The baseline methods which we compare our STAGER-I (with imbalanced episodic training) and STAGER (without imbalanced episodic training) with can be categorized into: (1) classic neural node classifiers including APPNP [24], and GPRGNN [8] which follow the same predict-then-propagate design as our STAGER; (2) few-shot neural node classifiers including MetaGNN [47], GPN [10], and G-META [21]; (3) an imbalanced node classifier GraphSMOTE [46] (short as G-SMOTE). Note that if we ablate the weight assigner g from the proposed STAGER, our model will degenerate into the GPRGNN [8] whose weights of receptive fields are the same for every node.

Implementation. For the APPNP and GPRGNN, we train them with two strategies: (1) pre-training models over the base classes and fine-tuning them over the novel classes or 'novel & base' classes, and (2) training models over the imbalanced labeled nodes and re-weighting nodes from the novel classes with high weights. From our experiment results (not shown due to space limitation), the performance from the second strategy is better and we report their best performance in the following subsections. For MetaGNN, GPN, and G-META, at the meta-training phase, we train them with the existing episodic training. At the meta-test phase, we fine-tune MetaGNN on the imbalanced labeled nodes. For GPN and G-META, at the meta-test phase, since their models and codes are designed for the balanced few-shot settings, we downsample the labeled nodes from base classes so that all the classes are few-shot. For the GraphSMOTE, we implement its downstream classifier as APPNP which shows strong performance. As it conducts node augmentation based on the nodes from novel classes only when the shot is 1, there is no augmentation space and we report the same results as APPNP. When shots are 3, GraphSMOTE augments novel classes first and then trains APPNP over the augmented data.

B. Main Results

Performance comparison on four datasets is presented in Table III. We have the following observations. First, existing few-shot node classifiers do not perform well on the generalized few-shot node classification problem. For instance, MetaGNN does not show advantages compared with classic methods such as APPNP; GPN and G-META can obtain a decent performance on the novel classes but cannot fully utilize the labeled nodes from the base classes during the meta-test phase, which in turn degrades its performance on the base classes. Second, compared with classic neural node classifiers (APPNP, GPRGNN), without imbalanced episodic training, the proposed STAGER already outperforms them in most cases and retains competitive in the remaining cases. Third, there is a trade-off between the performance on the base and novel classes and we observe that, in most cases, the proposed imbalanced episodic training can indeed significantly improve the performance on the novel classes while keep competitive performance on the base classes. Fourth, in all the cases, our models (STAGER and STAGER-I) obtain the *best overall performance* on all the settings consistently. Finally, GraphSMOTE only leverages the novel classes to conduct node augmentation whose advantage is restricted.

C. Case Study on Uncertainty vs. Propagation

In this case study, we design experiments to answer two research questions: (1) whether there exists a general epistemic uncertainty gap between the many-shot classes and few-shot

[2]https://nijianmo.github.io/amazon/index.html
[3]https://www.aminer.cn/data/?nav=openData
[4]https://github.com/abojchevski/graph2gauss/tree/master/data
[5]https://github.com/pricexu/STAGER

