Learning Knowledge-diverse Experts for Long-tailed Graph Classification

ZHENGYANG MAO, School of Computer Science, State Key Laboratory for Multimedia Information Processing, PKU Anker LLM Lab, Peking University, Beijing, China
WEI JU, College of Computer Science, Sichuan University, Chengdu, China
SIYU YI, School of Mathematics, Sichuan University, Chengdu, China
YIFAN WANG, School of Information Technology and Management, University of International Business and Economics, Beijing, China
ZHIPING XIAO, College of Computer Science, University of Washington, Seattle, WA, USA
QINGQING LONG, Computer Network Information Center, Chinese Academy of Sciences, Beijing, China
NAN YIN, Mohamed bin Zayed University of Artificial Intelligence, Masdar City, United Arab Emirates
XINWANG LIU, College of Computer, National University of Defense Technology, Changsha, China
MING ZHANG, School of Computer Science, State Key Laboratory for Multimedia Information Processing, PKU Anker LLM Lab, Peking University, Beijing, China

Graph neural networks (GNNs) have shown remarkable success in graph-level classification tasks. However, most of the existing GNN-based studies are based on balanced datasets, while many real-world datasets exhibit long-tailed distributions. In such datasets, the tail classes receive limited attention during training, leading to prediction bias and degraded performance. To address this issue, a range of long-tailed learning strategies have been proposed, such as data re-balancing and transfer learning. However, these approaches encounter several challenges, including insufficient representation capacity for tail classes and their evaluation solely on uniform test data, limiting their capacity to handle unknown class distributions. To tackle these challenges, we introduce a novel framework, namely **Knowledge-diverse Experts (KDEX)** for long-tailed graph classification. Our KDEX leverages a dynamic memory module to enable the transfer of knowledge from head to tail, which

This work is partially supported by National Key Research and Development Program of China with Grant Number 2023YFC3341203, the National Natural Science Foundation of China (NSFC Grant Numbers 62306014 and 62276002) as well as the Postdoctoral Fellowship Program (Grade A) of CPSF under Grant Number BX20240239.
Authors' Contact Information: Zhengyang Mao, School of Computer Science, State Key Laboratory for Multimedia Information Processing, PKU-Anker LLM Lab, Peking University, Beijing, China; e-mail: zhengyang.mao@stu.pku.edu.cn; Wei Ju (corresponding author), College of Computer Science, Sichuan University, Chengdu, China; e-mail: juwei@scu.edu.cn; Siyu Yi, School of Mathematics, Sichuan University, Chengdu, China; e-mail: siyuyi@scu.edu.cn; Yifan Wang, School of Information Technology and Management, University of International Business and Economics, Beijing, China; e-mail: yifanwang@uibe.edu.cn; Zhiping Xiao (corresponding author), College of Computer Science, University of Washington, Seattle, WA, USA; e-mail: patxiao@uw.edu; Qingqing Long, Computer Network Information Center, Chinese Academy of Sciences, Beijing, China; e-mail: qqlong@cnic.cn; Nan Yin, Mohamed bin Zayed University of Artificial Intelligence, Masdar City, United Arab Emirates; e-mail: yinnan8911@gmail.com; Xinwang Liu, College of Computer, National University of Defense Technology, Changsha, China; e-mail: xinwangliu@nudt.edu.cn; Ming Zhang (corresponding author), School of Computer Science, State Key Laboratory for Multimedia Information Processing, PKU-Anker LLM Lab, Peking University, Beijing, China; e-mail: mzhang_cs@pku.edu.cn.

ACM 1556-472X/2025/1-ART32
https://doi.org/10.1145/3705323

improves the representation ability of the tail. To deal with unknown test distributions, KDEX introduces a knowledge-diverse expert training approach to train experts with different capacities in managing various test distributions. Moreover, we train the hierarchical router in a self-supervised manner to dynamically aggregate each knowledge-diverse expert during testing. Experimental results on multiple benchmarks reveal that our KDEX outperforms current baselines in both standard and test-agnostic long-tailed graph classification.

CCS Concepts: • **Computing methodologies** → **Neural networks**; **Learning latent representations**;

Additional Key Words and Phrases: Graph Neural Network, Long-tailed Learning, Multi-expert Learning

Associate Editor: Arijit Khan

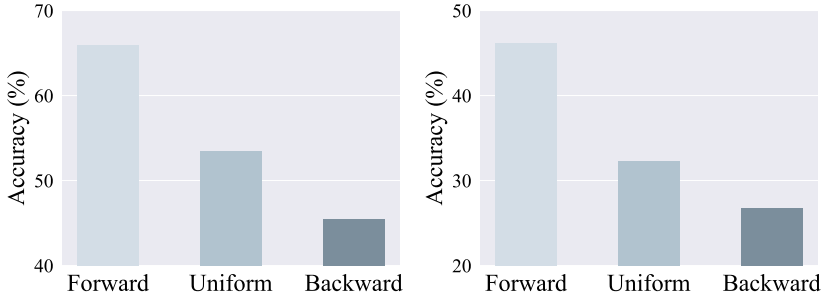
ACM Reference format:
Zhengyang Mao, Wei Ju, Siyu Yi, Yifan Wang, Zhiping Xiao, Qingqing Long, Nan Yin, Xinwang Liu, and Ming Zhang. 2025. Learning Knowledge-diverse Experts for Long-tailed Graph Classification. *ACM Trans. Knowl. Discov. Data.* 19, 2, Article 32 (January 2025), 24 pages.
https://doi.org/10.1145/3705323

1 Introduction

The advanced capabilities of **graph neural networks (GNNs)** [28, 61] to model structured data have recently led to extensive research on graph-level classification. GNNs have become a key technology for learning graph representations [23, 38], demonstrating impressive achievements in a wide range of applications, such as graph-level classification [32, 39, 42], novel drug discovery [36, 69], dynamic systems [40, 41, 75], traffic flow forecasting [10, 25, 82], and recommender systems [22, 33, 50, 51]. The message-passing mechanism in GNNs enables the model to capture both the structural and attributive information. A readout function then combines the representations of individual nodes into a comprehensive graph representation, which is utilized for the downstream tasks.

Despite the impressive performance of GNNs, most of the existing studies are based on datasets with an approximately balanced distribution [17], where each class has a similar number of instances. However, in practical scenarios, many datasets follow a long-tailed distribution [8, 24], where a small number of dominant head classes hold the majority of instances, whereas many tail classes are characterized by a very scarce presence of instances. For example, in the real-world dataset MSRC_21C [47], the head class has about 13.87% of the instances, while the tail class with the least amount only accounts for only 0.4%. The long-tailed distribution poses a great challenge to GNNs, as the model tends to be biased towards the head classes [2, 77]. Consequently, the performance of GNNs on long-tailed datasets is significantly degraded, with the tail classes achieving poor accuracy, which hinders the potential and effectiveness of GNNs trained on real-world long-tailed data.

Long-tailed data distribution has presented a significant challenge for machine learning in recent years, prompting the development of various methods [44, 49, 63, 67]. These approaches fall into three main groups: data re-sampling, cost-sensitive learning, and transfer learning. The re-sampling method [5, 13] seeks to equalize the class distribution by over-sampling tail samples or under-sampling head samples. Cost-sensitive learning [3, 60] re-weights the losses across different classes to ensure that each class contributes appropriately to training. Transfer learning includes head-to-tail knowledge transfer [7, 34] and **model ensemble (ME)** [30, 84]. Head-to-tail knowledge transfer guides feature augmentation for tail-class samples using knowledge from the data-rich head classes, resulting in more informative representations and better tail-class performance. Ensemble learning combines multiple expert networks to achieve reliable and robust predictions. To facilitate long-tailed learning for graph-structured data, many GNN-based methods [78] have been proposed for long-tailed classification. For instance, Wang et al. [65] established a k-nearest neighbor graph



(a) Letter-high (b) Synthie

Fig. 1. Performance of the cost-sensitive learning approach under long-tailed (forward), uniform, and inverse long-tailed (backward) test distributions. Here we manually processed the datasets into three forms of distributions.

and utilized graph-of-graph propagation to aggregate neighboring graph representations. These methods have shown significant improvements in handling long-tailed data distribution for graph-structured data. RAHNet [45] introduced a method for enhancing the tail class by retrieving relevant graphs, whereas CoMe [70] implemented knowledge distillation to improve cooperation among multiple experts. HIERTAIL [62] provides a novel approach to addressing the long-tailed problem by framing it as a multi-task learning framework and introduces a generalization bound that is influenced by the range of losses across all tasks and the complexity of the tasks.

