Mastering Long-Tail Complexity on Graphs: Characterization, Learning, and Generalization

Haohui Wang
Virginia Tech
Blacksburg, United States
haohuiw@vt.edu

Baoyu Jing
University of Illinois Urbana-Champaign
Urbana, United States
baoyu.jing@hotmail.com

Kaize Ding
Northwestern University
Evanston, United States
kaize.ding@northwestern.edu

Yada Zhu
IBM Research
Yorktown Heights, United States
yzhu@us.ibm.com

Wei Cheng
NEC Labs America
Princeton, United States
weicheng@nec-labs.com

Si Zhang
Meta
Sunnyvale, United States
sizhang@meta.com

Yonghui Fan
Amazon AGI
Sunnyvale, United States
fhjfyh@hotmail.com

Liqing Zhang
Virginia Tech
Blacksburg, United States
lqzhang@cs.vt.edu

Dawei Zhou
Virginia Tech
Blacksburg, United States
zhoud@vt.edu

Abstract

In the context of long-tail classification on graphs, the vast majority of existing work primarily revolves around the development of model debiasing strategies, intending to mitigate class imbalances and enhance the overall performance. Despite the notable success, there is very limited literature that provides a theoretical tool for characterizing the behaviors of long-tail classes in graphs and gaining insight into generalization performance in real-world scenarios. To bridge this gap, we propose a generalization bound for long-tail classification on graphs by formulating the problem in the fashion of multi-task learning, i.e., each task corresponds to the prediction of one particular class. Our theoretical results show that the generalization performance of long-tail classification is dominated by the overall loss range and the task complexity. Building upon the theoretical findings, we propose a novel generic framework HierTail for long-tail classification on graphs. In particular, we start with a hierarchical task grouping module that allows us to assign related tasks into hypertasks and thus control the complexity of the task space; then, we further design a balanced contrastive learning module to adaptively balance the gradients of both head and tail classes to control the loss range across all tasks in a unified fashion. Extensive experiments demonstrate the effectiveness of HierTail in characterizing long-tail classes on real graphs, which achieves up to 12.9% improvement over the leading baseline method in balanced accuracy.

KDD '24, August 25–29, 2024, Barcelona, Spain

ACM ISBN 979-8-4007-0490-1/24/08
https://doi.org/10.1145/3637528.3671880

CCS Concepts

• **Computing methodologies** → **Machine learning**; • **Information systems** → **Data mining**; • **Theory of computation** → **Graph algorithms analysis**.

Keywords

Long-tail Learning, Generalization, Graph Mining

ACM Reference Format:
Haohui Wang, Baoyu Jing, Kaize Ding, Yada Zhu, Wei Cheng, Si Zhang, Yonghui Fan, Liqing Zhang, and Dawei Zhou. 2024. Mastering Long-Tail Complexity on Graphs: Characterization, Learning, and Generalization. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '24), August 25–29, 2024, Barcelona, Spain.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3637528.3671880

1 Introduction

The graph serves as a fundamental data structure for modeling a diverse range of relational data, ranging from financial transaction networks [8, 47] to social science [10]. In recent years, Graph Neural Networks (GNNs) have achieved outstanding performance on node classification tasks [18, 52, 54, 65] because of their ability to learn expressive representations from graphs. Despite the remarkable success, the performance of GNNs is primarily attributed to the availability of high-quality and abundant annotated data [11, 16, 19, 42, 53]. Nevertheless, unlike many graph benchmark datasets developed in the lab environment, it is often the case that many high-stake domains naturally exhibit a long-tail distribution, i.e., a few head classes (the majority classes) with rich and well-studied data and massive tail classes (the minority classes) with scarce and under-explored data. For example, in financial transaction networks, a few head classes correspond to the normal transaction types (e.g., credit card payment, wire transfer), and the numerous tail classes can represent a variety of fraudulent transaction types (e.g., money laundering, synthetic identity

Figure 1: An illustrative figure of long-tail distribution in the collaboration network (Cora-Full), where the green and red curves show balanced accuracy (bAcc) (%) of GCN and HierTail for node classification on each class. Blue and yellow bars represent the class frequency of unlabeled and labeled nodes.

transaction). Despite the rare occurrences of fraudulent transactions, detecting them can prove crucial [1, 44]. Another example is the collaboration network. As shown in Figure 1, the Cora-Full network [4] encompasses 70 classes categorized by research areas, showcasing a starkly imbalanced data distribution—from as few as 15 papers in the least represented area to as many as 928 papers in the most populated one. The task complexity (massive number of classes, data imbalance) coupled with limited supervision imposes significant computational challenges on GNNs.

Important as it could be, there is limited literature that provides a theoretical grounding to characterize the behaviors of long-tail classes on graphs and understand the generalization performance in real environments. To bridge the gap, we provide insights and identify three fundamental challenges in the context of long-tail classification on graphs. First (***C1. Highly skewed data distribution***), the data exhibits extremely skewed class memberships. Consequently, the head classes contribute more to the learning objective and can be better characterized by GNNs; the tail classes contribute less to the objective and thus suffer from higher systematic errors [63]. Second (***C2. Label scarcity***), due to the rarity and diversity of tail classes in nature, it is often more expensive and time-consuming to annotate tail classes than head classes [38]. What is worse, training GNNs from scarce labels may result in representation disparity and inevitable errors [48, 50, 50, 66, 67], which amplifies the difficulty of debiasing GNN from the highly skewed data distribution. Third (***C3. Task complexity***), with the increasing number of classes under the long-tail setting, the difficulty of separating the margin [14] of classes is dramatically increasing. There is a high risk of encountering overlapped regions between classes with low prediction confidence [35, 62]. To deal with the long-tail classes, the existing literature mainly focuses on augmenting the observed graph [40, 51, 64] or reweighting the class-wise loss functions [43, 60]. Despite the existing achievements, a natural research question is that: *can we further improve the overall performance by learning more knowledge from both head classes and tail classes?*

To answer the aforementioned question, we provide a generalization bound of long-tail classification on graphs. The key idea is to formulate the long-tail classification problem in the fashion of multi-task learning [26, 45], i.e., each task corresponds to the prediction of one specific class. In particular, the generalization bound is in terms of the range of losses across all tasks and the complexity of the task space. Building upon the theoretical findings, we propose HierTail, a generic learning framework to characterize long-tail classes on graphs. Specifically, motivated by controlling the complexity of the task space, we employ a hierarchical structure for task grouping to tackle C2 and C3. It assigns related tasks into hypertasks, allowing the information learned in one class to help train another class, particularly benefiting tail classes. Furthermore, we implement a balanced contrastive module to address C1 and C2, which effectively balances the contributions of head classes and tail classes to the gradient. This module reduces the loss of tail tasks while ensuring the performance of head tasks, thus controlling the range of losses across all tasks.

The main contributions of this paper are summarized below.

- **Problem Definition.** We formalize the long-tail classification problem on graphs and develop a novel metric named long-tailedness ratio for characterizing properties of long-tail distributed data.
- **Theory.** We derive a generalization bound for long-tail classification on graphs, which inspires our proposed framework.
- **Algorithm.** We propose a novel approach named HierTail that (1) extracts shareable information across classes via hierarchical task grouping and (2) balances the contributions of head classes and tail classes to the gradient.
- **Evaluation.** We systematically evaluate the performance of HierTail with eleven baseline models on six real-world datasets for long-tail classification on graphs. The results demonstrate the effectiveness of HierTail and verify our theoretical findings.

2 Preliminary

In this section, we introduce the background and give the formal problem definition. We represent a graph as $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{X})$, where $\mathcal{V}$ represents the set of nodes, $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ represents the set of edges, $\mathbf{X} \in \mathbb{R}^{n \times d}$ represents the node feature matrix, n is the number of nodes, and d is the feature dimension. $\mathbf{A} \in \{0,1\}^{n \times n}$ is the adjacency matrix, where $\mathbf{A}_{ij} = 1$ if there is an edge $\mathbf{e}_{ij} \in \mathcal{E}$ from $\mathbf{v}_i$ to $\mathbf{v}_j$ in $\mathcal{G}$ and $\mathbf{A}_{ij} = 0$ otherwise. $\mathcal{Y} = \{y_1, \ldots, y_n\}$ is the set of labels, $y_i \in \{1, \ldots, T\}$ is the label of the i^{th} node. There are T classes in total, and T can be notably large.

Long-Tail Classification refers to the classification problem in the presence of a massive number of classes, highly skewed class-membership distribution, and label scarcity. Here we let $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{n}$ represent a dataset with long-tail distribution. We define $\mathcal{D}_t$ as the set of instances belonging to class t. Without the loss of generality, we have $\mathcal{D} = \{\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_T\}$, where $|\mathcal{D}_1| \geq |\mathcal{D}_2| \geq \cdots \gg |\mathcal{D}_T|$, $\sum_{t=1}^{T} |\mathcal{D}_t| = n$. Tail classes may encounter label scarcity, having few or even only one instance, while head classes have abundant instances. To measure the skewness of long-tail distribution, Wu et al. [51] introduces the Class-Imbalance Ratio as $\frac{\min_t(|\mathcal{D}_t|)}{\max_t(|\mathcal{D}_t|)}$, i.e., the ratio of the size of the smallest minority class to the size of the largest majority class.

Long-Tailedness Ratio. Suppose we are given a graph $\mathcal{G}$ with long-tail class-membership distribution. While Class-Imbalance Ratio [51] measures the imbalance level of observed data, it overlooks the task complexity in the task of long-tail classification. As the

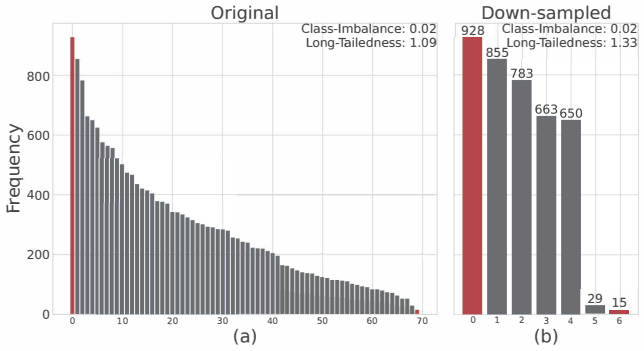

Figure 2: Comparison between two long-tail distribution metrics on (a) the hard case of the original Cora-Full dataset and (b) the easy case of the down-sampled Cora-Full dataset. We observe that the class-imbalance ratio falls short in characterizing the task complexity of two datasets, while the long-tailedness ratio does.

number of classes increases, the difficulty of the classification task therefore increases. Taking the Cora-Full collaboration network as shown in Figure 2 as an example, we down-sampled 7 classes from the original Cora-Full dataset. Although the class-imbalance ratio remains the same, i.e., 0.02 for both the original and down-sampled datasets, the task complexity varies significantly, i.e., 70 classes in Figure 2 (a) v.s 7 classes in Figure 2 (b). In light of this, we introduce a novel quantile-based metric named the long-tailedness ratio to jointly quantify the class-imbalance ratio and task complexity for the long-tail datasets. The formal definition of the long-tailedness ratio is provided as follows:

Definition 1 (Long-Tailedness Ratio)**.** *Suppose we have a dataset $\mathcal{D}$ with long-tail classes that follow a descending order in terms of the number of instances. The long-tailedness ratio is*

$$\texttt{Ratio}_{LT}(p) = \frac{Q(p)}{T - Q(p)}, \tag{1}$$

where $Q(p) = min\{y : Pr(\mathcal{Y} \leq y) = p, 1 \leq y \leq T\}$ is the quantile function of order $p \in (0, 1)$ for variable $\mathcal{Y}$, T is the number of classes. The numerator represents the number of classes to which p percent instances belong, and the denominator represents the number of classes to which the else $(1 - p)$ percent instances belong in $\mathcal{D}$.