Dataset	Setting	Class	APPNP	GPRGNN	MetaGNN	GPN	G-META	G-SMOTE	STAGER	STAGER-I
Amazon Clothing	5w1s	Base	67.4±1.6	64.7±0.5	64.0±0.5	46.1±3.3	48.7±2.7	67.4±1.6	**69.3±1.1**	67.3±0.4
		Novel	31.4±0.9	31.5±5.4	28.3±0.4	36.1±5.1	39.2±2.9	31.4±0.9	32.4±2.0	**41.3±1.0**
		All	48.5±1.0	47.3±3.0	45.4±0.4	40.9±2.9	43.7±2.2	48.5±1.0	50.0±1.0	**53.7±0.5**
	5w3s	Base	70.5±0.9	69.7±1.0	66.1±1.2	62.9±1.9	63.3±1.7	69.4±0.7	**72.3±1.5**	68.4±1.0
		Novel	48.6±2.5	50.1±3.8	40.4±0.8	46.1±6.7	47.6±6.3	45.6±2.3	53.9±2.1	**66.0±2.4**
		All	59.1±1.2	59.4±2.1	52.6±1.0	54.1±3.4	55.1±3.5	57.0±1.4	62.7±1.1	**67.2±1.1**
	10w1s	Base	73.3±0.3	70.7±1.2	67.6±0.5	42.7±2.4	48.2±2.1	73.3±0.3	**76.7±1.5**	66.7±0.5
		Novel	45.2±0.6	37.7±3.1	41.5±0.5	39.7±5.7	39.9±4.9	45.2±0.6	43.1±2.7	**59.6±0.6**
		All	58.6±0.3	53.5±1.6	54.0±0.4	40.9±3.5	43.9±2.0	58.6±0.3	59.0±1.6	**63.0±0.5**
	10w3s	Base	69.2±0.6	67.5±1.1	65.6±1.2	59.5±2.7	57.4±1.9	68.1±0.7	**70.9±0.7**	69.3±0.4
		Novel	61.4±0.4	58.0±1.5	53.6±0.2	49.6±7.1	54.1±2.8	53.6±3.8	61.8±1.2	**64.6±0.7**
		All	65.2±0.4	62.5±1.3	59.2±0.2	54.3±3.4	55.6±1.6	60.5±1.9	66.2±0.8	**66.8±0.5**
Amazon Elec.	5w1s	Base	60.1±1.8	58.4±0.9	59.7±0.3	19.1±2.1	22.5±3.1	60.1±1.8	**65.8±2.1**	63.9±1.0
		Novel	7.8±0.8	5.1±1.1	6.4±0.3	16.6±5.4	15.3±6.7	7.8±0.8	8.0±0.7	**19.7±1.6**
		All	27.2±0.4	24.8±0.4	26.2±0.2	17.5±3.8	18.0±5.0	27.2±0.4	29.4±1.4	**36.1±1.1**
	5w3s	Base	64.2±1.8	55.1±0.9	63.0±0.7	43.7±1.6	43.6±2.4	63.0±1.4	**69.1±1.6**	69.0±2.9
		Novel	21.6±1.5	13.3±2.0	23.1±0.2	32.7±4.8	28.1±5.6	12.0±4.0	29.8±2.7	**40.7±2.2**
		All	37.4±1.5	28.8±1.4	37.9±0.3	36.8±3.4	33.9±3.5	30.9±2.5	44.3±1.8	**51.2±2.3**
	10w1s	Base	64.4±1.2	59.7±1.3	53.1±1.6	18.5±1.4	20.8±2.2	64.4±1.2	**69.0±0.9**	61.3±0.8
		Novel	8.0±1.3	5.7±1.1	4.9±0.1	15.3±3.7	15.0±3.7	8.0±1.3	11.3±1.5	**15.4±0.3**
		All	34.4±1.0	31.0±1.1	27.7±0.2	16.8±2.3	17.7±2.0	34.4±1.0	**38.3±1.2**	36.9±0.4
	10w3s	Base	58.6±0.4	55.2±0.9	48.8±0.7	43.8±1.7	46.3±1.6	62.9±0.7	**72.3±1.1**	66.5±0.1
		Novel	22.4±1.1	14.8±1.0	16.5±0.2	27.5±2.9	26.2±3.2	13.8±0.3	20.3±2.4	**38.1±2.3**
		All	39.4±0.5	33.7±0.7	31.6±0.4	35.1±1.4	35.6±1.8	36.8±0.2	44.7±1.5	**51.4±1.1**