Despite the promising results demonstrated by these state-of-the-art algorithms, most existing approaches still face three significant challenges: (i) *Inadequate representation ability for the tail classes.* A major challenge is the insufficient number of samples for tail classes. Previous methods [6, 46] focusing on balancing head and tail classes during training tend to overlook the significance of effectively transferring knowledge to the tail classes with limited data. Consequently, this can lead to inadequate representation for the tail classes, which in turn adversely impacts their ability to generalize well. (ii) *Inability to manage unknown test distributions.* Many long-tailed learning methods operate under the assumption that the test set distribution is uniform, which is not always the case in practical scenarios. Here, we introduce the concept of an inverse long-tailed test distribution, where the majority class during training becomes the minority class in testing. Take the MIMIC database [19] as an example, where common medical conditions dominate the training data, but the models trained on this database might encounter cases where rare diseases become more prominent due to evolving health trends or new outbreaks. In this case, the rare conditions make up a larger proportion in the testing phase, emphasizing the need for the model to adapt to varying distributions. Figure 1 demonstrates that models trained with current long-tailed recognition approaches (e.g., cost-sensitive learning) perform effectively on test data with uniform or long-tailed class distribution but fail to handle the inverse long-tailed test class distribution. (iii) *Inefficient aggregation of diverse experts.* SADE [79] utilizes a learnable weight to combine various experts. However, our findings indicate that involving all experts in the classification of simple head samples may introduce unnecessary computational overhead and noise, which results in inefficient aggregation and potentially negative knowledge transfer. Therefore, developing efficient aggregation mechanisms to leverage the strengths of diverse experts is crucial for improving the performance of long-tailed learning.

After recognizing the limitations of current algorithms, we present a systematic framework named **Knowledge-diverse Experts (KDEX)** for long-tailed graph classification. In order to tackle the issue of inadequate representation capability for tail classes, we introduce a novel dynamic

memory augmentation (MA) module, which combines direct features and memory features in an adaptive manner to improve the model's generalization ability. Specifically, direct features are obtained by a GNN encoder, while the memory features reflect the discriminative centroids of the direct feature, which complement the lack of sufficient supervision of direct features for the scarce tail class. To enable knowledge learned from data-abundant classes to transfer to data-scarce tail classes, direct features are further fused with suitable memory features dynamically. In addressing the issue of test-agnostic long-tailed classification, we observed that models trained using traditional softmax loss suit for the long-tailed test distribution. However, models equipped with existing long-tailed learning approaches perform better on a uniform class distribution [79]. Drawing inspiration from this observation, our proposed KDEX employs a knowledge-diverse expert learning strategy that generates experts with varying distribution preferences. Each expert is optimized with an expertise-guided objective function to handle specific distributions. In addition, we introduce a hierarchical router that selects experts based on their expertise and reduces expert involvement for simpler test samples. Comprehensive experiments are carried out on both standard and test-agnostic long-tailed graph classification tasks, demonstrating that our approach significantly improves the classification performance on realistic long-tailed graph datasets. The main contributions of this work can be summarized as follows:

—A novel framework called KDEX is proposed to deal with classification with long-tailed graph data. KDEX addresses the problem of poor representation ability for tail classes by developing a dynamic memory mechanism. This mechanism adaptively merges direct and memory-based features, enhancing the knowledge transfer from data-abundant classes to data-scarce classes by integrating direct features with suitable memory features.
—We introduce a strategy called knowledge-diverse expert learning, designed to train experts using various objective functions specifically for managing test-agnostic long-tailed classification. Moreover, a hierarchical router is developed to dynamically aggregate each knowledge-diverse expert at test time in a self-supervised manner.
—Extensive experiments are carried out on both conventional and test-agnostic long-tailed graph classification tasks, which validates the proposed KDEX's superiority over current leading models.

2 Related Work

2.1 Graph Classification

Graph-level classification aims to predict the label of each graph. This problem is different from node classification, which focuses on individual node properties. For graph-level classification, the objective is to extract a meaningful representation that can better represent the entire graph [21, 43, 74]. Current approaches generally fall into two main groups: traditional graph kernel-based methods and GNN-based models. Traditional graph kernel-based methods, such as graphlets [57], random walk [26], and subtree kernel [56], decompose a graph into sub-structures and represent them as a vector of frequencies. These methods have demonstrated the ability to capture different levels of features in graph classification. However, they are heuristic algorithms with poor scalability, flexibility, and high complexity. On the other hand, GNN-based models use convolutional operations to learn hierarchical representations of nodes via message-passing mechanism [12] and employ a readout function to derive representations of the entire graph [29, 59, 64, 71]. Compared to graph kernels, GNNs are more flexible and can handle graphs with varying sizes and structures. In our study, we make a step further and study an under-explored scenario where the training data follows a long-tailed distribution, and the distribution of test classes remains unknown.

2.2 Long-tailed Learning

Existing strategies in long-tailed learning are broadly divided into three groups: data re-sampling, cost-sensitive learning, and transfer learning. Data re-sampling [5, 13] aims to equalize the distribution of different classes, either by increasing the presence of tail class instances or reducing that of head class instances, which has widely used in vision and text domains. Cost-sensitive learning [3, 60] re-weights the loss across different classes to ensure appropriate contributions to training. While the first two approaches often focus on enhancing the performance of tail classes, they might inadvertently harm the accuracy of head classes. Transfer learning, on the other hand, focuses on leveraging knowledge from well-represented areas to enhance performance in less represented ones [80]. This category involves head-to-tail knowledge transfer [7, 34] and ensemble learning [30, 84].

To adapt long-tailed learning to graph-structured data, various GNN-based methods recently have been proposed for long-tail recognition on graphs [35, 37, 58, 76, 81, 83]. These approaches primarily involve designing node-specific transformations or utilizing structural features to improve long-tailed node-level classification. GraphSMOTE [81] extends the SMOTE [5] algorithm and synthesizes new features for tail nodes by interpolating between two tail nodes. ImGAGN [52] generates virtual tail nodes by mixing all existing tail nodes together. However, our proposed method KDEX differs from these approaches as it focuses on the under-explored and promising field of long-tailed graph-level classification. GraphDIVE [15] tackles the imbalance problem by leveraging multi-view graph representations and combining the expertise of multiple experts. These multi-view representations capture the diverse intrinsic characteristics of the graph's topological structure. HIERTAIL [62] also presents an innovative framework for long-tailed classification on graphs. It organizes related tasks into hypertasks and integrates a balanced contrastive learning module that adaptively adjusts gradients for both head and tail classes, offering unified control over the loss range across all tasks. RAHNet [45] directly tackles the long-tailed challenge at the graph level by retrieving relevant graphs to augment tail classes and employing balanced contrastive learning to enhance representations.

2.3 Mixture-of-Experts (MoE)

MoE employs multiple expert networks to divide the problem space into homogeneous segments [1]. In contrast to ensemble methods, MoE typically utilizes only a few expert models to generate the final prediction. Thus, a gating network is essential to decide how much each expert contributes to the final prediction. There are three types of gating networks, i.e., soft gating, hierarchical gating, and sparse gating. Soft gating [18] assigns a weight computed by a soft activation function to each expert, which determines how much each expert's prediction contributes to the final output. Hierarchical gating [20] splits the input space into overlapping areas and applies simple models to the data points within each area, with soft boundaries allowing data to lie simultaneously in multiple regions. Sparse gating [55] assigns weights only to specific experts, leading to significant improvements in model capacity and training time. Switch Transformer [11] further simplifies the sparse gating by using Softmax activation over the hidden state to select the top expert, resulting in superior scaling compared to earlier methods. Moreover, recent studies also combine MoE technique with GNN. For instance, GraphMETRO [66] utilizes an MoE architecture featuring a gating model and multiple expert models. Each expert generates a shift-invariant representation, which helps to naturally address the distributional shifts commonly found in graph data. This work takes a further step towards learning experts with diverse knowledge and utilizing a novel hierarchical router to aggregate them.