Essentially, the long-tailedness ratio implies the task complexity of long-tail classification and characterizes two properties of $\mathcal{D}$: (1) class-membership skewness, (2) # of classes. Intuitively, the higher the skewness of data distribution, the lower the ratio will be; the higher the complexity of the task space (i.e., massive number of classes), the lower the long-tailedness ratio. Figure 2 provides a case study on the Cora-Full dataset by comparing the long-tailedness ratio and class-imbalance ratio [51]. In general, we observe that the long-tailedness ratio better characterizes the differences on the original Cora dataset ($\texttt{Ratio}_{LT}(0.8) = 1.09$) and its down-sampled dataset ($\texttt{Ratio}_{LT}(0.8) = 1.33$). In our implementation, we choose $p = 0.8$ following the Pareto principle [36]. In Appendix A, we additionally offer insights into the utilization of the long-tailedness ratio for the enhanced comprehension of long-tail datasets and as a guiding factor for model selection in practice.

3 Algorithm

3.1 Theoretical Analysis

In this paper, we consider the long-tail problems with data imbalance and massive classes, an area with limited theoretical exploration. For the first time, we propose to reformulate the long-tail problems in the manner of multi-task learning, thereby leveraging the theoretical foundation of multi-task learning to gain insights into long-tail problems. In particular, we view the classification for each class as a learning task[1] on graph $\mathcal{G}$. A key assumption of multi-task learning is task relatedness, i.e., relevant tasks should share similar model parameters. Similarly, in long-tail learning, we aim to learn the related tasks (classes) concurrently to potentially enhance the performance of each task (classes). We propose to formulate the hypothesis g of long-tail model as $g = \{f_t\}_{t=1}^{T} \circ h$, where $\circ$ is the functional composition, $g_t(x) = f_t \circ h(x) \equiv f_t(h(x))$ for each classification task. The function $h : \mathbf{X} \to \mathbb{R}^K$ is the representation extraction function shared across different tasks, $f : \mathbb{R}^K \to \mathbb{R}$ is the task-specific predictor, and K is the dimension of the hidden layer. The training set for the t^{th} task $\mathcal{D}_t = \{(\mathbf{x}_i^t, y_i^t)\}_{i=1}^{n_t}$ contains n_t annotated nodes, $\mathbf{x}_i^t$ is the i^{th} training node in class t, and $y_i^t = t$ for all i. The task-averaged risk of representation h and predictors $f_1, \ldots, f_T$ is defined as $\epsilon(h, f_1, \ldots, f_T)$, and the corresponding empirical risk is defined as $\hat{\epsilon}(h, f_1, \ldots, f_T)$. To characterize the performance of head and tail classes in our problem setting, we formally define the loss range of $f_1, \ldots, f_T$ in Definition 2:

Definition 2 (Loss Range)**.** *The loss range of the T predictors $f_1, \ldots, f_T$ is defined as the difference between the lowest and highest values of the loss function across all tasks.*

$$\begin{aligned}\texttt{Range}(f_1, \ldots, f_T) = &\max_t \frac{1}{n_t} \sum_{i=1}^{n_t} l(f_t(h(\mathbf{x}_i^t)), y_i^t) \\ &- \min_t \frac{1}{n_t} \sum_{i=1}^{n_t} l(f_t(h(\mathbf{x}_i^t)), y_i^t),\end{aligned} \tag{2}$$

where $l(\cdot, \cdot)$ is a loss function. For the node classification task, $l(\cdot, \cdot)$ refers to the cross-entropy loss.

In the scenario of long-tail class-membership distribution, there often exists a tension between maintaining head class performance and improving tail class performance [63]. Minimizing the losses of the head classes may lead to a biased model, which increases the losses of the tail classes. Under the premise that the model could keep a good performance on head tasks, we conjecture that controlling the loss range could improve the performance on tail tasks and lead to a better generalization performance of the model. To verify our idea, we drive the loss range-based generalization error bound for long-tail classes on graphs in the following Theorem 1.

Theorem 1 (Generalization Error Bound)**.** *Given the node embedding extraction function $h \in \mathcal{H}$ and the task-specific classifier*

[1]Here we consider the number of tasks to be the number of classes for simplicity, while in Section 3.2 the number of tasks can be smaller than the number of classes after the task grouping operation.

$f_1, \ldots, f_T \in \mathcal{F}$, with probability at least $1-\delta, \delta \in [0,1]$, we have

$$\epsilon - \hat{\epsilon} \leq \sum_t \left(\frac{c_1 \rho R G(\mathcal{H}(\mathbf{X}))}{n_t T} + \sqrt{\frac{9 \ln(2/\delta)}{2 n_t T^2}} + \frac{c_2 \sup_{h \in \mathcal{H}} \|h(\mathbf{X})\| \mathsf{Range}(f_1, \ldots, f_T)}{n_t T} \right), \tag{3}$$

where $\mathbf{X}$ is the node feature, T is the number of tasks, n_t is the number of nodes in task t, R denotes the Lipschitz constant of fuctions in $\mathcal{F}$, loss function $l(\cdot,\cdot)$ is ρ-Lipschitz, $G(\cdot)$ denotes the Gaussian complexity [3], and c_1 and c_2 are universal constants.

Proof. The proof is provided in Appendix B. □

Remark #1: Theorem 1 implies that the generalization error is dominated by three key factors, including the Gaussian complexity of the shared representation extraction $h \in \mathcal{H}$, the loss range of the task-specific predictors $f_1, \ldots, f_T$, the number of classes with varying number of samples.

Remark #2: We can derive $\sum_t \frac{c_1 \rho R G(\mathcal{H}(\mathbf{X}))}{n_t T} \geq \frac{T c_1 \rho R G(\mathcal{H}(\mathbf{X}))}{\sum_t n_t} \geq \frac{c_1 \rho R G(\mathcal{H}(\mathbf{X}))}{\sum_t n_t}$ by utilizing Jensen's Inequality. The observation illustrates that when grouping all samples to one task rather than grouping all samples to T tasks, the first term of the upper bound becomes tight. Our conclusion for long-tail learning is different from multi-task learning in that each task corresponds to a fixed number of observed samples [32]. Conversely, in long-tail learning, task complexity is determined by the number of classes T, each class exhibiting varying numbers of samples $n_1, \ldots, n_T$. Hence, controlling the complexity of the task space could improve the generalization performance, which motivates the design of the hierarchical task grouping module in Section 3.2.

Remark #3: Reducing the loss range $\mathsf{Range}(f_1, \ldots, f_T)$ for all tasks results in a tight third term of the upper bound. This insight inspired the development of long-tail balanced contrastive learning module in Section 3.2, which aims to obtain better task-specific predictors $f'_1, \ldots, f'_T$ with $\mathsf{Range}(f'_1, \ldots, f'_T) < \mathsf{Range}(f_1, \ldots, f_T)$.

3.2 HierTail Framework

The overview of HierTail is presented in Figure 3, which consists of two major modules: M1. hierarchical task grouping and M2. long-tail balanced contrastive learning. Specifically, the Remark #2 of Theorem 1 inspires that controlling the task complexity with massive and imbalanced classes can potentially improve the generalization performance. Thus, M1 is designed to control the complexity of task space and capture the information shared across tasks by grouping tasks into the hypertasks to improve overall performance. As highlighted in Remark #3 above, controlling the loss range could improve the generalization performance. Therefore, in M2, we designed a long-tail balanced contrastive loss to balance the head classes and the tail classes. In the following subsections, we dive into the two modules of HierTail in detail.

M1. Hierarchical Task Grouping. We propose to address C2 (Label scarcity) and C3 (Task complexity) by leveraging the information learned in one class to help train another class. We implement task grouping to share information across different tasks via hierarchical pooling [12, 58], different from previous work which conducts node clustering and ignores the challenges in long-tail learning [24]. The core idea of our hierarchical pooling is to choose the important nodes (tasks) and preserve the original connections between the chosen nodes (tasks) and edges to generate a coarsened graph. As shown in Figure 4, the task grouping operation is composed of two steps: *Step 1.* we group nodes (tasks) into several tasks (hypertasks) and *Step 2.* learn the embeddings of the task (hypertask) prototypes. This operation can be easily generalized to the l^{th} layers, which leads to the hierarchical task grouping.

Specifically, we first generate a low-dimensional node embedding vector for each node $\mathbf{Z}^{(1)} = (\mathbf{z}_1^{(1)}, \ldots, \mathbf{z}_n^{(1)})$ via graph convolutional network (GCN) [23] layers. Next, we group nodes into tasks (with the same number of classes) and then group these tasks into hypertasks by stacking several task grouping layers. The l^{th} task grouping layer is defined as:

$$\begin{aligned} \mathcal{I} &= \texttt{TOP-RANK}(\texttt{PROJ}(\mathbf{Z}^{(l)}), T^{(l)}), \\ \mathbf{X}^{(l+1)} &= \mathbf{Z}^{(l)}(\mathcal{I}, :) \odot \left(\texttt{PROJ}(\mathbf{Z}^{(l)})\mathbf{1}_d^T\right), \\ \mathbf{A}^{(l+1)} &= \mathbf{A}^{(l)}(\mathcal{I}, \mathcal{I}), \end{aligned} \tag{4}$$

where $l = 1, \ldots L$ is the layer of hierarchical task grouping. We generate a new graph with selected important nodes, where these nodes serve as the prototypes of tasks (hypertasks), and $\mathcal{I}$ is the indexes of the selected nodes. $\texttt{PROJ}(\cdot,\cdot)$ is a projection function to score the node importance by mapping each embedding $\mathbf{z}_i^{(l)}$ to a scalar. TOP-RANK identifies the top $T^{(l)}$ nodes with the highest value after projection. The connectivity between the selected nodes remains as edges of the new graph, and the new adjacency matrix $\mathbf{A}^{(l+1)}$ and feature matrix $\mathbf{X}^{(l+1)}$ are constructed by row and/or column extraction. The subsequent GCN layer outputs the embeddings $\mathbf{Z}^{(l+1)}$ of the new graph based on $\mathbf{X}^{(l+1)}$ and $\mathbf{A}^{(l+1)}$. Notably, $\mathbf{Z}^{(1)}$ is the node embeddings, $\mathbf{Z}^{(2)}$ is the embeddings of the task prototypes corresponding to the classes, and $\mathbf{Z}^{(l)}(l > 2)$ is the hypertask prototype embeddings.