Aminer	5w1s	Base	40.8±1.0	38.2±1.7	40.4±0.4	19.4±1.5	25.4±1.7	40.8±1.0	**40.9±1.0**	36.1±1.1
		Novel	24.8±2.3	12.4±2.3	7.6±0.2	20.0±6.4	22.2±4.6	24.8±2.3	29.7±1.2	**37.2±1.6**
		All	32.5±1.3	24.8±1.1	23.5±0.3	19.7±3.1	23.7±2.6	32.5±1.3	35.7±0.6	**36.7±1.3**
	5w3s	Base	42.5±1.6	39.8±1.5	**42.6±0.4**	23.0±1.9	38.2±1.4	39.2±1.5	42.0±1.1	39.6±0.2
		Novel	33.1±0.9	29.3±2.5	33.4±0.3	21.4±4.2	34.9±4.4	36.3±1.3	36.2±0.8	**44.4±0.3**
		All	37.6±0.9	34.4±0.8	37.8±0.2	22.2±2.1	36.5±2.4	37.7±0.4	39.0±0.9	**42.1±0.3**
	10w1s	Base	41.2±1.3	41.0±0.8	**42.6±0.4**	19.6±1.5	23.4±1.8	41.2±1.3	40.3±1.6	40.4±0.4
		Novel	11.2±1.2	4.1±1.3	11.6±0.3	16.1±4.0	15.5±4.4	11.2±1.2	16.8±0.6	**21.8±1.1**
		All	25.7±0.6	22.0±0.7	26.6±0.2	17.8±2.2	19.3±2.3	25.7±0.6	28.2±1.1	**30.7±0.5**
	10w3s	Base	41.7±1.2	43.1±1.5	42.4±0.2	26.3±1.5	34.7±1.5	39.1±0.5	**46.2±0.8**	40.4±0.6
		Novel	21.6±0.3	17.7±1.8	24.5±0.3	18.7±3.8	23.4±4.8	23.0±1.4	21.6±0.6	**27.8±1.0**
		All	31.3±0.6	30.0±1.1	33.1±0.2	22.4±2.2	28.9±2.1	30.8±0.9	33.5±0.6	**33.9±0.7**
Cora Full	5w1s	Base	70.6±1.0	**72.7±1.3**	71.4±0.2	40.2±3.1	55.5±2.7	70.6±1.0	71.1±0.8	70.1±2.1
		Novel	25.3±0.8	21.3±0.9	20.8±0.3	12.9±4.7	19.0±6.6	25.3±0.8	33.2±1.4	**34.8±0.4**
		All	46.3±0.7	45.1±0.9	44.3±0.2	25.6±3.2	35.9±3.3	46.3±0.7	50.8±1.1	**51.2±0.8**
	5w3s	Base	73.3±1.1	76.5±1.3	70.1±0.2	44.6±2.0	55.5±1.5	73.0±0.9	70.8±0.4	**78.1±0.2**
		Novel	26.5±1.4	21.1±2.4	23.6±0.6	16.1±2.7	24.8±4.4	9.7±0.7	**32.0±2.3**	30.7±2.3
		All	48.2±1.1	46.8±1.7	45.2±0.3	29.3±1.6	39.1±2.5	39.1±0.3	50.0±1.3	**52.6±1.4**
	10w1s	Base	75.9±0.8	**76.2±0.8**	70.1±0.2	45.2±3.0	52.1±1.7	75.9±0.8	**76.2±0.7**	69.6±0.1
		Novel	20.1±1.7	15.1±1.3	11.5±0.2	14.6±2.3	16.9±3.1	20.1±1.7	**29.8±0.9**	24.3±0.5
		All	40.7±0.7	44.6±0.8	39.8±0.1	29.4±1.4	33.9±1.4	40.7±0.7	**52.2±0.1**	46.2±0.3
	10w3s	Base	74.8±0.6	74.7±0.6	68.0±0.3	42.8±1.5	52.7±1.8	67.5±0.5	**77.9±0.6**	74.4±0.4
		Novel	37.9±1.4	28.1±1.2	34.1±0.3	19.4±2.9	18.6±2.9	7.5±1.0	27.7±2.0	**40.0±1.4**
		All	55.7±0.7	50.7±0.7	50.5±0.2	30.7±1.2	35.0±1.3	36.5±0.7	52.0±0.7	**56.6±0.6**