3 Preliminaries

Long-tailed Dataset. Consider a graph dataset $\mathcal{G} = \{G_i = (V_i, E_i, \mathbf{X}i), y_i\}_{i=1}^{N}$ with C distinct classes, in which each G_i represents the ith graph comprising a set of nodes V_i, edges E_i, and a node attribute matrix $\mathbf{X}_i$. The class label for G_i is y_i, which falls within the set $\{1, \ldots, C\}$. The total number of graphs belonging to the jth class is N_j, such that the sum across all classes equals N, and it's assumed that $N_1 \geq N_2 \geq \cdots \geq N_C$. The **imbalance factor (IF)** of the dataset, defined as N_1/N_C, quantifies the level of class imbalance. In accordance with Zipf's law [48], a dataset exhibits a long-tailed distribution if $N_i = N_1 \times i^{-\mu}$ for $i = 1, \ldots, C$, where μ is a parameter that determines the imbalance severity, as reflected by the IF. For real-world data, the long-tailed class distribution probably does not exactly, but approximatively, follows Zipf's law.

Task Definition. The aim in addressing the long-tailed graph classification challenge is to develop a classifier that can effectively classify graphs within a long-tailed dataset $\mathcal{G}$ without bias. The classifier must be able to extract distinctive features for less represented tail classes without being dominated by the abundant head classes, and classify both head and tail classes accurately using certain balanced metrics. For instance, the average accuracy metric $\text{avg\_acc} = \frac{1}{C}\sum_{j=1}^{C} \text{acc}_j$ can be used to give equal importance to both head and tail classes, where acc_j denotes the accuracy of graphs in class j. For the standard long-tailed classification, the distribution of test classes $d_t(y)$ is designed to be uniform (i.e., $d_t(y) = 1/C$). The goal of test-agnostic long-tailed classification is to train a model on a skewed training distribution $d_s(y)$ that is adept at performing across various test distributions, whether they mirror the long-tailed training distribution ($d_t(y) = d_s(y)$), or even present an inverse long-tail pattern ($d_t(y) = \text{inv}\,(d_s(y))$). The function $\text{inv}(\cdot)$ is used to denote the reversal of the distribution such that the less frequent training classes become more common in the test set and vice versa.

GNN-based Classifier. In order to generate effective probability assignments for each graph, we employ GNN to derive the graph embedding, which can be defined as $f_\theta(\cdot)$. GNN is able to capture the information of node attributes and topological structure in a reasonable manner. More precisely, the propagation rule of a GNN layer can be written as:

$$\mathbf{h}_v^{(l+1)} = C_{\boldsymbol{\theta}}^{(l)}\left(\mathbf{h}_v^{(l)}, \mathcal{A}_{\boldsymbol{\theta}}^{(l)}(\{\mathbf{h}_{v'}^{(l)}\}_{v' \in \mathcal{N}(v)})\right), \tag{1}$$

where $\mathbf{h}v^{(l)}$ signifies the embedding of vertex v obtained at layer l, with $\mathcal{N}(v)$ indicating vertex v's adjacent vertexes. The functions $C\boldsymbol{\theta}^{(l)}$ and $\mathcal{A}_{\boldsymbol{\theta}}^{(l)}$ serve as the layer-specific operations for combining and aggregating vertex information, respectively. To obtain a graph-level representation, a *READOUT* function is involved, which formulates the graph representation for a given graph G as:

$$\mathbf{h} = READOUT(\{\mathbf{h}_v^{(L)} : v \in V\}). \tag{2}$$

Here V represents the vertex set of the graph G. By merging attribute and structural data, a comprehensive graph-level representation is crafted. This aggregated representation is initially processed through a **multi-layer perceptron (MLP)** and subsequently through a softmax function, which calculates the probability distributions for graph G.

4 The Proposed Method

KDEX mainly contains three modules, i.e., dynamic MA, knowledge-diverse experts learning, and hierarchical aggregation. Figure 2 illustrates a summary of the KDEX framework. In the following sections, we delve into the details of KDEX's three primary components.

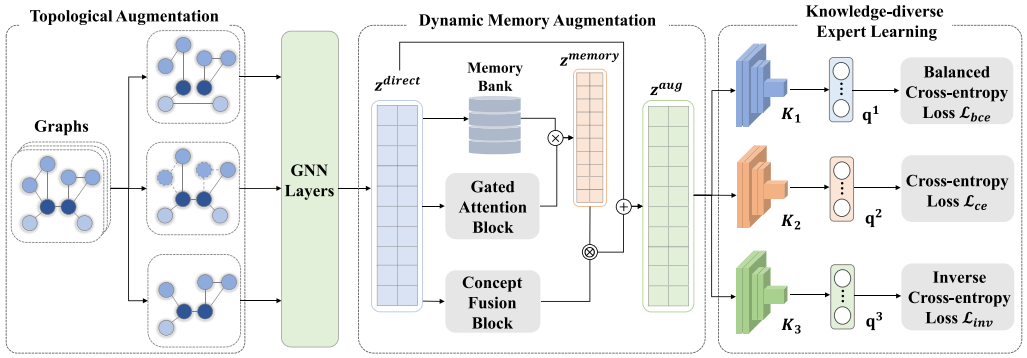


Fig. 2. Overview of the proposed framework KDEX. Dynamic MA and topological augmentation (TA) are performed to jointly enhance the representation learning of both head and tail classes. Using the learned representation, the framework trains multiple experts through various expertise-guided objective functions, each having unique knowledge to handle different class distributions.

4.1 Dynamic MA

We introduce a dynamic MA module aimed at facilitating the transfer of knowledge from data-abundant head classes to data-scarce tail classes. This module dynamically integrates direct features with newly proposed memory features, enhancing the representation of both head and tail classes comprehensively. Furthermore, we employ diverse **topological augmentations (TAs)** to increase the within-class variety for all classes.

4.1.1 MA. $\mathbf{z}^{direct}$ is the direct graph-level representation derived straight from the GNN encoder, specifically $\mathbf{z}^{direct} = \mathbf{h}$. The direct features often experience a decline in generalization capabilities for tail classes due to limited supervision. Therefore, we suggest enhancing the direct graph representation with an additional memory feature to improve discriminability. The memory feature, $\mathbf{z}^{memory}$, is cultivated through a memory association process that aggregates knowledge from both data-abundant classes and data-scarce classes, employing varying levels of memory activation. We dynamically combine $\mathbf{z}^{direct}$ and $\mathbf{z}^{memory}$ to obtain the final augmented embedding $\mathbf{z}^{aug}$, which enhances generalizability to novel instances, particularly for tail classes. The MA module can be represented as an augmented function $f_\phi(\cdot)$, which takes the direct feature as input and produces an enhanced embedding as output: $\mathbf{z}^{aug} = f_\phi(\mathbf{z}^{direct})$.

Construct Memory Bank. Motivated by [34], we construct the memory bank $\mathbf{M}$ using a collection of class prototypes $\mathbf{M} = \{\mathbf{p}_i\}_{i=1}^{C}$, where C corresponds to the counts of classes. Each memory prototype in $\mathbf{M}$ captures the collective information and acts as the centroid of its respective class, while maintaining its distinctiveness from other memory prototypes. The prototype for the ith class is computed as:

$$\mathbf{p}_i = \frac{1}{N_i} \sum_{n=1}^{N_i} \mathbf{z}_n^{direct}. \tag{3}$$

Here, N_i denotes the count of samples belonging to the ith class, and $\mathbf{z}_n^{direct}$ refers to the direct feature associated with the nth sample. To further diversify the memory bank, we can leverage $\mathbf{z}^{direct}$ to create k more prototypes for each category.