The number of tasks $T^{(l)}$ represents the level of abstraction of task grouping, which decreases as the task grouping layer gets deeper. In high-level layers ($l > 1$), the number of tasks may be smaller than the number of classes. By controlling $T^{(l)}$, information shared across tasks can be obtained to alleviate the *task complexity*, which is associated with characterizing an increasing number of classes under varying number of samples. Meanwhile, nodes that come from different classes with high-level semantic similarities can be assigned to one task. By sharing label information with other different classes within the same hypertask, the problem of *label scarcity* can be alleviated. In layer 2 (Figure 4), we consider a special case of 2 head classes (i.e., class 2 and 4) and 2 tail classes (i.e., class 1 and 3). By grouping the prototypes of classes 1, 2, and 3 into the same hypertask at a later task grouping layer, our method will automatically assign a unique hypertask label to all nodes belonging to the three classes.

In order to well capture the hierarchical structure of tasks and propagate information across different tasks, we need to restore the original resolutions of the graph to perform node classification. Specifically, we stack the same number of unpooling layers as the

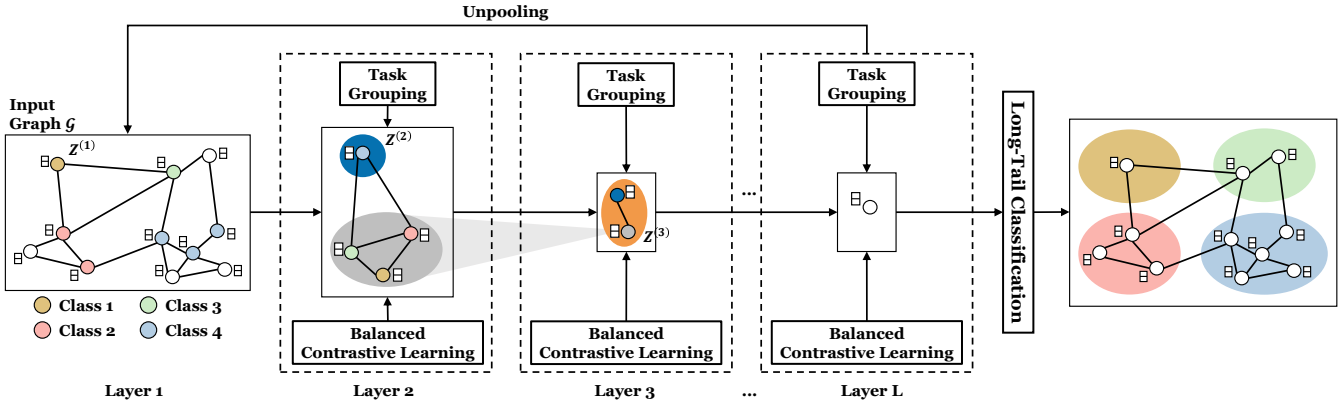

Figure 3: The proposed HierTail framework with L task-grouping layers.

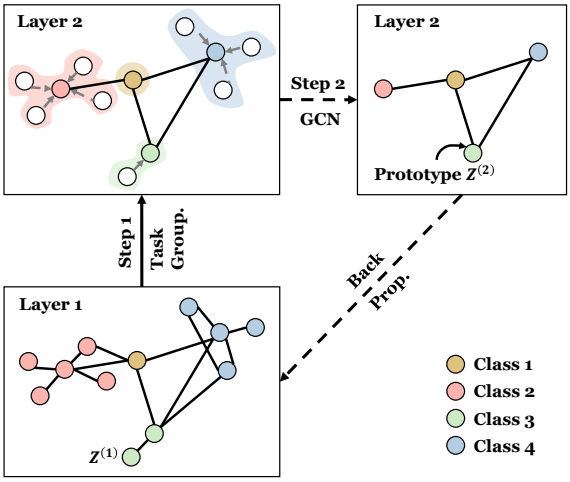

Figure 4: An illustrative figure for M1 with two task-grouping layers. Step 1: nodes are first grouped into four tasks (each representing a class). Step 2: We learn the embeddings of the task prototypes. Finally, the node embeddings are updated by back-propagation.

task grouping layers, which up-samples the features to restore the original resolutions of the graph.

$$\mathbf{X}^{(l+1)} = \texttt{DIST}\left(0_{n\times d}, \mathbf{X}^{(l+1)}, \mathcal{I}\right), \tag{5}$$

where DIST restores the selected graph to the resolution of the original graph by distributing row vectors in $X^{(l+1)}$ into matrix $0_{n\times d}$ based on the indices $\mathcal{I}$, $0_{n\times d}$ represents the initially all-zeros feature matrix, $X^{(l+1)} \in \mathbb{R}^{T^{(l)}\times d}$ represents the feature matrix of the current graph, and $\mathcal{I}$ represents the indices of the selected nodes in the corresponding task grouping layer. Finally, the corresponding blocks of the task grouping and unpooling layers are skip-connected by feature addition, and the final node embeddings are passed to an MLP layer for final predictions.

M2. Long-Tail Balanced Contrastive Learning. To address C1 (High-skewed data distribution) and C2 (Label scarcity), we propose a principled graph contrastive learning strategy for M1 (Hierarchical task grouping) by passing labels across multiple hierarchical layers. Unlike Graph contrastive learning (GCL) [13, 39, 52, 69] for learning unsupervised representation of graph data, in this paper, we propose to incorporate supervision signals into each layer of graph contrastive learning. Specifically, we employ supervised contrastive loss $\mathcal{L}_{SCL}$ on the labeled node to augment the original graph. It allows joint consideration of head and tail classes, which balances their contributions and alleviates the challenge of *high-skewed data distribution*. Additionally, we employ balanced contrastive loss $\mathcal{L}_{BCL}$ on each layer of HierTail. We group all nodes on the graph into several tasks, which facilitates label information to be passed among similar nodes during task grouping. These tasks are subsequently grouped into higher-level hypertasks, which enables label sharing across layers. Through the sharing of label information across nodes and layers, we effectively mitigate the challenge of *label scarcity* in tail classes.

Next, we introduce supervised contrastive loss $\mathcal{L}_{SCL}$ on the restored original graph. It makes node pairs of the same class close to each other while pairs not belonging to the same class far apart. The mathematical form of the loss function $\mathcal{L}_{SCL}$ on the i^{th} node $\mathbf{z}_i$ can be expressed as follows:

$$\mathcal{L}_{SCL}(\mathbf{z}_i) = -\frac{1}{n_t - 1} \times \sum_{j\in\mathcal{V}_t\setminus i} \log \frac{\exp\left(\mathbf{z}_i \cdot \mathbf{z}_j/\tau\right)}{\sum_{1\le q\le T} \frac{1}{n_q} \sum_{k\in\mathcal{V}_q} \exp\left(\mathbf{z}_i \cdot \mathbf{z}_k/\tau\right)}, \tag{6}$$

where $\mathbf{z}_i$ belongs to class t, $\mathcal{V}_t$ denotes all the nodes belonging to class t, z_k represents the embedding of the k^{th} node, and temperature τ controls the strength of penalties on negative node. $\mathcal{L}_{SCL}$ reduces the proportion of contributions from head classes and highlights the importance of tail classes to alleviate the bias caused by high-skewed data distribution.

Moreover, we introduce balanced contrastive loss $\mathcal{L}_{BCL}$ on a coarsened graph, where each node represents a task prototype. For the l^{th} task grouping layer, we group tasks in layer l into $T^{(l)}$ hypertasks and calculate the balanced contrastive loss based on the task embeddings $\mathbf{Z}^{(l)}$ and the hypertask prototypes $\mathbf{Z}^{(l+1)}$. It pulls

the task embeddings together with their corresponding hypertask prototypes and pushes them away from other prototypes. $\mathcal{L}_{BCL}$ on the i^{th} node $\mathbf{z}_i$ can be expressed as follows[2]:

$$\mathcal{L}_{BCL}(\mathbf{z}_i) = -\frac{1}{n_t} \times \sum_{j \in \mathcal{V}_t \setminus i} \log \frac{\exp\left(\mathbf{z}_i \cdot \mathbf{z}_j / \tau\right)}{\sum_{1 \le q \le T} \frac{1}{n_q+1} \sum_{k \in \mathcal{V}_q} \exp\left(\mathbf{z}_i \cdot \mathbf{z}_k / \tau\right)}, \tag{7}$$

where we suppose $\mathbf{z}_i$ belongs to hypertask t, here $\mathcal{V}_t$ denotes all the nodes within the t^{th} hypertask including the hypertask prototype $\mathbf{z}_t^{(l+1)}$, n_t represents the number of nodes in hypertask t, $\mathbf{z}_k = \mathbf{z}_k^{(l)}$ represents the embedding of the k^{th} node, and τ is the temperature. Therefore, $\mathcal{L}_{BCL}$ solves the long-tail classification in two aspects: (1) It potentially controls the range of losses for different tasks. The $n_q + 1$ term in the denominator averages over the nodes of each task so that each task has an approximate contribution for optimizing; (2) The set of T hypertask prototypes is added to obtain a more stable optimization for balanced contrastive learning. In summary, M2 combines supervised contrastive loss and balanced contrastive loss. With M2, we alleviate the label scarcity by passing label information across all nodes and all layers; and solve the data imbalance by balancing the performance of the head and tail classes.

Overall Objective Function. Our objective is to minimize the node classification loss (for few-shot annotated data), the unsupervised balanced contrastive loss (for task combinations in each layer), and the supervised contrastive loss (for categories), which is defined as:

$$\mathcal{L}_{total} = \mathcal{L}_{NC} + \gamma * (\mathcal{L}_{BCL} + \mathcal{L}_{SCL}), \tag{8}$$

where γ balances the contribution of the three terms. The node classification loss $\mathcal{L}_{NC}$ is defined as follows:

$$\mathcal{L}_{NC} = \sum_{i=1}^{T} \mathcal{L}_{CE}\left(g(\mathcal{G}), \mathcal{Y}\right), \tag{9}$$

where $\mathcal{L}_{CE}$ is the cross-entropy loss, $\mathcal{G}$ represents the input graph with few-shot labeled nodes, and $\mathcal{Y}$ represents the labels.

4 Experiments

To evaluate the effectiveness of HierTail for long-tail classification on graphs, we conduct experiments on six benchmark datasets with a large number of classes and data imbalance. Our model exhibits superior performances compared to various state-of-the-art baselines, as detailed in Section 4.2. Further, through ablation studies in Section 4.3, we demonstrate the necessity of each component of HierTail. We also report the parameter and complexity sensitivity, which shows that HierTail achieves a convincing performance with minimal tuning efforts and is scalable, as given in Section 4.4.