TABLE III: Performance comparison (mean±std accuracy (%)) on four datasets. 'NwKs' represents the 'N-way K-shot' setup for *novel classes*. Base, Novel, All represent accuracy on the base, novel, and all the classes, respectively. The best and the second-best results are bold and underlined respectively. Our goal is to improve the accuracy on the novel classes and at the same time keep competitive performance on the base classes.

classes (beyond novel vs. base classes)? (2) How to select g_1 (i.e., the preliminary classifier for weight assigner g)?

We use the following three metrics to measure the uncertainty in prediction vector $\mathbf{z}$: (1) Max: the largest entry of $\mathbf{z}$; (2) Gap: the gap between the largest entry and the second largest entry of $\mathbf{z}$; (3) Entropy: $-\sum_i \mathbf{z}[i]\log(\mathbf{z}[i])$. To clearly illustrate the influence of propagation on the prediction uncertainty, we select three direct models: (1) $\texttt{MLP}(\mathbf{X})$, (2) $\texttt{MLP}(\tilde{\mathbf{A}}\mathbf{X})$, (3) $\texttt{MLP}(\tilde{\mathbf{A}}\tilde{\mathbf{A}}\mathbf{X})$, where $\mathbf{X}$ is the node feature matrix and $\tilde{\mathbf{A}}$ is symmetrically normalized adjacency matrix. The `softmax` is applied to normalize the predictions. To measure the epistemic uncertainty, we adopt the dropout variational inference [18], [22] as we have introduced in Section III-B.

Our experiment design is as follows. First, 50 nodes are randomly selected from every class as test nodes. Second, half of the classes are randomly selected as the novel classes

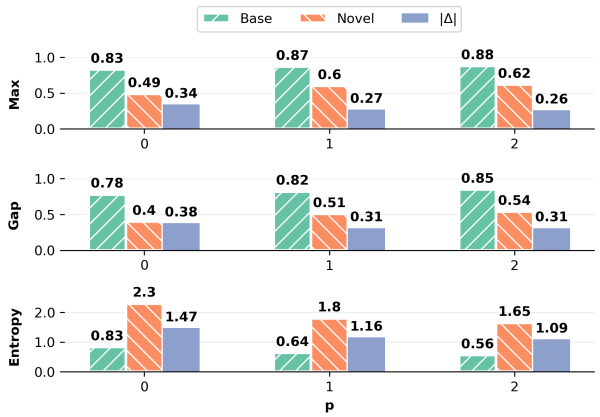

Fig. 3: Uncertainty comparison between classifiers with different propagation steps. For 'Max' and 'Gap', the smaller the more uncertain. For 'Entropy', the larger the more uncertain. Our goal is to select p where $|\Delta|$ is the largest.

and the remaining classes are the base classes. The training nodes are composed of 1 node per novel class and 50 nodes per base class. Third, the aforementioned three models are trained on the training nodes, and their prediction uncertainty on test nodes is evaluated by the three metrics (i.e., Max, Gap, Entropy) with the dropout variational inference. We repeat the above steps in 10 runs to report the average uncertainties on novel and base classes respectively. Notice that in every iteration, *the base/novel split is different*.

The experimental results are shown in Figure 3 where x-axis represents models $\texttt{MLP}(\mathbf{X})$, $\texttt{MLP}(\tilde{\mathbf{A}}\mathbf{X})$, and $\texttt{MLP}(\tilde{\mathbf{A}}\tilde{\mathbf{A}}\mathbf{X})$ respectively (i.e., the propagation step is $p = 0$, $p = 1$ and $p = 2$ respectively). $|\Delta|$ denotes the average uncertainty gap between base and novel classes. We have the following observations. First, generally, the uncertainty on the novel classes is much higher than that on the base classes. The above finding directly motivates our model design to set the uncertainty-related module (i.e., ϕ_2 in the weight assigner g) as a meta-learner and learn high-level knowledge (i.e., knowledge can be extracted from the broad base classes by constructing imbalanced episodes) to empower the shot-aware node classifier STAGER. Second, with the increment of propagation steps, the classifier (i.e. MLP) is less uncertain on both the novel and base classes. Importantly, the uncertainty gap $|\Delta|$ is also reduced. Therefore, to let the weight assigner g be sensitive about the difference of shots (reflected by the uncertainty) between the base and novel classes, we set g_1 as $\texttt{MLP}(\mathbf{X})$ to retain the uncertainty gap as large as possible.