Aggregate Memory Feature via Gated Attention Block. For any given graph sample, we associate the prototypes stored in the constructed memory bank by the gated attention block to assemble

the memory feature. The memory feature serves as the knowledge carrier between different categories, especially from data-abundant classes to data-scarce classes. To aggregate while excluding non-essential information, we employ an attention-based memory aggregation mechanism. This approach evaluates the significance of each class prototype and dynamically combines them by applying distinct weights for each prototype:

$$\mathbf{z}^{memory} = \mathbf{o}^{\top}\mathbf{M} = \sum_{i=1}^{C} \mathbf{o}_i \mathbf{p}_i, \tag{4}$$

where $\mathbf{M}$ is the memory bank matrix that contains C prototypes and $\mathbf{o} \in \mathbb{R}^C$ signifies the attention weights assigned to the prototype of each category, calculated from direct features. This is achieved by applying a softmax activation function to the outputs of a fully-connected layer, denoted as $\mathbf{W}_o$, to derive the attention weights:

$$\mathbf{o} = Softmax\left(\mathbf{W}_o \mathbf{z}^{direct}\right). \tag{5}$$

Compose Augmented Embedding via Concept Fusion Block. Memory features are vital for improving representations and compensating for the limited direct features in tail classes. However, memory features may not contain useful information for data-rich head classes. To address this issue, we introduce a concept fusion block, which evaluates the contribution of memory features to different classes and adaptively combines them with direct features. We define the resulting embedding as augmented embedding, which is a dynamic combination of direct and selected memory features. This approach boosts the representations of tail classes without compromising the quality of representations for head classes:

$$\mathbf{z}^{aug} = \mathbf{z}^{direct} + \mathbf{s} \otimes \mathbf{z}^{memory}, \tag{6}$$

where $\mathbf{s}$ is the concept fusion block, and $\otimes$ signifies the element-wise multiplication. For deriving the weights from $\mathbf{z}^{direct}$, we use a lightweight network that includes a fully-connected layer $\mathbf{W}_s$ and a $Tanh$ activation function:

$$\mathbf{s} = Tanh\left(\mathbf{W}_s \mathbf{z}^{direct}\right). \tag{7}$$

4.1.2 TA. Dynamic MA, which serves as global knowledge enrichment, has already enhanced the quality of classification on long-tailed data. However, the model's generalizability to massive unseen graphs is still limited by scarce data samples in the tail classes. To produce generalized and robust representations for tail classes, we further involve data augmentation to enrich the graph topological information. In particular, we utilize four key TA techniques as suggested in **graph contrastive learning (GraphCL)** [72]: attribute masking, node dropping, edge perturbation, and subgraph extraction.

In practice, we leverage the aforementioned graph augmentation techniques into expert learning to encourage the discrepancy and enhance the diversity of dynamic memory. Specifically, a graph G_i performs stochastic graph augmentations $\mathcal{T}(\cdot|G_i)$ to obtain semantically preserved augmented graph $\hat{G}_i$ by randomly choosing from one augmentation strategy. Then the direct feature can be recalculated as $\mathbf{z}^{direct} = f_\theta(\hat{G})$.

4.2 Knowledge-diverse Expert Learning

One of the primary challenges in training an MoE model on long-tailed datasets is the necessity to learn experts with diverse and effective capabilities from a single dataset. The current ensemble-based methods to address long-tailed datasets focus solely on training experts to handle a uniform distribution of test classes, which limits their efficacy for diverse class distributions.

According to [79], the loss function used during training is strongly correlated with the learned class distribution. Models trained with softmax loss perform well on long-tailed distributions, while long-tailed methods excel in uniform distributions. Therefore, we introduce a knowledge-diverse expert framework, where each expert has different expertise in handling various distributions. This framework includes three types of experts: the uniform expert, the forward expert, and the backward expert. The uniform expert ensures high classification performance on uniform distributions, while the forward and backward experts acquire knowledge from long-tailed and inverse long-tailed distributions, respectively. Notably, all three knowledge-diverse experts utilize the same GNN encoder and dynamic MA module. After obtaining the augmented embedding, each expert forwards the learned representation through the expert-specific fully-connected classifier layers to obtain the prediction probabilities, distinguished as $\mathbf{q}^1$, $\mathbf{q}^2$, and $\mathbf{q}^3$ for the three experts. We train them using three distinct expertise-guided losses to optimize their capabilities.

The Uniform Expert. For the uniform expert K_1, our goal is to train an expert that excels at handling uniform class distributions. We opt for the balanced cross-entropy loss to train this expert, motivated by the success of logit-adjusted losses [46]. The softmax probability assigned to the cth class can be represented as $\frac{\exp(\mathbf{q}_c^1)}{\sum_{i=1}^{C}\exp(\mathbf{q}_i^1)}$. To mitigate the effects of a skewed class distribution, we implement a balanced softmax approach that adjusts the class prediction probabilities based on the prior occurrence rates of the classes in the training data: $\frac{\exp(\mathbf{q}_c^1+\log \pi^c)}{\sum_{i=1}^{C}\exp(\mathbf{q}_i^1+\log \pi^i)}$. Here, we use π^c to denote the label frequency of the cth class in the training set, i.e., $\pi^c = \frac{N_c}{N}$. Given the uniform expert's prediction probabilities $\mathbf{q}^1$, and labels y, we can formally define the balanced cross-entropy loss as:

$$\mathcal{L}_{bce}\left(\mathbf{q}^1, y\right) = \frac{1}{N}\sum_{i=1}^{N} -\log \frac{\exp\left(\mathbf{q}_{i,y_i}^1 + \log \pi^{y_i}\right)}{\sum_{j=1}^{C}\exp\left(\mathbf{q}_{i,j}^1 + \log \pi^{j}\right)}. \tag{8}$$

Here, $\mathbf{q}_{i,j}^1$ denotes the probability assigned to the jth class for the ith sample in the predictions.

The Forward Expert. In the training process, the forward expert K_2 is optimized by the softmax cross-entropy loss to enhance its ability to learn from the originally skewed distribution of the training classes. This is due to the cross-entropy loss naturally giving greater weight to the head classes, the forward expert is well-suited to perform well in these classes. Given the prediction probabilities $\mathbf{q}^2$, and corresponding labels y, the loss function used to optimize the forward expert is:

$$\mathcal{L}_{ce}\left(\mathbf{q}^2, y\right) = \frac{1}{N}\sum_{i=1}^{N} -\log \frac{\exp\left(\mathbf{q}_{i,y_i}^2\right)}{\sum_{j=1}^{C}\exp\left(\mathbf{q}_{i,j}^2\right)}. \tag{9}$$

The Backward Expert. We aim to train the backward expert K_3 to effectively handle the long-tailed class distribution, with a focus on the tail classes. In order to achieve this, we propose using an inverse cross-entropy loss function that is similar to logit-adjusted losses. To be specific, we alter the logit-adjusted losses and add an inverse training prior $\tilde{\pi}$, the softmax probability to be expressed as: $\frac{\exp(\mathbf{q}_c^3+\log \pi^c-\log \tilde{\pi}^c)}{\sum_{i=1}^{C}\exp(\mathbf{q}_i^3+\log \pi^i-\log \tilde{\pi}^i)}$. Here the inverse training prior $\tilde{\pi}$ reverses the training label frequencies π, calculated as $\tilde{\pi}^i = \frac{N_{C-i}}{N}$. With the backward expert's prediction probabilities $\mathbf{q}^3$, and labels y, we can define the inverse cross-entropy loss as:

$$\mathcal{L}_{inv}\left(\mathbf{q}^3, y\right) = \frac{1}{N}\sum_{i=1}^{N} -\log \frac{\exp\left(\mathbf{q}_{i,y_i}^3 + \log \pi^{y_i} - \lambda \log \tilde{\pi}^{y_i}\right)}{\sum_{j=1}^{C}\exp\left(\mathbf{q}_{i,j}^3 + \log \pi^{j} - \lambda \log \tilde{\pi}^{j}\right)}, \tag{10}$$

where λ is a hyper-parameter that determines the extent of adjustment based on the inverse frequencies of the labels.

To optimize our proposed framework KDEX, we combine the balanced cross-entropy loss $\mathcal{L}_{bce}$, cross-entropy loss $\mathcal{L}_{ce}$, and inverse loss $\mathcal{L}_{inv}$ for joint optimization. The total training loss, $\mathcal{L}_{train}$, is thus defined as:

$$\mathcal{L}_{train} = \mathcal{L}_{bce} + \mathcal{L}_{ce} + \mathcal{L}_{inv}, \tag{11}$$

and the complete training procedure of our KDEX is outlined in Algorithm 1.