4.1 Experiment Setup

Datasets: We evaluate our proposed framework on Cora-Full [4], BlogCatalog [46], Email [57], Wiki [34], Amazon-Clothing [33], and Amazon-Electronics [33] datasets to perform node classification task. The first four datasets naturally have smaller $\texttt{Ratio}_{LT}$, and they are randomly sampled according to train/valid/test ratios = 1:1:8 for each category. While the last two datasets with larger $\texttt{Ratio}_{LT}$, we manually process them to achieve harsh long-tail with $\texttt{Ratio}_{LT} \approx 0.25$. We remove low-degree nodes and their corresponding edges to downsample while maintaining the connections between the remaining nodes. For valid/test sets, we sample 25/55 nodes from each category. To sum up, HierTail is evaluated based on four natural datasets, and two additional datasets with semi-synthetic long-tail settings. The statistics, the original class-imbalance ratio, and the original long-tailedness ratio ($\texttt{Ratio}_{LT}(0.8)$ as defined in Definition 1) of each dataset are summarized in Table 1.

[2]We use the same contrastive loss for each layer. To clarify, we omit layer (l).

Table 1: Dataset statistics.

Dataset	#Nodes	#Edges	#Attributes	#Classes	Imb.	$\texttt{Ratio}_{LT}$
Cora-Full	19,793	146,635	8,710	70	0.016	1.09
BlogCatalog	10,312	333,983	64	38	0.002	0.77
Email	1,005	25,571	128	42	0.009	0.79
Wiki	2,405	25,597	4,973	17	0.022	1.00
Amazon-Clothing	24,919	91,680	9,034	77	0.097	1.23
Amazon-Electronics	42,318	43,556	8,669	167	0.107	1.67

Comparison Baselines: We compare HierTail with five imbalanced classification methods and six GNN-based long-tail classification methods.

- Classical long-tail learning methods: Origin (utilizing a GCN [23] as the encoder and an MLP as the classifier), Over-sampling [6], Re-weighting [59], SMOTE [7], and Embed-SMOTE [2].
- GNN-based long-tail learning methods: GraphSMOTE$_T$ [64], GraphSMOTE$_O$ [64], GraphMixup [51], ImGAGN [40], GraphENS [37], and LTE4G [60].

Implementation Details: We run all the experiments with 10 random seeds and report the evaluation metrics along with standard deviations. Considering the long-tail class-membership distribution, balanced accuracy (bAcc), Macro-F1, and Geometric Means (G-Means) are used as the evaluation metrics, and accuracy (Acc) is used as the traditional metric. For a fair comparison, we use vanilla GCN as backbone and set the hidden layer dimensions of all GCNs in baselines and HierTail to 128 for Cora-Full, Amazon-Clothing, Amazon-Electronics and 64 for BlogCatalog, Email, Wiki. We use Adam [21] optimizer with learning rate 0.01 and weight deacy $5e-4$. The maximum training epoch for all the models is set to 10,000. If there is no additional setting in the original papers, we set the early stop epoch to 1,000, i.e., we force the training to stop if there is no improvement in F1 value on the validation set in 1000 epochs. For baseline methods, we use the same default hyperparameter values as in the original paper. For our model, the weight γ of contrastive loss is set to 0.01, and the temperature τ of contrastive learning is selected in $\{0.01, 0.1, 1.0\}$ for different datasets. We set the depth of the hierarchical graph neural network to 3; node embeddings are calculated for the first layer, the number of tasks is set to the number of categories for the second layer, and the number of tasks is half the number of categories for the third layer. All the experiments are conducted on an A100 SXM4 80GB GPU.

4.2 Performance Analysis

Overall Evaluation. We compare HierTail with eleven methods on six real-world graphs, and the performance of node classification

Table 2: Comparison of different methods in node classification task.

	Method	Cora-Full				BlogCatalog			
		bAcc	Macro-F1	G-Means	Acc	bAcc	Macro-F1	G-Means	Acc
Classical	Origin	52.8 ± 0.6	54.5 ± 0.7	72.5 ± 0.4	62.7 ± 0.5	7.1 ± 0.4	7.3 ± 0.4	26.4 ± 0.7	15.1 ± 1.0
	Over-sampling	52.7 ± 0.7	54.4 ± 0.6	72.4 ± 0.5	62.7 ± 0.4	7.1 ± 0.3	7.2 ± 0.3	26.3 ± 0.6	15.1 ± 1.2
	Re-weight	52.9 ± 0.5	54.4 ± 0.5	72.5 ± 0.3	62.6 ± 0.4	7.2 ± 0.4	7.3 ± 0.5	26.4 ± 0.8	15.1 ± 0.8
	SMOTE	52.7 ± 0.6	54.4 ± 0.5	72.4 ± 0.4	62.7 ± 0.4	7.1 ± 0.4	7.2 ± 0.5	26.3 ± 0.8	15.3 ± 1.2
	Embed-SMOTE	52.9 ± 0.5	54.4 ± 0.5	73.9 ± 0.4	62.6 ± 0.4	7.1 ± 0.5	7.3 ± 0.5	26.3 ± 0.9	14.8 ± 0.8
GNN-based	GraphSMOTE$_T$	54.2 ± 0.8	54.7 ± 0.8	73.4 ± 0.6	62.1 ± 0.6	8.6 ± 0.4	8.5 ± 0.5	28.9 ± 0.7	18.3 ± 1.1
	GraphSMOTE$_O$	54.1 ± 0.8	54.5 ± 0.7	73.3 ± 0.5	62.0 ± 0.6	8.6 ± 0.4	8.5 ± 0.4	28.9 ± 0.6	18.3 ± 0.9
	GraphMixup	53.9 ± 1.3	53.9 ± 1.3	73.2 ± 0.9	61.4 ± 1.2	8.0 ± 0.6	7.9 ± 0.8	27.9 ± 1.2	18.8 ± 0.8
	ImGAGN	9.3 ± 1.1	6.6 ± 1.0	30.2 ± 1.9	20.9 ± 2.1	6.2 ± 0.6	4.9 ± 0.5	24.6 ± 1.3	20.5 ± 1.3
	GraphENS	55.0 ± 0.6	54.2 ± 0.5	73.9 ± 0.4	62.1 ± 0.4	9.0 ± 0.6	8.9 ± 0.5	30.8 ± 0.9	12.8 ± 1.1
	LTE4G	55.8 ± 0.6	54.5 ± 0.4	74.5 ± 0.4	61.6 ± 0.4	6.9 ± 0.5	6.7 ± 0.6	26.0 ± 0.9	11.7 ± 1.3
	Ours	**55.8** ± 0.5	**57.1** ± 0.5	**74.5** ± 0.3	**64.7** ± 0.7	**9.8** ± 0.2	**9.6** ± 0.1	**30.9** ± 0.4	**23.2** ± 0.6
	Method	Email				Wiki			
		bAcc	Macro-F1	G-Means	Acc	bAcc	Macro-F1	G-Means	Acc
Classical	Origin	48.9 ± 4.5	45.2 ± 4.3	69.5 ± 3.2	**66.7** ± 2.1	48.2 ± 1.5	49.9 ± 1.9	68.6 ± 1.1	64.2 ± 0.9
	Over-sampling	48.4 ± 4.2	45.4 ± 3.7	69.2 ± 3.1	66.4 ± 2.0	47.3 ± 2.1	48.7 ± 2.2	67.9 ± 1.5	63.6 ± 1.4
	Re-weight	47.9 ± 4.6	44.2 ± 4.2	68.8 ± 3.4	66.3 ± 1.7	48.1 ± 2.1	49.7 ± 2.5	68.5 ± 1.6	64.0 ± 1.4
	SMOTE	48.4 ± 4.2	45.4 ± 3.7	69.2 ± 3.1	66.4 ± 2.0	47.3 ± 2.1	48.7 ± 2.2	67.9 ± 1.5	63.6 ± 1.4
	Embed-SMOTE	47.9 ± 4.6	44.2 ± 4.2	68.8 ± 3.3	66.2 ± 1.7	48.1 ± 2.1	49.7 ± 2.5	68.5 ± 1.6	63.9 ± 1.4
GNN-based	GraphSMOTE$_T$	43.4 ± 2.9	39.1 ± 2.8	65.5 ± 2.2	60.4 ± 1.5	50.3 ± 1.7	51.8 ± 2.2	70.1 ± 1.2	65.8 ± 0.9
	GraphSMOTE$_O$	42.3 ± 3.1	38.3 ± 2.9	64.7 ± 2.4	60.1 ± 2.3	49.6 ± 2.3	51.1 ± 2.7	69.6 ± 1.7	65.5 ± 1.2
	GraphMixup	43.2 ± 2.3	38.1 ± 2.3	65.4 ± 1.7	60.1 ± 1.7	50.3 ± 2.9	51.2 ± 2.9	70.0 ± 2.1	65.1 ± 1.3
	ImGAGN	27.6 ± 3.4	26.8 ± 2.9	52.0 ± 3.2	46.5 ± 3.5	41.2 ± 5.7	42.3 ± 6.4	63.2 ± 4.9	65.5 ± 5.8
	GraphENS	50.5 ± 3.1	43.7 ± 3.3	**71.1** ± 2.2	62.0 ± 2.7	50.8 ± 3.3	50.1 ± 3.4	70.3 ± 2.4	61.7 ± 4.4
	LTE4G	46.4 ± 2.5	39.3 ± 2.4	67.8 ± 1.8	57.8 ± 3.1	51.0 ± 2.9	49.7 ± 1.9	70.5 ± 2.1	60.4 ± 2.1
	Ours	**50.5** ± 3.0	**46.6** ± 3.0	70.7 ± 2.1	65.4 ± 1.7	**52.8** ± 2.0	**54.1** ± 2.3	**71.9** ± 1.4	**67.2** ± 1.1

Table 3: Comparison of different methods in node classification task on semi-synthetic long-tail datasets with long-tailedness ratio $\texttt{Ratio}_{LT}(0.8) \approx 0.25$.