D. Ablation Study

Weight assigner. The weight assigner g is the key component of STAGER. If we remove the weight assigner and directly set the weights assigned to receptive fields as trainable parameters, our model will degenerate to GPRGNN [8]. Hence, we study the performance comparison between GPRGNN [8] and STAGER(without imbalanced episodic training) for the 10-way 3-shot setup and present the accuracy on the novel classes and all the classes in Table IVa. Notice that the data is re-organized from Table III where Amazon-C and Amazon-E represent Amazon Clothing and Amazon Electronics datasets, respectively. Clearly, we observe that with the weight assigner, in most cases, the accuracy on novel classes and all the classes gets boosted significantly.

Dataset	STAGER (without g)		STAGER (with g)	
	Novel	All	Novel	All
Amazon-C	58.0±1.5	62.5±1.3	61.8±1.2 (+3.8)	66.2±0.8 (+3.7)
Amazon-E	14.8±1.0	33.7±0.7	20.3±2.4 (+5.5)	44.7±1.5 (+11.0)
Aminer	16.1±1.7	28.2±1.1	21.6±0.6 (+5.5)	33.5±0.6 (+5.3)
Cora-full	28.1±1.2	50.7±0.7	27.7±2.0 (-0.4)	52.0±0.7 (+1.3)

(a) Weight assigner g

Dataset	STAGER-I (without rank)		STAGER-I (with rank)	
	Novel	All	Novel	All
Amazon-C	57.8±1.1	61.7±0.7	64.6±0.7 (+6.8)	66.8±0.5 (+5.1)
Amazon-E	20.6±1.5	43.2±1.0	38.1±2.3 (+17.5)	51.4±1.1 (+8.2)
Aminer	23.6±0.9	35.9±0.7	27.8±1.0 (+4.2)	33.9±0.7 (-2.0)
Cora-full	34.8±0.9	49.1±0.8	40.0±1.4 (+5.2)	56.6±0.6 (+7.5)

(b) Rank operator

TABLE IV: Ablation study on the weight assigner g (a) and the rank operator (b) (mean±std accuracy (%)). The number in parentheses indicates the performance comparison with the ablated variants in the left columns.

Dataset	STAGER		STAGER-I	
	Novel	All	Novel	All
Amazon-C	61.8±1.2	66.2±0.8	64.6±0.7 (+2.8)	66.8±0.5 (+0.6)
Amazon-E	20.3±2.4	44.7±1.5	38.1±2.3 (+17.8)	51.4±1.1 (+6.7)
Aminer	21.6±0.6	33.5±0.6	27.8±1.0 (+6.2)	33.9±0.7 (+0.4)
Cora-full	27.7±2.0	52.0±0.7	40.0±1.4 (+12.3)	56.6±0.6 (+4.6)

(a) Imbalanced episodic training for STAGER

Dataset	MetaGNN		MetaGNN-I	
	Novel	All	Novel	All
Amazon-C	53.6±0.2	59.2±0.2	56.4±0.3 (+2.8)	61.9±0.4 (+2.7)
Amazon-E	16.5±0.2	31.6±0.4	22.7±0.4 (+6.2)	42.5±0.6 (+10.9)
Aminer	40.2±0.5	38.1±0.2	36.5±0.8 (-3.7)	37.0±0.7 (-1.1)
Cora-full	34.1±0.3	50.5±0.2	33.1±0.5 (-1.0)	40.0±0.8 (-10.5)

(b) Imbalanced episodic training for MetaGNN

TABLE V: Ablation study on the imbalanced episodic training for STAGER (a) and MetaGNN (b) (mean±std accuracy (%)). The number in parentheses indicates the performance comparison with the ablated variants in the left columns.

Rank operator. We claimed in the main content that the rank operator is designed for extracting uncertainty from the prediction vectors. Here, we study the performance comparison between STAGER-I (without rank operator) and STAGER-I (with rank operator) for the 10-way 3-shot setup and present the accuracy on the novel classes and all the classes in Table IVb where Amazon-C and Amazon-E represent Amazon Clothing and Amazon Electronics datasets, respectively. Clearly, we observe that the rank operator, in most cases, can significantly boost the accuracy on novel classes and all the classes.