Algorithm 1: The Training Procedure of KDEX

Input: Training dataset $\mathcal{G} = \{G_i, y_i\}_{i=1}^{N}$; Maximum iterations T_{max};
Output: GNN encoder $f_\theta(\cdot)$; Knowledge-diverse experts K_1, K_2, and K_3;

for $t = 1$ to T_{max} **do**
  Obtain the direct feature $\mathbf{z}^{direct}$ with $f_\theta(\cdot)$;
  Generate memory prototypes $\mathbf{M}$ using Equation (3);
  Assemble memory feature $\mathbf{z}^{memory}$ using Equation (4);
  Compose augmented embedding $\mathbf{z}^{aug}$ using Equation (6);
  Obtain prediction probabilities $\mathbf{q}^1$, $\mathbf{q}^2$, and $\mathbf{q}^3$ based on K_1, K_2, and K_3;
  Calculate $\mathcal{L}_{bce}$ with $\mathbf{q}^1$ and y using Equation (8);
  Calculate $\mathcal{L}_{ce}$ with $\mathbf{q}^2$ and y using Equation (9);
  Calculate $\mathcal{L}_{inv}$ with $\mathbf{q}^3$ and y using Equation (10);
  Perform back-propagation and optimize the parameters by minimizing $\mathcal{L}_{train}$ in Equation (11);
end for

4.3 Hierarchical Expert Aggregation

Using the knowledge-diverse expert learning strategy, KDEX trains three experts to specialize in different class distributions. SADE [79] and GraphMETRO [66] both utilized a method for aggregating expert inputs using learnable weights. However, we observed that this method tends to involve all experts, even for simple head samples, potentially introducing unnecessary noise and computational overhead. To address the above challenge, we develop a hierarchical expert routing module as shown in Figure 3, which leverages a two-level router to efficiently control the aggregation of the experts. Moreover, motivated by the success of self-supervised learning in deep neural models, we also propose to optimize the routing parameters in a self-supervised manner.

4.3.1 Hierarchical Routing. To address the first challenge, we first investigate the performance of three knowledge-diverse experts on various test distributions. From Table 1, we can observe that the forward and the backward experts are highly skilled in the head and tail classes, respectively. However, it can be found that in most test distributions, the uniform expert outperforms others in terms of overall accuracy.

Motivated by this finding, we propose a two-level hierarchical routing mechanism, in which we assign the uniform expert for all the testing instances and dynamically control the engagement of the forward and backward experts. The first-level router decides whether to engage the other two experts in the decision-making process, distinguishing our method from existing soft gating mechanisms like GraphMETRO, which treats every expert equally.The second-level router fuses the forward and backward experts with different weights, only if it passes through the first router. This second-level router operates similarly to a soft gating model. The hierarchical routing mechanism improves

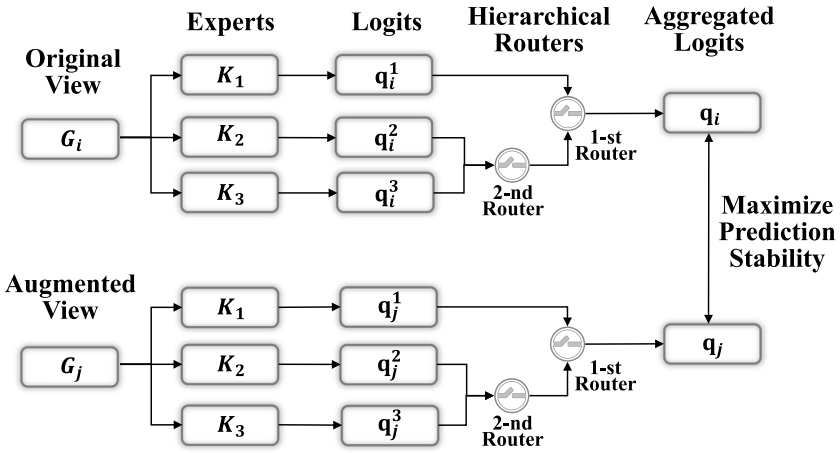


Fig. 3. Illustration of the hierarchical expert aggregation. The predictions of knowledge-diverse experts are aggregated by a two-level router, and the hierarchical routers are optimized by maximizing the prediction stability between two views.

Table 1. Accuracy of Knowledge-diverse Experts Tested across a Range of Distributions, Namely Uniform, Forward, and Backward Long-tailed Distributions with IFs of 5 and 10

Test distribution	The uniform expert				The forward expert				The backward expert			
	Head	Med	Tail	All	Head	Med	Tail	All	Head	Med	Tail	All
Forward_10	80.37	56.18	35.33	65.91	83.33	52.73	28.00	66.81	40.74	47.27	40.00	42.19
Forward_5	79.37	56.37	32.50	61.26	81.87	54.49	17.50	58.79	40.93	51.35	50.00	46.02
Uniform	72.60	56.32	31.47	50.72	76.40	47.20	16.00	42.50	38.60	43.52	40.40	40.96
Backward_5	72.00	58.00	32.34	43.96	72.00	50.63	16.29	32.93	34.00	48.12	45.43	44.36
Backward_10	84.44	52.47	36.61	42.05	82.22	45.88	20.00	30.98	44.44	36.47	44.00	42.64

The experts are trained on long-tailed Letter-high datasets with IF of 50. Classes are divided into the head, med, and tail regions according to their numbers of samples.

the efficiency and performance of the knowledge-diverse experts during testing by selecting the experts with the best expertise and reducing unnecessary expert engagement dynamically.

First-level Router. We employ the first-level router to make a binary decision y_{on} on whether to allocate the forward and backward router for each training example individually. Since we lack information on the class distribution and the label during testing, we propose to determine the router state based on the confidence of the uniform expert's prediction. Intuitively, if the uniform expert is unable to provide confidence prediction for an unknown testing instance, but the forward and the backward experts have the corresponding knowledge to handle the given instance, then the router should output $y_{\text{on}} = 1$ ideally or $y_{\text{on}} = 0$ otherwise. Thus, we propose to train the first-level router using a simple confidence-based binary classifier. This process includes projecting the prediction output $\mathbf{q}^1$ of the uniform expert through a fully connected layer $\mathbf{W}_1$, and applying the Sigmoid function $S(x)$ to obtain a real activation value in the range of $[0, 1]$:

$$r_1(G_i) = S(\mathbf{W_1}\mathbf{q}_i^1), \tag{12}$$

where $r_1(\cdot)$ is the resulting first-level routing function. For the given test instance G_i, y_{on} is set to 1 when $r_1(G_i) > \gamma$, where γ is a threshold hyper-parameter and is emprically set to 0.5.

Second-level Router. The second-level router is employed only when $y_{\text{on}} = 1$, and it serves the purpose of dispatching the testing sample to the right candidate among the forward and backward experts. Inspired by the wide application of the MoE framework [11], we employ a gating function to regulate the combination of experts. These functions assign weights in a soft manner, determined by the similarity between the graph embedding $f_\theta(G_i)$ and a learnable gating prototype $\mathbf{w}_k$. This can be mathematically expressed as follows:

$$r_k(G_i) = \frac{\exp\left(f_\theta(G_i) \cdot \mathbf{w}_k\right)}{\sum_{k'=2}^{3} \exp\left(f_\theta(G_i) \cdot \mathbf{w}_{k'}\right)}, \tag{13}$$

where $r_2(\cdot)$ and $r_3(\cdot)$ denote the gating functions for the forward and backward experts, respectively.

Finally, with the two-level hierarchical routers, we can efficiently aggregate the three experts and obtain the output prediction probability of the KDEX framework for the unknown test graph G_i as:

$$\mathcal{R}(G_i) = \mathbf{q}_i^1 + \mathbb{I}(r_1(G_i) > \gamma) \sum_{k=2}^{3} r_k(G_i) \cdot \mathbf{q}_i^k, \tag{14}$$

where $\mathbb{I}(\cdot)$ is the indicator function to determine whether the first-level router should be turned on. By aggregating the experts hierarchically, the knowledge-diverse experts can efficiently adapt to different test class distributions with minimal expert engagement, which significantly enhances both the performance and efficiency of the proposed model.

4.3.2 Self-supervised Aggregation. Through the hierarchical routing mechanism, the experts within KDEX are effectively combined to address various test distributions. The remaining challenge is how to optimize the routing parameters using unlabeled test data. Motivated by SADE [79], which proposes a key insight that the routing functions should be aware of the adeptness of each expert and ensure stable allocation of strong experts to the testing samples despite perturbations. In light of this, we adopt the relative prediction stability as a means to optimize the routing functions for the unknown test class distribution.