	Method	Amazon-Clothing				Amazon-Electronics			
		bAcc	Macro-F1	G-Means	Acc	bAcc	Macro-F1	G-Means	Acc
Classical	Origin	9.9 ± 0.2	9.5 ± 0.2	31.3 ± 0.3	9.9 ± 0.2	16.9 ± 0.2	15.2 ± 0.2	41.0 ± 0.3	16.9 ± 0.2
	Over-sampling	9.9 ± 0.2	9.5 ± 0.2	31.3 ± 0.3	9.9 ± 0.2	16.8 ± 0.1	15.1 ± 0.1	40.9 ± 0.2	16.8 ± 0.1
	Re-weight	10.0 ± 0.2	9.6 ± 0.2	31.4 ± 0.3	10.0 ± 0.2	17.0 ± 0.2	15.2 ± 0.2	41.1 ± 0.3	17.0 ± 0.2
	SMOTE	10.0 ± 0.1	9.5 ± 0.2	31.4 ± 0.2	10.0 ± 0.1	16.9 ± 0.2	15.1 ± 0.2	41.0 ± 0.3	16.9 ± 0.2
	Embed-SMOTE	9.9 ± 0.2	9.5 ± 0.2	31.3 ± 0.3	9.9 ± 0.2	17.0 ± 0.2	15.2 ± 0.2	41.1 ± 0.3	17.0 ± 0.2
GNN-based	GraphSMOTE$_T$	11.7 ± 0.2	10.4 ± 0.3	34.0 ± 0.3	11.7 ± 0.2	18.2 ± 0.2	15.6 ± 0.2	42.5 ± 0.2	18.2 ± 0.2
	GraphSMOTE$_O$	11.7 ± 0.2	10.4 ± 0.3	34.0 ± 0.3	11.7 ± 0.2	18.2 ± 0.2	15.5 ± 0.2	42.5 ± 0.2	18.2 ± 0.2
	GraphMixup	10.9 ± 0.5	9.3 ± 0.7	32.8 ± 0.7	10.9 ± 0.5	18.1 ± 0.4	15.5 ± 0.5	42.5 ± 0.5	18.1 ± 0.4
	ImGAGN	12.9 ± 0.2	9.2 ± 0.1	35.7 ± 0.2	12.9 ± 0.2	13.7 ± 0.2	11.0 ± 0.0	36.9 ± 0.2	13.7 ± 0.2
	GraphENS	11.6 ± 2.7	10.9 ± 2.7	33.6 ± 4.3	11.6 ± 2.7	19.2 ± 3.8	17.2 ± 3.6	43.5 ± 4.4	19.2 ± 3.8
	LTE4G	15.5 ± 0.3	16.0 ± 0.5	39.1 ± 0.3	15.5 ± 0.3	20.9 ± 0.3	19.9 ± 0.3	45.7 ± 0.3	20.9 ± 0.3
	Ours	**17.1** ± 0.5	**16.8** ± 0.6	**41.1** ± 0.6	**17.1** ± 0.5	**23.6** ± 0.9	**21.0** ± 1.3	**48.5** ± 1.0	**23.6** ± 0.9

is reported in Table 2 and Table 3. In general, we have the following observations: (1) HierTail consistently performs well on all datasets under various long-tail settings and especially outperforms other baselines on harsh long-tail settings (e.g., $\texttt{Ratio}_{LT}(0.8) \approx 0.25$), which demonstrates the effectiveness and generalizability of our model. More precisely, taking the Amazon-Electronics dataset (which has 167 classes and follows the Pareto distribution with "80-20 Rule") as an example, the improvement of our model on bAcc (Acc) is 12.9% compared to the second best model (LTE4G). It implies that HierTail can not only solve the highly skewed data but also capture a massive number of classes. (2) Classical long-tail learning methods have the worst performance because they ignore graph

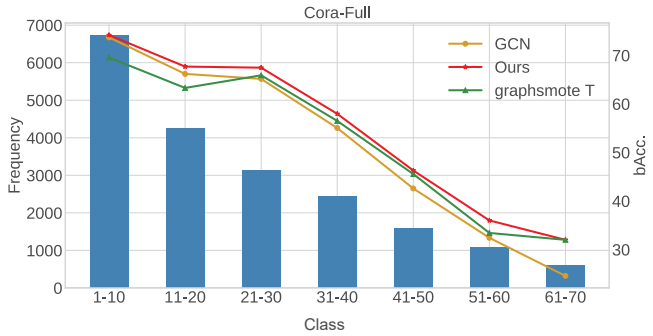

Figure 5: Performance on groups of ten classes in Cora-Full, where the yellow, red and green curves show bAcc (%) of GCN, HierTail and GraphSMOTE_T for node classification.

structure information and only conduct oversampling or reweighting in the feature space. HierTail improves bAcc up to 36.1% on the natural dataset (BlogCatalog) and 71.0% on the manually processed dataset (Amazon-Clothing) compared to the classical long-tail learning methods. (3) GNN-based long-tail learning methods achieve the second-best performance (excluding the Email dataset), which implies that it is beneficial to capture or transfer knowledge on the graph topology, but these models ignore the massive number of classes. In particular, since ImGAGN only considers the high-skewed distribution, as the number of classes increases (from Wiki to Cora-Full), the model becomes less effective. Our model outperforms these GNN-based methods on almost all the natural datasets and metrics (excluding Email), such as up to 12.9% improvement on the manually processed dataset (Amazon-Electronics).

Performance on Each Class. To observe the performance of our model for the long-tail classification, we plot the model performance (bAcc) on each class in Figure 1 and for groups of ten classes in Figure 5. We find that HierTail outperforms the original GCN method (which fails to consider the long-tail class-membership distribution), especially on the tail classes.

4.3 Ablation Study

Table 4 presents the node classification performance on Cora-Full when considering (a) complete HierTail; (b) hierarchical task grouping, balanced contrastive loss, and node classification loss; (c) hierarchical task grouping, supervised contrastive loss, and node classification loss; (d) hierarchical task grouping and node classification loss; and (e) only node classification loss. From the results, we have several interesting observations: (1) Hierarchical task grouping (M1) helps the model better share information across tasks, which achieves impressive improvement on Cora-Full by up to 3.2% ((d) > (e)). (2) Long-tail balanced contrastive learning module (M2) leads to an increase in bAcc by 2.4%, which shows its strength in improving long-tail classification by ensuring accurate node embeddings ((a) > (d)). (3) Supervised contrastive learning leads to an improvement from 54.3 to 55.8 in bAcc ((a) > (b)). (4) Balanced contrastive learning leads to an improvement from 54.6 to 55.8 in bAcc ((a) > (c)). Overall, the ablation study firmly attests both modules are essential in successful long-tail classification on graphs.

Table 4: Ablation study on each component of HierTail.

Components				Cora-Full			
M1	M2		$\mathcal{L}_{CE}$	bAcc	Macro-F1	G-Means	Acc
	$\mathcal{L}_{BCL}$	$\mathcal{L}_{SCL}$					
✓	✓	✓	✓	**55.8** ± 0.5	**57.1** ± 0.5	**74.5** ± 0.3	**64.7** ± 0.7
✓	✓		✓	54.3 ± 0.7	56.1 ± 0.7	73.5 ± 0.5	64.4 ± 0.4
✓		✓	✓	54.6 ± 0.4	56.2 ± 0.4	73.7 ± 0.3	64.3 ± 0.4
✓			✓	54.5 ± 0.5	56.2 ± 0.4	73.6 ± 0.3	64.5 ± 0.4
			✓	52.8 ± 0.6	54.5 ± 0.7	72.5 ± 0.4	62.7 ± 0.5

Table 5: Hyperparameter analysis on Cora-Full with respect to the number of tasks in the second and third layers.

	Cora-Full			
	bAcc	Macro-F1	G-Means	Acc
[198, 70]	55.5	56.7	74.2	64.6
[70, 35]	55.8	57.1	74.5	64.7
[2, 1]	54.9	56.8	73.9	65.5

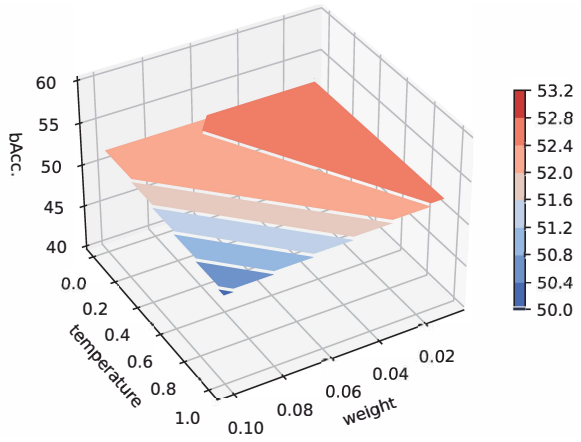

Figure 6: Hyperparameter analysis on Cora-Full with respect to weight γ and temperature τ.

4.4 Parameter and Complexity Analysis

Hyperparameter Analysis. We configure the number of tasks in the second layer to align with the number of classes in Section 3.2. To investigate the potential effects of overclustering [17, 20] where the number of clusters is larger than the number of classes, we conduct experiments by adjusting the number of tasks in the second and third layers. Table 5 illustrates the impact of varying the number of tasks on model performance. The experimental results reveal that our model achieves great performance within a certain reasonable range of hyperparameters. However, there is a slight performance degradation when the number of hypertasks is small.

In addition, we study the following hyperparameters: (1) the weight γ to balance the contribution of three losses and (2) the temperature τ of balanced contrastive loss in M2. As shown in Figure 6, the fluctuation of the bAcc (z-axis) is less than 5%. The bAcc is slightly lower when both weight γ and temperature τ become larger. Overall, we find HierTail is reliable and not sensitive to the hyperparameters under a wide range.

Complexity Analysis. We report the running time and memory usage of HierTail, GCN, and LTE4G (a efficient state-of-the-art

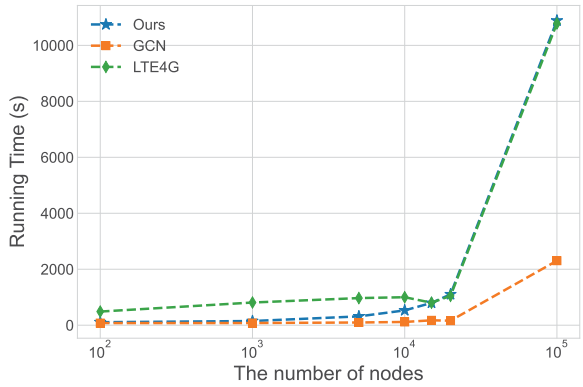

Figure 7: Time complexity analysis w.r.t. the number of nodes.

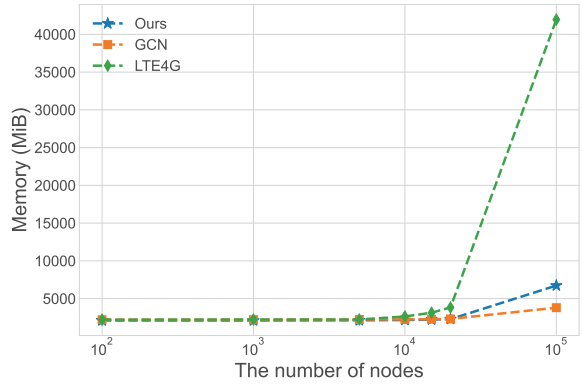

Figure 8: Space omplexity analysis w.r.t. the number of nodes.

method). For better visualize the performance, we run the experiment on an increasing graph size, i.e., from 100 to 100,000 nodes. As depicted in Figure 7, our approach HierTail consistently exhibits superior or similar running time compared to the LTE4G method. Although our method has slightly higher running time than GCN, the gap between our approach and GCN remains modest especially when for graph sizes smaller than 10^4. The relationship between the running time of our model and the number of nodes is similarly linear. The best space complexity of our method can reach $O(nd + d^2 + |\mathcal{E}|)$, which is linear in the number of nodes and the number of edges. From the memory usage given in Figure 8, it is shown that HierTail exhibits significantly superior memory usage compared to LTE4G and closely approximates the memory usage of GCN. The results illustrate the scalability of our method.