Imbalanced episodic training. To study the effectiveness of imbalanced episodic training, first, we study the performance

comparison between STAGER (without imbalanced episodic training) and STAGER-I (with imbalanced episodic training) for the 10-way 3-shot setup. The accuracy on the novel classes and all the classes is presented in Table Va and this part of the data is re-organized from Table III; then we implement imbalanced episodic training on MetaGNN (notated as MetaGNN-I) for the 10-way 3-shot setup to see the performance comparison on the novel classes and all the classes in Table Vb. Amazon-C and Amazon-E represent Amazon Clothing and Amazon Electronics datasets, respectively.

We observe that for STAGER-I, compared with STAGER, in most cases accuracy on the novel classes and that on all the classes gets improved, which demonstrates the effectiveness of imbalanced episodic training. For MetaGNN-I we observe that the performance improvement is not stable and may even hurt the performance dramatically. The key reason is that existing few-shot node classifiers are designed for balanced scenarios, and not be able to address the imbalanced distribution of shots between the base classes and the novel classes. Hence, importing imbalanced shot distribution into the meta-training phase may even affect the learning of existing few-shot node classifiers. We also tried to implement imbalanced episodic training on GPN [10] and G-META [21] but they suffer from the out-of-memory problem due to the great number of labeled base nodes so we do not report them here.

E. Hyperparameter sensitivity study.

We conduct hyperparameter sensitivity study on the steps of propagation p from Eq. (4c), which controls the size of the set of prediction matrices $\{\mathbf{H}^{(0)}, \ldots, \mathbf{H}^{(p)}\}$. We search p from $[7, 12]$ and present the accuracy of the base, novel, and all classes on the Cora-full dataset. From Figure 4, we observe that in general, performance of STAGER is stable w.r.t. p, and when $p = 9$ our model obtains the best performance.

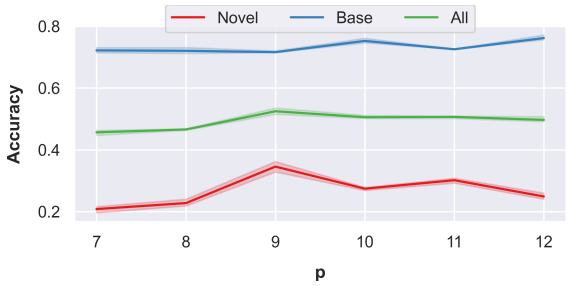

Fig. 4: Sensitivity study of the propagation steps p.

V. RELATED WORK

Few-shot learning on graphs. Few-shot learning (FSL) is a classic problem [37]. Its concrete instantiations on graphs grab increasing attention. MetaGNN [47] learns to initialize a graph neural network. AMMGNN [41] learns to generate task-specific attribute matrices. G-META [21] and GPN [10] collapses classes into topology-aware prototypes. Lan et al. [25] introduce an embedding transformation function for the learning of novel node embeddings. Yao et al. [43] transfer knowledge from auxiliary graphs to the target graphs. Recently, Liu [26] explore that location embeddings can be transferable priori across tasks. Besides, for the few-shot graph classification problem, Chauhan et al. [7] cluster graphs into super-classes based on the spectrum of the normalized Laplacian. Bose et al. [5] explore the link prediction task given limited known edges.

GraphSMOTE [46] investigates a similar problem called *imbalanced node classification*. Here, we clarify the connections and differences between GraphSMOTE and our work. GraphSMOTE [46] focuses on augmenting the minority classes (i.e., novel classes in our setting), and particularly it interpolates the minority nodes to increase the size of minority classes. We model the problem in a meta-learning fashion to fully exploit the knowledge from base classes. Additionally, both problems are about the node classification on imbalanced label distributions, but the number of shots of novel (i.e., minority) classes in our setting is much smaller (usually less than 10, or even 1) than the imbalanced node classification settings, which limits the effectiveness of GraphSMOTE since it mainly exploits the information from the labeled minority nodes themselves. From another perspective, if the novel classes are given few shots, augmenting it to have the balanced shots as base classes will dramatically increase the size of the network, and negatively impact the model efficiency. Besides, ImGAGN [31] studies the node-level representation task on imbalanced node distribution.