The maximization of the prediction stability strategy is employed to learn the routing functions by generating two distinct views of the testing data. Specifically, for a given original view of graph G_i, we also use the same TA techniques introduced in Section 4.1.2 to generate a perturbed view $G_j \sim \mathcal{T}(\cdot|G_i)$. Following GraphCL [72], a mini-batch of B testing samples are fed into our framework to obtain prediction probabilities, which are then aggregated using hierarchical routing. The aggregated prediction probabilities for the nth sample in the mini-batch are denoted as $\mathbf{q}_{n,i}$ and $\mathbf{q}_{n,j}$ for the original and perturbed views, respectively. We employ the cosine similarity loss to maximize the prediction stability between the two correlated views:

$$\mathcal{L}_{rout} = -\frac{1}{B} \sum_{n=1}^{B} \mathbf{q}_{n,i} \cdot \mathbf{q}_{n,j}. \tag{15}$$

During test-time training, updates are applied solely to the parameters of the routing layers. Since experts with greater proficiency often exhibit higher prediction concordance for classes within their area of expertise, enhancing prediction stability would lead to increased involvement of these skilled experts, enabling more effective handling of the unknown test class distribution.

4.4 Time Complexity Analysis

Given N as the total count of graphs in the dataset and $|V|$ representing the average node count per graph, the computational complexity for generating embeddings via the GNN encoder is $O(NLD|V|)$, with L denoting the GNN encoder's layer count and D the embedding dimension. The time complexity of obtaining augmented embedding is $O(ND + CD)$, where C signifies the class

Table 2. Statistics of the Graph Datasets Used in Our Experiments

Dataset	# Graphs	# Average nodes	# Average edges	# Classes
COLLAB	5,000	74.49	2,457.78	3
ENZYMES	600	32.63	62.14	6
MNIST	60,000	75	1,393.27	10
Letter-high	2,250	4.67	4.50	15
Letter-low	2,250	4.67	4.50	15
COIL-DEL	3,900	21.54	54.24	100
ogbg-ppa	158,110	243.4	2,266.1	37

count. For computing prediction probabilities of knowledge-diverse experts, the time complexity is $O(KND^2)$, with K being the expert count. The hierarchical routing's computational demand is also $O(ND + CD)$. Consequently, the overall computational complexity of KDEX is to $O(NLD|V| + ND + CD + KND^2)$.

5 Experiments

5.1 Experimental Setup

5.1.1 Benchmark Datasets. In our evaluation of the proposed KDEX, we compare it with various baseline models across seven publicly available datasets from different domains. These include: social networks (COLLAB [68]), bioinformatics (ENZYMES [54], ogbg-ppa [16]), and vision datasets (MNIST [9], Letter-high [53], Letter-low [53], COIL-DEL [53]). The statistics of the datasets are reported in Table 2.

To strictly ensure that the datasets follow Zipf's law [48], we transformed the training datasets into long-tailed versions by applying various IFs, whereas we kept the validation and test datasets balanced for standard long-tailed classification. For each dataset, we select specific IFs to guarantee that the smallest tail class in terms of samples has a training size ranging from 2 to 4 (except for MNIST due to its larger scale). For the test-agnostic long-tailed classification, we adjusted the test sets to exhibit both long-tailed and inverse long-tailed distributions, simulating situations where test distributions are unknown.

5.1.2 Baselines. In our evaluation of the proposed KDEX, we compare it with a range of competing baselines. These baselines encompass four main strategies: (a) Data re-sampling methods, including up-sampling [4]; (b) Loss re-balancing methods, including **class-balanced (CB)** loss [6] and **logit adjusted cross-entropy (LACE)** loss [46]; (c) Information augmentation methods, including graph augmentation [73] and G^2GNN [65]; (d) Contrastive learning-based methods: GraphCL [72] and supervised contrastive learning (SupCon) [27]. One recent baseline RAHNet [45] with hybrid strategies is also included in evaluation.

5.1.3 Implementation Details. In our experimentation, we utilized GraphSAGE [14] as the core GNN encoder, complemented by a two-layer MLP classifier. The Adam optimizer was the default choice, with a learning rate of 0.0001 for preliminary training processes. And we set the batch size to 32. For our proposed KDEX, we set the hyper-parameter λ for inverse cross-entropy to 1.0, and the augmentation ratio δ to 0.1. During the test-time training period, we applied a weight decay of 0.0005, while maintaining the same optimizer and learning rate settings as those used in the initial training phase. Implementations for baseline models were conducted in PyTorch, based on publicly

Table 3. Results (Mean%±SD%) of Long-tailed Graph Classification

Model	COLLAB		ENZYMES		MNIST		Letter-high		Letter-low		COIL-DEL	
	IF = 10	IF = 20	IF = 15	IF = 30	IF = 50	IF = 100	IF = 25	IF = 50	IF = 25	IF = 50	IF = 10	IF = 20
SAGE	63.07 ± 0.90	53.33 ± 0.73	30.66 ± 1.83	25.16 ± 2.03	68.67 ± 1.38	63.46 ± 1.69	51.06 ± 1.81	42.16 ± 1.58	86.00 ± 1.22	84.32 ± 1.16	38.80 ± 1.27	31.32 ± 1.41
US	72.33 ± 0.96	70.25 ± 1.01	32.33 ± 1.47	28.50 ± 1.76	64.69 ± 1.85	59.78 ± 2.15	53.62 ± 1.77	44.20 ± 1.82	88.48 ± 1.49	86.72 ± 0.97	39.20 ± 0.99	26.96 ± 1.33
CB loss	68.78 ± 1.15	65.85 ± 1.34	32.19 ± 1.43	26.83 ± 1.66	68.85 ± 1.67	63.40 ± 1.97	53.76 ± 1.80	45.06 ± 1.87	88.01 ± 1.18	86.46 ± 1.34	41.72 ± 1.65	32.34 ± 1.79
LACE loss	68.33 ± 1.21	64.77 ± 1.30	33.63 ± 1.39	26.70 ± 1.57	69.72 ± 1.65	64.59 ± 1.74	55.62 ± 1.75	47.68 ± 1.81	88.81 ± 1.41	86.34 ± 1.20	41.96 ± 1.84	32.18 ± 1.82
Aug	72.85 ± 0.99	71.14 ± 1.20	32.08 ± 1.35	26.75 ± 1.61	72.18 ± 1.25	68.17 ± 1.47	57.18 ± 1.46	47.21 ± 2.17	88.32 ± 0.95	86.40 ± 1.31	38.18 ± 1.45	30.80 ± 1.79
G^2GNN_n	73.94 ± 1.49	71.89 ± 1.62	35.00 ± 1.77	29.17 ± 1.85	70.91 ± 1.81	66.73 ± 1.94	58.91 ± 1.82	51.12 ± 1.95	89.49 ± 1.39	87.98 ± 1.51	38.32 ± 1.94	27.98 ± 2.10
G^2GNN_e	74.50 ± 1.45	72.76 ± 1.70	35.83 ± 1.89	29.50 ± 1.93	73.69 ± 1.57	70.31 ± 1.72	58.85 ± 1.97	49.96 ± 2.12	89.84 ± 1.27	87.80 ± 1.40	39.18 ± 1.87	31.06 ± 1.95
GraphCL	69.33 ± 1.20	67.36 ± 1.31	36.66 ± 1.61	29.83 ± 1.74	69.37 ± 1.49	65.12 ± 1.41	57.34 ± 1.75	48.93 ± 1.87	89.28 ± 1.21	87.89 ± 1.17	42.02 ± 1.75	33.19 ± 1.91
SupCon	69.25 ± 1.19	67.14 ± 1.32	37.08 ± 1.70	30.67 ± 1.78	69.76 ± 1.50	64.88 ± 1.54	57.29 ± 1.67	48.93 ± 1.82	89.12 ± 1.32	87.36 ± 1.30	42.93 ± 1.48	34.20 ± 1.75
RAHNet	74.31 ± 1.37	72.93 ± 1.45	**39.02 ± 1.52**	**34.16 ± 1.65**	75.79 ± 1.23	72.60 ± 1.41	59.79 ± 1.87	52.90 ± 1.99	90.19 ± 1.67	89.28 ± 1.45	45.32 ± 1.42	**38.48 ± 1.59**
KDEX	**76.79 ± 1.10**	**74.35 ± 1.18**	38.67 ± 1.54	33.54 ± 1.59	**76.67 ± 0.91**	**72.77 ± 1.15**	**63.80 ± 1.64**	**53.29 ± 1.75**	**91.55 ± 1.43**	**90.48 ± 1.11**	**47.48 ± 1.28**	37.32 ± 1.50

Each dataset is processed with two varying degrees of IFs. The highest classification results are highlighted in bold, while the second highest are underlined for clarity. SAGE, US, and Aug represent GraphSAGE, up-sampling, and augmentation, respectively.

available codes from their original studies. The evaluation focused on the top-1 average accuracy from 5 runs, for both standard and test-agnostic long-tailed classification scenarios.