5 Related Work

Long-tail Problems. Long-tail data distributions are common in real-world applications [63]. Several methods are proposed to solve the long-tail problem, such as data augmentation methods [6, 27] and cost-sensitive methods [9, 59, 68]. However, the vast majority of previous efforts focus on independent and identically distributed (i.i.d.) data, which cannot be directly applied to graph data. Recently, several related works for long-tail classification on graphs [15, 28, 30, 37, 40, 49, 51, 56, 60, 61] have attracted attention. Despite this, the long-tail approaches often lack a theoretical basis. The most relevant work lies in imbalanced classification. Cao et al. [5] and Kini et al. [22] present model-related bounds on the error and the SVM margins, while Yang and Xu [55] provide the error bound of a linear classifier on data distribution and dimension. In addition, previous long-tail work is performed under the class imbalance settings where the number of classes can be small, and the number of minority nodes may not be small; but for long-tail learning, the number of classes is large, and the tail nodes are scarce. In this paper, we provide a theoretical analysis of the long-tail problem and conduct experiments on long-tail datasets.

Graph Neural Networks. Graph neural networks emerge as state-of-the-art methods for graph representation learning, which capture the structure of graphs. Recently, several attempts have been focused on extending pooling operations to graphs. In order to achieve an overview of the graph structure, hierarchical pooling [12, 25, 29, 41, 58] techniques attempt to gradually group nodes into clusters and coarsen the graph recursively. Gao and Ji [12] propose an encoder-decoder architecture based on gPool and gUnpool layers. However, these approaches are generally designed to enhance the representation of the whole graph. In this paper, we aim to explore node classification with the long-tail class-membership distribution via hierarchical pooling methods.

6 Conclusion

In this paper, we investigate long-tail classification on graphs, which intends to improve the performance on both head and tail classes. By formulating this problem in the fashion of multi-task learning, we propose the generalization bound dominated by the range of losses across all tasks and the task complexity. Building upon the theoretical findings, we also present HierTail. It is a generic framework with two major modules: M1. Hierarchical task grouping to control the complexity of the task space and address C2 (Label scarcity) and C3 (Task complexity); and M2. Long-tail balanced contrastive learning to control the range of losses across all tasks and solve C1 (High-skewed data distribution) and C2 (Label scarcity). Extensive experiments on six real-world datasets, where HierTail consistently outperforms state-of-art baselines, demonstrate the efficacy of our model for capturing long-tail classes on graphs.

Reproducibility: Our code and data are released at https://github.com/wanghh7/HierTail.

Acknowledgements

We thank the anonymous reviewers for their constructive comments. This work is supported by 4-VA, Cisco, Commonwealth Cyber Initiative, DARPA under the contract No. HR00112490370, Deloitte & Touche LLP, DHS CINA, the National Science Foundation under Award No. IIS-2339989, and Virginia Tech. The views and conclusions are those of the authors and should not be interpreted as representing the official policies of the funding agencies or the government.

References

[1] Leman Akoglu, Hanghang Tong, and Danai Koutra. 2015. Graph based anomaly detection and description: a survey. *Data mining and knowledge discovery* 29, 3 (2015), 626–688.

[2] Shin Ando and Chun Yuan Huang. 2017. Deep over-sampling framework for classifying imbalanced data. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 770–785.

[3] Peter L Bartlett and Shahar Mendelson. 2002. Rademacher and Gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research* 3, Nov (2002), 463–482.

[4] Aleksandar Bojchevski and Stephan Günnemann. 2018. Deep Gaussian Embedding of Graphs: Unsupervised Inductive Learning via Ranking. In *International Conference on Learning Representations (ICLR'18)*.

[5] Kaidi Cao, Colin Wei, Adrien Gaidon, Nikos Arechiga, and Tengyu Ma. 2019. Learning imbalanced datasets with label-distribution-aware margin loss. In *Advances in Neural Information Processing Systems (NeurIPS'19, Vol. 32)*.

[6] Nitesh V Chawla. 2003. C4. 5 and imbalanced data sets: investigating the effect of sampling method, probabilistic estimate, and decision tree structure. In *International Conference on Machine Learning (ICML'03, Vol. 3)*. PMLR, 66.

[7] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. 2002. SMOTE: synthetic minority over-sampling technique. *Journal of artificial intelligence research* 16 (2002), 321–357.

[8] Yingtong Dou, Zhiwei Liu, Li Sun, Yutong Deng, Hao Peng, and Philip S. Yu. 2020. Enhancing Graph Neural Network-Based Fraud Detectors against Camouflaged Fraudsters. In *International Conference on Information and Knowledge Management (CIKM'20)*. ACM, 315–324.

[9] Charles Elkan. 2001. The foundations of cost-sensitive learning. In *International joint conference on artificial intelligence (IJCAI'01, Vol. 17)*. Lawrence Erlbaum Associates Ltd, 973–978.

[10] Wenqi Fan, Yao Ma, Qing Li, Yuan He, Eric Zhao, Jiliang Tang, and Dawei Yin. 2019. Graph Neural Networks for Social Recommendation. In *International World Wide Web Conference (WWW'19)*. ACM, 417–426.

[11] Shengyu Feng, Baoyu Jing, Yada Zhu, and Hanghang Tong. 2022. Adversarial graph contrastive learning with information regularization. In *International World Wide Web Conference (WWW'22)*. 1362–1371.

[12] Hongyang Gao and Shuiwang Ji. 2019. Graph U-Nets. In *International Conference on Machine Learning (ICML'19)*. PMLR, 2083–2092.

[13] Kaveh Hassani and Amir Hosein Khas Ahmadi. 2020. Contrastive Multi-View Representation Learning on Graphs. In *International Conference on Machine Learning (ICML'20, Vol. 119)*. PMLR, 4116–4126.

[14] Marti A. Hearst, Susan T Dumais, Edgar Osuna, John Platt, and Bernhard Scholkopf. 1998. Support vector machines. *IEEE Intelligent Systems and their applications* 13, 4 (1998), 18–28.

[15] Fenyu Hu, Liping Wang, Qiang Liu, Shu Wu, Liang Wang, and Tieniu Tan. 2022. GraphDIVE: Graph Classification by Mixture of Diverse Experts. In *International Joint Conference on Artificial Intelligence (IJCAI'22)*. 2080–2086.

[16] Weihua Hu, Bowen Liu, Joseph Gomes, Marinka Zitnik, Percy Liang, Vijay S. Pande, and Jure Leskovec. 2020. Strategies for Pre-training Graph Neural Networks. In *International Conference on Learning Representations (ICLR'20)*.

[17] Xu Ji, Joao F Henriques, and Andrea Vedaldi. 2019. Invariant information clustering for unsupervised image classification and segmentation. In *IEEE/CVF international conference on computer vision (ICCV'19)*. IEEE, 9865–9874.

[18] Baoyu Jing, Chanyoung Park, and Hanghang Tong. 2021. Hdmi: High-order deep multiplex infomax. In *International World Wide Web Conference (WWW'21)*. 2414–2424.

[19] Jongmin Kim, Taesup Kim, Sungwoong Kim, and Chang D. Yoo. 2019. Edge-Labeling Graph Neural Network for Few-Shot Learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR'19)*. IEEE, 11–20.

[20] Yunji Kim and Jung-Woo Ha. 2022. Contrastive Fine-grained Class Clustering via Generative Adversarial Networks. In *International Conference on Learning Representations (ICLR'22)*.

[21] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *International Conference on Learning Representations (ICLR'15)*.

[22] Ganesh Ramachandra Kini, Orestis Paraskevas, Samet Oymak, and Christos Thrampoulidis. 2021. Label-imbalanced and group-sensitive classification under overparameterization. *Advances in Neural Information Processing Systems* 34 (2021), 18970–18983.

[23] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *International Conference on Learning Representations (ICLR'17)*.

[24] Sung Moon Ko, Sungjun Cho, Dae-Woong Jeong, Sehui Han, Moontae Lee, and Honglak Lee. 2023. Grouping Matrix Based Graph Pooling with Adaptive Number of Clusters. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI'23)*. AAAI Press, 8334–8342.

[25] Junhyun Lee, Inyeop Lee, and Jaewoo Kang. 2019. Self-Attention Graph Pooling. In *International Conference on Machine Learning (ICML'19, Vol. 97)*. PMLR, 3734–3743.

[26] Dongyue Li, Haotian Ju, Aneesh Sharma, and Hongyang R Zhang. 2023. Boosting multitask learning on graphs through higher-order task affinities. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD'23)*. ACM, 1213–1222.

[27] Xu-Ying Liu, Jianxin Wu, and Zhi-Hua Zhou. 2008. Exploratory undersampling for class-imbalance learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 39, 2 (2008), 539–550.

[28] Zemin Liu, Qiheng Mao, Chenghao Liu, Yuan Fang, and Jianling Sun. 2022. On Size-Oriented Long-Tailed Graph Classification of Graph Neural Networks. In *International World Wide Web Conference (WWW'22)*. ACM, 1506–1516.

[29] Yao Ma, Suhang Wang, Charu C. Aggarwal, and Jiliang Tang. 2019. Graph Convolutional Networks with EigenPooling. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD'19)*. ACM, 723–731.

[30] Zhengyang Mao, Wei Ju, Yifang Qin, Xiao Luo, and Ming Zhang. 2023. RAHNet: Retrieval Augmented Hybrid Network for Long-tailed Graph Classification. In *ACM International Conference on Multimedia (MM'23)*. ACM, 3817–3826.

[31] Andreas Maurer. 2016. A chain rule for the expected suprema of Gaussian processes. *Theoretical Computer Science* 650 (2016), 109–122.

[32] Andreas Maurer, Massimiliano Pontil, and Bernardino Romera-Paredes. 2016. The Benefit of Multitask Representation Learning. *Journal of Machine Learning Research* 17, 1 (2016), 2853–2884.

[33] Julian McAuley, Rahul Pandey, and Jure Leskovec. 2015. Inferring networks of substitutable and complementary products. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD'15)*. ACM, 785–794.

[34] Péter Mernyei and Cătălina Cangea. 2020. Wiki-CS: A Wikipedia-Based Benchmark for Graph Neural Networks. *arXiv preprint arXiv:2007.02901* (2020).

[35] Anshul Mittal, Kunal Dahiya, Sheshansh Agrawal, Deepak Saini, Sumeet Agarwal, Purushottam Kar, and Manik Varma. 2021. Decaf: Deep extreme classification with label features. In *International Conference on Web Search and Data Mining (WSDM'21)*. ACM, 49–57.