Generalized few-shot learning. Recently, with the development of memory and generative models, more works are developed for the generalized few-shot learning problem [13]. For instance, Wang et al. [42] propose a hallucinator component to augment the few-shot classes; Gidaris and Komodakis [19] address this problem by a two-stage training paradigm; Liu et al. [27] make one step further with memory component and study an open world recognition task. In addition, ONCE [30] addresses a problem named incremental few-shot learning where the revisiting of base classes is not allowed. Tan et al. [34] study a long-tailed rare category objective recognition problem. The tasks of above works are all defined on non-graph data. Nonetheless, our proposed imbalanced episodic training could benefit these existing or even future solutions formulated from the meta-learning perspective.

VI. CONCLUSION

In this paper, we study the generalized few-shot node classification problem and propose a novel model named STAGER. By equipping a shot-aware module, weight assigner, the proposed STAGER effectively mitigates the bias brought by the imbalanced distribution of the number of nodes from various classes and adapts to distinct weights assignments for both the many-shot scenario and the few-shot scenario simultaneously. To ensure the meta-training is consistent with the meta-test, we propose imbalanced episodic training which generalizes the conventional episodic training in meta-learning. Comprehensive experiments on four real-world datasets demonstrate the effectiveness of the proposed STAGER and imbalanced episodic training.

ACKNOWLEDGMENT

This work is supported by NSF (1947135, 2106825, and 2134079), the NSF Program on Fairness in AI in collaboration with Amazon (1939725), DARPA (HR001121C0165), NIFA (2020-67021-32799), ARO (W911NF2110088, W911NF2110030), ONR (N00014-21-1-4002), and ARL (W911NF2020124). The content of the information in this document does not necessarily reflect the position or the policy of the Government or Amazon, and no official endorsement should be inferred. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

REFERENCES

[1] Charu C Aggarwal and Nan Li. On node classification in dynamic content-based networks. In *SDM*, 2011.

[2] Ahmed Alaa and Mihaela Van Der Schaar. Discriminative jackknife: Quantifying uncertainty in deep learning via higher-order influence functions. In *ICML*, 2020.

[3] Smriti Bhagat, Graham Cormode, and S Muthukrishnan. Node classification in social networks. In *Social network data analytics*, pages 115–148. Springer, 2011.

[4] Aleksandar Bojchevski and Stephan Günnemann. Deep gaussian embedding of graphs: Unsupervised inductive learning via ranking. In *ICLR*, 2018.

[5] Avishek Joey Bose, Ankit Jain, Piero Molino, and William L Hamilton. Meta-graph: Few shot link prediction via meta learning. *arXiv preprint arXiv:1912.09867*, 2019.

[6] Wei-Lun Chao, Soravit Changpinyo, Boqing Gong, and Fei Sha. An empirical study and analysis of generalized zero-shot learning for object recognition in the wild. In *ECCV*, 2016.

[7] Jatin Chauhan, Deepak Nathani, and Manohar Kaul. Few-shot learning on graphs via super-classes based on graph spectral measures. In *ICLR*, 2020.

[8] Eli Chien, Jianhao Peng, Pan Li, and Olgica Milenkovic. Adaptive universal generalized pagerank graph neural network. In *ICLR*, 2021.

[9] Kaize Ding, Jianling Wang, James Caverlee, and Huan Liu. Meta propagation networks for graph few-shot semi-supervised learning. AAAI, 2022.

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

[11] Kaize Ding, Zhe Xu, Hanghang Tong, and Huan Liu. Data augmentation for deep graph learning: A survey. *arXiv preprint arXiv:2202.08235*, 2022.

[12] Kaize Ding, Qinghai Zhou, Hanghang Tong, and Huan Liu. Few-shot network anomaly detection via cross-network meta-learning. In *TheWebConf*, 2021.

[13] Zhibo Fan, Yuchen Ma, Zeming Li, and Jian Sun. Generalized few-shot object detection without forgetting. In *CVPR*, 2021.

[14] Rafael Felix, Ian Reid, Gustavo Carneiro, et al. Multi-modal cycle-consistent generalized zero-shot learning. In *ECCV*, 2018.

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

[16] Luca Franceschi, Michele Donini, Paolo Frasconi, and Massimiliano Pontil. Forward and reverse gradient-based hyperparameter optimization. In *ICML*. PMLR, 2017.