5.2 Comparisons on Long-tailed Graph Classification

This section presents the performance assessment of KDEX in the context of standard long-tailed graph classification, benchmarking it against a range of long-tailed learning baseline approaches. The results conducted on seven benchmark datasets with varying IFs, are summarized in Table 3. Insights gained from these quantitative findings include the following key observations:

- —With the rise in the disparity between head and tail classes (IF), there's a noticeable drop in performance for both the baseline methods and our proposed KDEX. For example, a doubling of the IF leads to a 17.43% reduction in the classification accuracy of the GraphSAGE model on the Letter-high dataset. This observation aligns with our initial hypothesis that GNNs are particularly vulnerable to long-tailed distributions and their performance is degraded in these scenarios.
- —The findings suggest that methods involving information augmentation tend to surpass those based on data re-sampling and cost-sensitive learning. This superiority might stem from the fact that rebalancing strategies fail to actively introduce new knowledge to enhance the underrepresented classes. Additionally, baselines utilizing contrastive learning show consistently stable performance across various datasets.
- —Our method consistently achieves the best results on most datasets. This superior performance can be attributed to the use of dynamic memory to transfer additional knowledge from well-represented classes to less-represented classes, which further leads to representations that preserve better generalization ability. Notably, KDEX shows a significant improvement over other methods in scenarios with severe imbalances. These findings highlight the effectiveness of leveraging a diverse set of experts to navigate the challenges presented by varying class distributions.

5.3 Results on Test-agnostic Long-tailed Classification

We assess the effectiveness of our proposed KDEX on test-agnostic long-tailed classification in this section. To simulate the scenario of unknown test distributions, we convert the testing dataset from a uniform distribution to long-tailed (Forward-LT) and inverse long-tailed (Backward-LT) distributions with different IFs, resulting in seven different test distributions. Subsequently, we evaluate all models trained with long-tailed class distribution on these various test distributions. In addition to the baselines previously evaluated, we replaced our hierarchical routing module with the expert aggregation module from SADE [79] for comparative analysis, which we refer to as

Table 4. Test-agnostic Long-tailed Classification Result of Different Methods on Four Different Benchmarks

Model	ENZYMES							MNIST						
	Forward-LT			Uniform	Backward-LT			Forward-LT			Uniform	Backward-LT		
	10	5	2	1	2	5	10	50	20	10	1	10	20	50
GraphSAGE	63.68	56.40	48.60	30.66	31.44	23.20	17.37	88.18	84.06	80.84	68.67	64.62	62.13	58.19
Up-sampling	41.58	43.20	42.53	32.33	34.93	30.00	25.26	82.03	76.25	73.65	64.69	62.04	58.12	56.87
CB loss	54.21	51.20	44.57	32.19	34.17	24.80	19.47	88.56	84.20	80.53	68.85	62.80	60.29	57.75
LACE loss	56.89	53.60	48.60	33.63	36.01	31.60	27.41	87.65	83.51	80.21	69.72	67.81	66.12	63.75
Augmentation	56.31	54.40	51.39	32.08	35.96	30.40	27.89	89.37	86.25	81.08	72.18	69.88	68.20	66.09
GraphCL	63.21	57.04	50.31	36.66	37.37	30.44	29.80	87.98	82.84	80.11	69.37	62.30	60.22	58.75
SupCon	63.34	56.89	51.03	37.08	35.52	29.58	28.90	88.23	83.12	80.32	69.76	62.19	61.03	58.63
RAHNet	65.31	58.33	**53.46**	**39.02**	**39.01**	35.49	31.23	88.92	86.29	81.93	75.79	70.34	68.71	66.01
KDEX-SADE	65.94	58.20	52.41	38.33	38.25	**35.60**	31.22	89.15	85.82	83.36	74.54	71.26	69.54	66.41
KDEX	**66.47**	**59.00**	53.18	38.67	38.98	**35.60**	**31.57**	**89.67**	**86.81**	**83.88**	**76.67**	**71.37**	**69.98**	**67.94**

Model	Letter-high							Letter-low						
	Forward-LT			Uniform	Backward-LT			Forward-LT			Uniform	Backward-LT		
	10	5	2	1	2	5	10	10	5	2	1	2	5	10
GraphSAGE	73.62	62.40	53.53	51.06	47.33	43.31	34.94	92.52	91.72	89.70	86.00	86.73	87.67	87.47
Up-sampling	57.36	54.43	54.75	53.62	51.17	51.12	44.39	93.18	92.03	90.88	88.48	88.78	88.12	87.47
CB loss	65.93	60.15	55.72	53.76	53.10	53.38	45.49	92.96	92.48	91.40	88.01	88.17	87.72	87.13
LACE loss	71.63	63.60	59.65	55.62	56.68	54.88	49.67	92.53	91.63	90.87	88.81	89.50	88.67	88.54
Augmentation	69.01	63.75	61.22	57.18	56.99	54.43	48.57	93.72	92.58	91.82	88.32	89.61	89.28	88.76
GraphCL	68.16	65.68	63.99	57.34	59.37	60.26	54.78	92.69	91.03	90.70	89.28	89.50	88.60	88.47
SupCon	68.05	65.23	63.56	57.29	59.41	60.30	54.76	92.58	91.33	90.49	89.12	89.34	88.71	88.29
RAHNet	69.13	65.98	66.30	59.79	59.27	58.71	57.03	93.03	92.87	92.23	90.19	90.12	89.84	89.50
KDEX-SADE	69.01	66.06	65.71	61.60	61.83	60.55	58.85	93.42	92.93	92.13	90.18	90.04	89.97	89.23
KDEX	**73.84**	**71.57**	**70.13**	**63.53**	**62.69**	**61.17**	**59.39**	**93.84**	**93.23**	**92.40**	**91.47**	**90.65**	**90.41**	**89.76**

The top performances are highlighted in bold.

KDEX-SADE. The results on four benchmarks with various test class distributions are presented in Table 4. Based on the table, we can draw several conclusions.

First of all, GraphSAGE outperforms several long-tailed learning baselines on most forward long-tailed test distributions owing to the Softmax activation it used, which naturally simulates the long-tailed training distribution. Nevertheless, it struggles to adapt to other distributions, resulting in poor performance on the uniform and inverse long-tailed distributions.

Moreover, existing long-tailed learning approaches cannot adjust to various test class distributions effectively. The techniques of up-sampling and loss re-weighting mainly concentrate on acquiring a balanced classifier that performs well on uniform distributions. Nonetheless, the classifier learned using these techniques only specializes in uniform distributions, which can cause a decline in performance when dealing with other test distributions.

Finally, our KDEX has demonstrated its ability to handle diverse test distributions. KDEX utilizes several knowledge-diverse experts, each specializing in a particular class distribution. Furthermore, KDEX surpasses KDEX-SADE by employing a hierarchical router to combine the outputs from various experts rather than utilizing soft aggregation approach. This result indicates that adaptively excluding redundant experts from the classification process aids in mitigating the adverse impacts of noise introduced by less effective expert models. We highlight that the superiority of KDEX becomes even more apparent as test distributions become more skewed, highlighting its robustness in handling imbalanced test distributions.

Table 5. Comparison with Several Variants for Ablation Study

	ME	MA	TA	KEL	ENZYMES Accuracy	ENZYMES Δ	Letter-high Accuracy	Letter-high Δ	COIL-DEL Accuracy	COIL-DEL Δ
M_1	✓				32.00 ± 1.82	-	54.25 ± 1.67	-	40.60 ± 1.41	-
M_2	✓	✓			36.50 ± 1.60	+4.50	58.08 ± 1.65	+3.83	45.88 ± 1.37	+5.28
M_3	✓	✓	✓		37.83 ± 1.57	+5.83	60.09 ± 1.64	+5.84	46.88 ± 1.30	+6.28
M_4	✓			✓	35.50 ± 1.43	+3.50	58.45 ± 1.71	+4.20	44.60 ± 1.34	+4.00
M_5	✓	✓	✓	✓	**38.67 ± 1.54**	**+6.67**	**63.80 ± 1.64**	**+9.55**	**47.48 ± 1.28**	**+6.88**

The top performances are highlighted in bold. KEL, knowledge-diverse experts learning; ME, model ensemble; MA, memory augmentation; TA, topological augmentation.