[36] Vilfredo Pareto, Ann Stranquist Schwier, and Alfred Nye Page. 1971. Manual of political economy. (1971).

[37] Joonhyung Park, Jaeyun Song, and Eunho Yang. 2022. GraphENS: Neighbor-Aware Ego Network Synthesis for Class-Imbalanced Node Classification. In *International Conference on Learning Representations (ICLR'22)*.

[38] Dan Pelleg and Andrew Moore. 2004. Active learning for anomaly and rare-category detection. In *Advances in Neural Information Processing Systems (NeurIPS'04)*. 1073–1080.

[39] Jiezhong Qiu, Qibin Chen, Yuxiao Dong, Jing Zhang, Hongxia Yang, Ming Ding, Kuansan Wang, and Jie Tang. 2020. Gcc: Graph contrastive coding for graph neural network pre-training. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD'20)*. ACM, 1150–1160.

[40] Liang Qu, Huaisheng Zhu, Ruiqi Zheng, Yuhui Shi, and Hongzhi Yin. 2021. ImGAGN: Imbalanced network embedding via generative adversarial graph networks. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD'21)*. ACM, 1390–1398.

[41] Ekagra Ranjan, Soumya Sanyal, and Partha Talukdar. 2020. ASAP: Adaptive Structure Aware Pooling for Learning Hierarchical Graph Representations. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI'20, Vol. 34)*. AAAI Press, 5470–5477.

[42] Victor Garcia Satorras and Joan Bruna Estrach. 2018. Few-Shot Learning with Graph Neural Networks. In *International Conference on Learning Representations (ICLR'18)*.

[43] Min Shi, Yufei Tang, Xingquan Zhu, David Wilson, and Jianxun Liu. 2020. Multi-Class Imbalanced Graph Convolutional Network Learning. In *International Joint Conference on Artificial Intelligence (IJCAI'20)*. 2879–2885.

[44] Tommie W Singleton and Aaron J Singleton. 2010. *Fraud auditing and forensic accounting*. Vol. 11. John Wiley and Sons.

[45] Xiaozhuang Song, Shun Zheng, Wei Cao, James Yu, and Jiang Bian. 2022. Efficient and effective multi-task grouping via meta learning on task combinations. In *Advances in Neural Information Processing Systems (NeurIPS'22)*.

[46] Lei Tang and Huan Liu. 2009. Relational Learning via Latent Social Dimensions. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD'09)*. ACM, 817–826.

[47] Daixin Wang, Jianbin Lin, Peng Cui, Quanhui Jia, Zhen Wang, Yanming Fang, Quan Yu, Jun Zhou, Shuang Yang, and Yuan Qi. 2019. A semi-supervised graph attentive network for financial fraud detection. In *International Conference on Data Mining (ICDM'19)*. IEEE, 598–607.

[48] Yiwei Wang, Wei Wang, Yuxuan Liang, Yujun Cai, and Bryan Hooi. 2021. Mixup for node and graph classification. In *International World Wide Web Conference (WWW'21)*. ACM, 3663–3674.

[49] Yu Wang, Yuying Zhao, Neil Shah, and Tyler Derr. 2022. Imbalanced Graph Classification via Graph-of-Graph Neural Networks. In *International Conference on Information and Knowledge Management (CIKM'22)*. ACM, 2067–2076.

[50] Longfeng Wu, Bowen Lei, Dongkuan Xu, and Dawei Zhou. 2023. Towards Reliable Rare Category Analysis on Graphs via Individual Calibration. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD'23)*. ACM, 2629–2638.

[51] Lirong Wu, Jun Xia, Zhangyang Gao, Haitao Lin, Cheng Tan, and Stan Z Li. 2023. GraphMixup: Improving class-imbalanced node classification by reinforcement mixup and self-supervised context prediction. In *Machine Learning and Knowledge Discovery in Databases*. Springer, 519–535.

[52] Dongkuan Xu, Wei Cheng, Dongsheng Luo, Haifeng Chen, and Xiang Zhang. 2021. InfoGCL: Information-Aware Graph Contrastive Learning. In *Advances in Neural Information Processing Systems (NeurIPS'21, Vol. 34)*. 30414–30425.

[53] Ling Yang, Liangliang Li, Zilun Zhang, Xinyu Zhou, Erjin Zhou, and Yu Liu. 2020. DPGN: Distribution Propagation Graph Network for Few-Shot Learning.

In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR'20)*. IEEE.
[54] Xiaocheng Yang, Mingyu Yan, Shirui Pan, Xiaochun Ye, and Dongrui Fan. 2023. Simple and Efficient Heterogeneous Graph Neural Network. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI'23, Vol. 37)*. AAAI Press, 10816–10824.
[55] Yuzhe Yang and Zhi Xu. 2020. Rethinking the value of labels for improving class-imbalanced learning. In *Advances in Neural Information Processing Systems (NeurIPS'20, Vol. 33)*. 19290–19301.
[56] Si-Yu Yi, Zhengyang Mao, Wei Ju, Yong-Dao Zhou, Luchen Liu, Xiao Luo, and Ming Zhang. 2023. Towards long-tailed recognition for graph classification via collaborative experts. *IEEE Transactions on Big Data* (2023).
[57] Hao Yin, Austin R. Benson, Jure Leskovec, and David F. Gleich. 2017. Local Higher-Order Graph Clustering. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD'17)*. ACM, 555–564.
[58] Rex Ying, Jiaxuan You, Christopher Morris, Xiang Ren, William L. Hamilton, and Jure Leskovec. 2018. Hierarchical Graph Representation Learning with Differentiable Pooling. In *Advances in Neural Information Processing Systems (NeurIPS'18)*. Curran Associates Inc., 4805–4815.
[59] Bo Yuan and Xiaoli Ma. 2012. Sampling + reweighting: Boosting the performance of AdaBoost on imbalanced datasets. In *International Joint Conference on Neural Networks*. IEEE, 1–6.
[60] Sukwon Yun, Kibum Kim, Kanghoon Yoon, and Chanyoung Park. 2022. LTE4G: Long-Tail Experts for Graph Neural Networks. In *International Conference on Information and Knowledge Management (CIKM'22)*. ACM, 2434–2443.
[61] Chunhui Zhang, Chao Huang, Yijun Tian, Qianlong Wen, Zhongyu Ouyang, Youhuan Li, Yanfang Ye, and Chuxu Zhang. 2023. When Sparsity Meets Contrastive Models: Less Graph Data Can Bring Better Class-Balanced Representations. In *International Conference on Machine Learning (ICML'23, Vol. 202)*. PMLR, 41133–41150.
[62] Min-Ling Zhang and Zhi-Hua Zhou. 2013. A review on multi-label learning algorithms. *IEEE transactions on knowledge and data engineering* 26, 8 (2013), 1819–1837.
[63] Yifan Zhang, Bingyi Kang, Bryan Hooi, Shuicheng Yan, and Jiashi Feng. 2023. Deep long-tailed learning: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45, 9 (2023), 10795–10816.
[64] Tianxiang Zhao, Xiang Zhang, and Suhang Wang. 2021. GraphSMOTE: Imbalanced Node Classification on Graphs with Graph Neural Networks. In *International Conference on Web Search and Data Mining (WSDM'21)*. ACM, 833–841.
[65] Baojian Zhou, Yifan Sun, and Reza Babanezhad Harikandeh. 2023. Fast Online Node Labeling for Very Large Graphs. In *International Conference on Machine Learning (ICML'23, Vol. 202)*. PMLR, 42658–42697.
[66] Dawei Zhou and Jingrui He. 2024. Rare Category Analysis for Complex Data: A Review. *Comput. Surveys* 56, 5 (2024), 123:1–123:35.
[67] 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 *International Conference on Information and Knowledge Management (CIKM'19)*. ACM, 2357–2360.
[68] Zhi-Hua Zhou and Xu-Ying Liu. 2005. Training cost-sensitive neural networks with methods addressing the class imbalance problem. *IEEE Transactions on Knowledge and Data Engineering* 18, 1 (2005), 63–77.
[69] Yanqiao Zhu, Yichen Xu, Qiang Liu, and Shu Wu. 2021. An Empirical Study of Graph Contrastive Learning. In *Neural Information Processing Systems Track on Datasets and Benchmarks*.

A Details of $\texttt{Ratio}_{LT}(p)$

To better characterize class-membership skewness and number of classes, we introduce a novel quantile-based metric named long-tailedness ratio for the long-tail datasets.

$$\texttt{Ratio}_{LT}(p) = \frac{Q(p)}{T - Q(p)},$$

where $Q(p) = min\{y : Pr(\mathcal{Y} \leq y) = p, 1 \leq y \leq T\}$ is the quantile function of order $p \in (0, 1)$ for variable $\mathcal{Y}$, T is the number of categories. The numerator represents the number of categories to which p percent instances belong, and the denominator represents the number of categories to which the else $(1-p)$ percent instances belong in $\mathcal{D}$.

The hyperparameter p allows end users to control the number of classes in the head of the long-tail distribution. If there is no specific definition of the head class in certain domains, we suggest simply following the Pareto principle ($p = 0.8$). Using the same p value for two long-tail datasets allows us to compare the complexity. Otherwise, if the $\texttt{Ratio}_{LT}(p)$ of two datasets are measured based on different p values, they are not comparable. If there is a specific definition of the head class in certain domains, we can directly calculate the number of head classes and thus infer the p value.

In addition, in light of class-imbalance ratio and long-tailedness ratio, we gain a better understanding of the datasets and methods to use. (1) High class-imbalance ratio and low $\texttt{Ratio}_{LT}$ imply high-skewed data distribution, and we may encounter a large number of categories. In such situations, a long-tail method that is designed for data imbalance and an extreme number of classes may be necessary to achieve optimal results. (2) High class-imbalance ratio and high $\texttt{Ratio}_{LT}$ suggest that the task complexity is low with a relatively small number of categories and the dataset may be imbalanced. Therefore, imbalanced classification approaches such as re-sampling or re-weighting may be effective. (3) Low class-imbalance ratio and low $\texttt{Ratio}_{LT}$ imply high task complexity but relatively balanced samples. In such cases, extreme classification methods would be preferred. (4) Low class-imbalance ratio and high $\texttt{Ratio}_{LT}$ suggest that the dataset may not follow a long-tail distribution, and ordinary machine learning methods may achieve great performance.

B Details of Theoretical Analysis

We obtain the range-based generalization error bound for long-tail categories in the following steps: (S1) giving the loss-related generalization error bound based on the Gaussian complexity-based bound in Lemma 1; (S2) deriving the generalization error bound (Theorem 1) related to representation extraction h and the range of task-specific predictors $f_1, \ldots, f_T$ based on the loss-related error bound in S1, the property of Gaussian complexity in Lemma 2, and the chain rule of Gaussian complexity in Lemma 3.

First, we have the following assumptions from the previous work [32].