[17] Luca Franceschi, Paolo Frasconi, Saverio Salzo, Riccardo Grazzi, and Massimiliano Pontil. Bilevel programming for hyperparameter optimization and meta-learning. In *ICML*. PMLR, 2018.

[18] Yarin Gal and Zoubin Ghahramani. Bayesian convolutional neural networks with bernoulli approximate variational inference. *arXiv preprint arXiv:1506.02158*, 2015.

[19] Spyros Gidaris and Nikos Komodakis. Dynamic few-shot visual learning without forgetting. In *CVPR*, 2018.

[20] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural networks*, 1989.

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

[22] Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision? In *NIPS*, 2017.

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

[24] Johannes Klicpera, Aleksandar Bojchevski, and Stephan Günnemann. Predict then propagate: Graph neural networks meet personalized pagerank. In *ICLR*, 2018.

[25] Lin Lan, Pinghui Wang, Xuefeng Du, Kaikai Song, Jing Tao, and Xiaohong Guan. Node classification on graphs with few-shot novel labels via meta transformed network embedding. *NeurIPS*, 2020.

[26] Zemin Liu, Yuan Fang, Chenghao Liu, and Steven CH Hoi. Relative and absolute location embedding for few-shot node classification on graph. In *AAAI*, 2021.

[27] Ziwei Liu, Zhongqi Miao, Xiaohang Zhan, Jiayun Wang, Boqing Gong, and Stella X Yu. Large-scale long-tailed recognition in an open world. In *CVPR*, 2019.

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

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

[30] Juan-Manuel Perez-Rua, Xiatian Zhu, Timothy M Hospedales, and Tao Xiang. Incremental few-shot object detection. In *CVPR*, 2020.

[31] Liang Qu, Huaisheng Zhu, Ruiqi Zheng, Yuhui Shi, and Hongzhi Yin. Imgagn: Imbalanced network embedding via generative adversarial graph networks. In Feida Zhu, Beng Chin Ooi, and Chunyan Miao, editors, *SIGKDD*, 2021.

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

[33] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *JMLR*, 2014.

[34] Jingru Tan, Changbao Wang, Buyu Li, Quanquan Li, Wanli Ouyang, Changqing Yin, and Junjie Yan. Equalization loss for long-tailed object recognition. In *CVPR*, 2020.

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

[36] Jiliang Tang, Charu Aggarwal, and Huan Liu. Node classification in signed social networks. In *SDM*, 2016.

[37] Sebastian Thrun. Is learning the n-th thing any easier than learning the first? In *NIPS*, 1996.

[38] Sebastian Thrun. Lifelong learning algorithms. In *Learning to learn*, pages 181–209. Springer, 1998.

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

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

[41] Ning Wang, Minnan Luo, Kaize Ding, Lingling Zhang, Jundong Li, and Qinghua Zheng. Graph few-shot learning with attribute matching. In *CIKM*, 2020.

[42] Yu-Xiong Wang, Ross Girshick, Martial Hebert, and Bharath Hariharan. Low-shot learning from imaginary data. In *CVPR*, 2018.

[43] Huaxiu Yao, Chuxu Zhang, Ying Wei, Meng Jiang, Suhang Wang, Junzhou Huang, Nitesh Chawla, and Zhenhui Li. Graph few-shot learning via knowledge transfer. In *AAAI*, 2020.

[44] Han-Jia Ye, Hexiang Hu, and De-Chuan Zhan. Learning adaptive classifiers synthesis for generalized few-shot learning. *International Journal of Computer Vision*, pages 1–24, 2021.

[45] Yunlong Yu, Zhong Ji, Jungong Han, and Zhongfei Zhang. Episode-based prototype generating network for zero-shot learning. In *CVPR*, 2020.

[46] Tianxiang Zhao, Xiang Zhang, and Suhang Wang. Graphsmote: Imbalanced node classification on graphs with graph neural networks. In *WSDM*, 2021.

[47] Fan Zhou, Chengtai Cao, Kunpeng Zhang, Goce Trajcevski, Ting Zhong, and Ji Geng. Meta-gnn: On few-shot node classification in graph meta-learning. In *CIKM*, 2019.