Assumption 1 (R-Lipschitz Function). *Assume each function $f \in \mathcal{F}$ is R-Lipschitz in ℓ_2 norm, i.e., $\forall \mathbf{x}, \mathbf{x}' \in \mathcal{X}$,*

$$\left|f(\mathbf{x}) - f\left(\mathbf{x}'\right)\right| \leq R\left\|\mathbf{x} - \mathbf{x}'\right\|_2.$$

Assumption 2 (ρ-Lipschitz Loss). *Assume the loss function $l(\cdot, \cdot)$ is ρ-Lipschitz if $\exists\ \rho > 0$ such that $\forall \mathbf{x} \in \mathcal{X}$, $y, y' \in \mathcal{Y}$ and $f, f' \in \mathcal{H}$, the following inequalities hold:*

$$\left|l\left(f'(\mathbf{x}), y\right) - l(f(\mathbf{x}), y)\right| \leq \rho\left|f'(\mathbf{x}) - f(\mathbf{x})\right|,$$
$$\left|l\left(f(\mathbf{x}), y'\right) - l(f(\mathbf{x}), y)\right| \leq \rho\left|y' - y\right|.$$

Based on Maurer et al. [32], we can derive the Gaussian complexity-based bound on the training set $\mathbf{X}$ as follows (S1).

Lemma 1 (Gaussian Complexity-Based Bound). *Let $\mathcal{F}$ be a class of functions $f : \mathbf{X} \to [0, 1]^T$, and $\mathbf{x}_i^t$ represents i^{th} instances belonging to class t. Then, with probability greater than $1 - \delta$ and for all $f \in \mathcal{F}$, we have the following bound*

$$\begin{aligned}&\frac{1}{T}\sum_t\left(\mathbb{E}_{\mathbf{X}\sim\mu_t}\left[f_t(\mathbf{X})\right] - \sum_i \frac{1}{n_t} f_t\left(\mathbf{x}_i^t\right)\right)\\ &\leq \sum_t\left(\frac{\sqrt{2\pi}G(\mathbf{Z})}{n_t T} + \sqrt{\frac{9\ln(2/\delta)}{2n_t T^2}}\right),\end{aligned} \tag{10}$$

where $\mu_1, \ldots, \mu_T$ are probability measures, $\mathbf{Z} \subset \mathbb{R}^n$ is the random set obtained by $\mathbf{Z} = \left\{\left(f_t\left(\mathbf{x}_i^t\right)\right) : f_t \in \mathcal{F}\right\}$, and G is Gaussian complexity.

Proof. Following Theorem 8 in [32], we have $\mathbb{E}_{\mathbf{X}\sim\mu_t}[f_t(\mathbf{X})] - \sum_i \frac{1}{n_t} f_t(\mathbf{x}_i^t) \leq \frac{\sqrt{2\pi}G(\mathbf{Z})}{n_t} + \sqrt{\frac{9\ln(2/\delta)}{2n_t}}$. Next, we perform the summation operation for t. □

Lemma 1 yields that the task-averaged estimation error is bounded by the Gaussian complexity in multi-task learning. Next, we will give the key property of the Gaussian averages of a Lipschitz image in Lemma 2, and will present the chain rule of Gaussian complexity in Lemma 3.

Lemma 2 (Property of Gaussian Complexity, Corollary 11 in [32])**.** *Suppose $\mathbf{Z} \subseteq \mathbb{R}^n$ and $\phi : \mathbf{Z} \to \mathbb{R}^m$ is (Euclidean) Lipschitz continuous with Lipschitz constant R, we have*

$$G(\phi(\mathbf{Z})) \leq RG(\mathbf{Z}). \tag{11}$$

Lemma 3 (Chain Rule of Gaussian Complexity)**.** *Suppose we have $S = \left\{\left(l(f_t(h(X_i^t)), Y_i^t)\right) : f_t \in \mathcal{F} \text{ and } h \in \mathcal{H}\right\} \subseteq \mathbb{R}^n$. $\mathcal{F}$ is a class of functions $f : \mathbf{Z} \to \mathbb{R}^m$, all of which have Lipschitz constant at most R, $\mathbf{Z} \subseteq \mathbb{R}^n$ has (Euclidean) diameter $D(\mathbf{Z})$. Then, for any $z_0 \in \mathbf{Z}$,*

$$\begin{aligned} G(S) \leq\ & c_1 \rho R G(\mathbf{Z}) + c_2 D(\mathbf{Z}) \mathsf{Range}(f_1, \ldots, f_T) \\ & + \rho G\left(\mathcal{F}\left(z_0\right)\right), \end{aligned}$$

where c_1 and c_2 are universal constants.

Proof. By the Lipschitz property of the loss function $l(\cdot,\cdot)$ and the contraction lemma 2, we have $G(S) \leq \rho G(S')$, where $S' = \left\{\left(f_t(h(X_i^t))\right) : f_t \in \mathcal{F} \text{and } h \in \mathcal{H}\right\} \subseteq \mathbb{R}^n$. Let

$$R(\mathcal{F}) = \sup_{\mathbf{z},\mathbf{z}'\in\mathbf{Z},\mathbf{z}\neq\mathbf{z}'} \mathbb{E} \sup_{f\in\mathcal{F}} \frac{\langle \gamma, f(\mathbf{z}) - f(\mathbf{z}')\rangle}{\|\mathbf{z} - \mathbf{z}'\|}. \tag{12}$$

where γ is a vector of independent standard normal variables. Following Theorem 2 in [31], we have

$$\begin{aligned} G\left(S\right) \leq & c_1 \rho R G(\mathcal{H}(\mathbf{X})) + c_2 \rho D(\mathcal{H}(\mathbf{X})) R(\mathcal{F}) \\ & + \rho \min_{z\in\mathbf{Z}} G(\mathcal{F}(z)). \end{aligned} \tag{13}$$

where c_1 and c_2 are constants. Furthermore,

$$\begin{aligned} & \rho \sup_{\mathbf{z},\mathbf{z}'\in\mathbf{Z},\mathbf{z}\neq\mathbf{z}'} \mathbb{E} \sup_{f\in\mathcal{F}} \frac{\langle \gamma, f(\mathbf{z}) - f(\mathbf{z}')\rangle}{\|\mathbf{z} - \mathbf{z}'\|} \\ =& \sup_{\mathbf{z},\mathbf{z}'\in\mathbf{Z},\mathbf{z}\neq\mathbf{z}'} \frac{\|l(f(\mathbf{z}), y) - l(f(\mathbf{z}'), y')\|}{\|f(\mathbf{z}) - f(\mathbf{z}')\|} \mathbb{E} \sup_{f\in\mathcal{F}} \frac{\langle \gamma, f(\mathbf{z}) - f(\mathbf{z}')\rangle}{\|\mathbf{z} - \mathbf{z}'\|} \\ \leq & \sup_{\mathbf{z},\mathbf{z}'\in\mathbf{Z},\mathbf{z}\neq\mathbf{z}'} \mathbb{E} \sup_{f\in\mathcal{F}} \frac{\langle \gamma, l(f(\mathbf{z}), y) - l(f(\mathbf{z}'), y')\rangle}{\|\mathbf{z} - \mathbf{z}'\|} \\ \leq & \sup_{\mathbf{z},\mathbf{z}'\in\mathbf{Z},\mathbf{z}\neq\mathbf{z}'} \mathbb{E}\left[\sup_{f\in\mathcal{F}} \langle \gamma, l\left(f(\mathbf{z}), y\right)\rangle - \sup_{f\in\mathcal{F}} \left\langle \gamma, l\left(f(\mathbf{z}'), y'\right)\right\rangle\right] \\ \leq & \sup_{\mathbf{z},\mathbf{z}'\in\mathbf{Z},\mathbf{z}\neq\mathbf{z}'} \left[\frac{1}{n}\sum l(f(h(\mathbf{X})), y) - \frac{1}{n}\sum l(f(h(\mathbf{X}')), y')\right] \\ \leq & \max_t \frac{1}{n_t}\sum_{i=1}^{n_t} l(f_t(h(\mathbf{x}_i^t)), y_i^t) - \min_t \frac{1}{n_t}\sum_{i=1}^{n_t} l(f_t(h(\mathbf{x}_i^t)), y_i^t). \end{aligned} \tag{14}$$

□

Finally, we can move to the second step and then derive the generalization error bound related to h and $f_1, \ldots, f_T$ under the setting of long-tail categories on graphs. With the previous assumptions, the generalization bound is given as in the following Theorem 1.

Theorem 1 (Generalization Error Bound)**.** *Given the node embedding extraction function $h \in \mathcal{H}$ and the task-specific classifier $f_1, \ldots, f_T \in \mathcal{F}$, with probability at least $1 - \delta, \delta \in [0, 1]$, we have*

$$\begin{aligned} \epsilon - \hat{\epsilon} \leq \sum_t \Bigg(& \frac{c_1 \rho R G(\mathcal{H}(\mathbf{X}))}{n_t T} + \sqrt{\frac{9\ln(2/\delta)}{2n_t T^2}} \\ & + \frac{c_2 \sup_{h\in\mathcal{H}} \|h(\mathbf{X})\| \mathsf{Range}(f_1, \ldots, f_T)}{n_t T} \Bigg), \end{aligned} \tag{3}$$

where $\mathbf{X}$ is the node feature, T is the number of tasks, n_t is the number of nodes in task t, R denotes the Lipschitz constant of fuctions in $\mathcal{F}$, loss function $l(\cdot,\cdot)$ is ρ-Lipschitz, $G(\cdot)$ denotes the Gaussian complexity [3], and c_1 and c_2 are universal constants.

Proof. By Lemma 1, we have that

$$\epsilon - \hat{\epsilon} \leq \sum_t \left(\frac{\sqrt{2\pi}G(S)}{n_t T} + \sqrt{\frac{9\ln(2/\delta)}{2n_t T^2}} \right), \tag{15}$$

where $S = \left\{\left(l(f_t(h(X_i^t)), Y_i^t)\right) : f_t \in \mathcal{F} \text{ and } h \in \mathcal{H}\right\} \subseteq \mathbb{R}^n$. Next, because we have $f_t(0) = 0$ for all $f_t \in \mathcal{F}$, the last term in (13) vanishes. Substitution in (13) and using Lemma 3, we have

$$G(S) \leq c_1 \rho R G(\mathcal{H}(\mathbf{X})) + c_2 \sqrt{T} D(\mathcal{H}(\mathbf{X})) \mathsf{Range}(f_1, \ldots, f_T). \tag{16}$$

Finally, we bound $D(\mathcal{H}(\mathbf{X})) \leq 2 \sup_h \|h(\mathbf{X})\|$ and substitution in (15), the proof is completed. □

Theorem 1 shows that the generalization performance of long-tail categories on graphs can be improved by (1) reducing the loss range across all tasks $\mathsf{Range}(f_1, \ldots, f_T)$, as well as (2) controlling the task complexity related to T.