5.4 Ablation Study

We carry out an ablation study to evaluate the effectiveness of the major components in our KDEX. Specifically, we investigate the impact of ME, MA, TA, and knowledge-diverse experts learning modules on the overall classification performance. To this end, we gradually add each of the four key components to the baseline model and report the classification accuracy and performance gain (Δ) for each variant on three datasets, namely ENZYMES, Letter-high, and COIL-DEL. The results are presented in Table 5. The variants include: (1) M_1, which is the baseline model with only the ME technique; (2) M_2, which combines model ensembling with MA; (3) M_3, which further integrates TA to form the complete dynamic MA module; (4) M_4, which studies the impact of knowledge-diverse experts by excluding dynamic MA; and (5) M_5, which includes all four additional components and forms the complete KDEX.

The results indicate that every supplementary component positively impacts the model's classification performance. Specifically, M_2 brings significant improvement upon the baseline model by incorporating MA, which validates the effectiveness of MA in transferring knowledge from data-abundant classes to data-insufficient classes. Adding TA in M_3 leads to further improvement, indicating that it improves the generalization ability to unseen test graphs. In M_4, we exclude the influence of dynamic MA and study the impact of knowledge-diverse experts, which shows that fusing the decisions of knowledge-diverse experts contributes to the adeptness of the model in handling different classes. Finally, incorporating all four components in M_5 achieves the highest accuracy on all three datasets. Overall, our ablation study suggests that each of the additional components can enhance the performance of KDEX, and the combination of all four components leads to the best results.

5.5 Hyper-parameter Sensitivity Study

We perform various experiments to evaluate how varying settings of different hyper-parameters influence the model's performance in both standard and test-agnostic long-tailed classification scenarios.

5.5.1 Influence of Different Hyper-parameters in Standard Long-tailed Classification. We investigate the sensitivity of three hyper-parameters in the standard long-tailed classification task, including the inverse cross-entropy weight λ, number of prototypes per class, and augmentation ratio δ. The results are shown in Figure 4. Figure 4(a) illustrates the performance of our model across different λ settings on two datasets. We observed that increasing λ from 0.5 to 1.0 improved classification accuracy on both datasets. However, setting λ to an excessively large value may limit accuracy gains due to overemphasizing the inverse long-tailed distribution. Figure 4(b) shows that increasing the number of prototypes per class results in slight performance improvements.

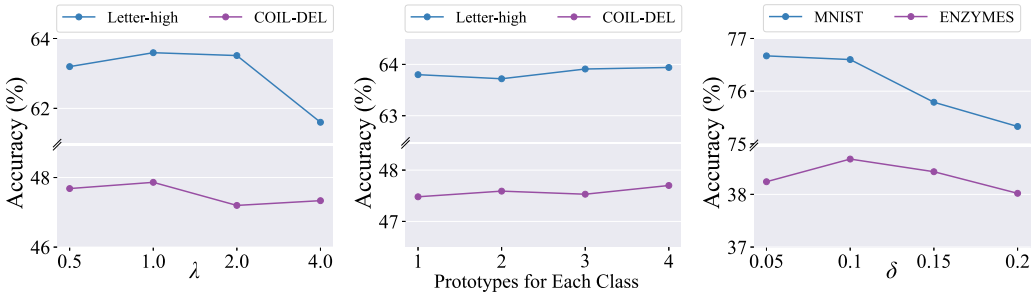


Fig. 4. Sensitivity analysis of multiple hyper-parameters: (a) Inverse cross-entropy weight λ on Letter-high and COIL-DEL datasets; (b) number of prototypes per class on Letter-high and COIL-DEL datasets; (c) augmentation ratio δ on MNIST and ENZYMES datasets.

Table 6. Test-agnostic Long-tailed Classification Result w.r.t Different Settings of Hyper-parameter λ and γ

λ	Letter-high							γ	Letter-high						
	Forward-LT			Uniform	Backward-LT				Forward-LT			Uniform	Backward-LT		
	10	5	2	1	2	5	10		10	5	2	1	2	5	10
0.5	72.96	70.53	68.62	63.16	61.55	59.96	59.17	0.3	73.62	71.36	**70.14**	63.42	62.60	60.95	59.30
1.0	**73.62**	**71.57**	**70.13**	**63.53**	**62.69**	61.17	**59.39**	0.5	73.62	**71.57**	70.13	**63.53**	62.69	**61.17**	**59.39**
2.0	72.96	70.68	67.74	61.93	61.38	**61.31**	57.63	0.7	**74.42**	71.51	70.05	**63.53**	62.45	60.95	59.18
4.0	72.16	68.68	65.99	61.12	59.37	60.26	54.78	0.9	73.08	71.07	70.05	63.25	**62.76**	61.02	59.26

The top performances are highlighted in bold.

However, since learning more prototypes can introduce additional computational overhead, we use one prototype per class as our default setting. Figure 4(c) demonstrates the performance with various TA ratios. It reveals that a ratio of 0.1 is appropriate for both datasets, as over-large values may compromise the original graph's structure.

5.5.2 Influence of λ and γ in Test-agnostic Classification. We further investigate the influence of inverse cross entropy weight λ and first-level router threshold γ in the task of test-agnostic long-tailed classification. Table 6 demonstrates the test-agnostic long-tailed classification results with various settings of λ and γ on the Letter-high dataset. Consistent with our previous findings, the highest test-agnostic classification accuracy is achieved when λ is set to 1.0. Notably, increasing λ appropriately can improve the effectiveness of both forward and backward long-tailed classification. These results indicate that a stronger backward expert not only enhances the model's capability to handle inverse long-tailed distribution but also leads to higher performance in forward long-tailed classification. Regarding γ, our model remains robust across different threshold settings, as changing it from 0.3 to 0.9 results in minor performance variation. This indicates that the first-level router can dynamically adapt to different threshold values through self-supervised optimization, effectively determining the engagement of additional experts.

5.6 Impact of Dynamic MA

A comparison of classification accuracy of different regions is conducted in this section, along with a case study demonstrating the role of dynamic MA in promoting knowledge transfer from head to tail classes.

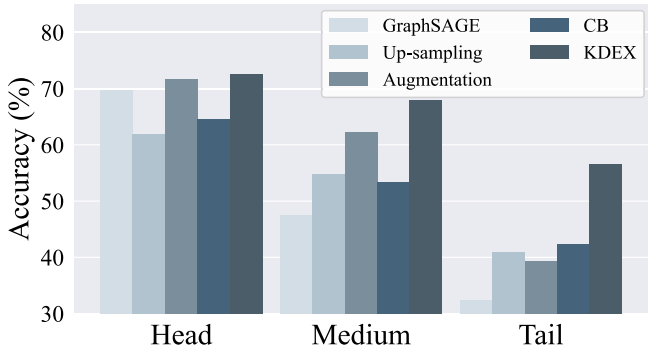


Fig. 5. Comparison of classification performance across various methods for different regions of classes within the Letter-high dataset.

5.6.1 Region Accuracy Comparison. We study the effects of dynamic MA on the accuracy of region classification. To accomplish this, we compare the performance across head, medium, and tail classes with various baselines. The results are presented in Figure 5. Our findings reveal that although all methods show good results on the data-abundant head classes, they exhibit a significant decline in performance on tail classes. For instance, on the Letter-high dataset, the GraphSAGE model shows a performance gap of approximately 40% between head and tail classes. Moreover, we found that the data re-sampling strategy, which involves up-sampling and CB loss, boosts the performance for tail classes at the expense of head class performance. Finally, our proposed KDEX framework enhances the classification accuracy of head, medium, and tail classes simultaneously by facilitating knowledge transfer from data-abundant head classes to data-scarce tail classes through dynamic memory. KDEX not only prevents under-fitting of head classes, as may occur with existing re-balancing techniques, but also boosts generalization ability by increasing within-class diversity with dynamic memory.

5.6.2 Case Study on MA. We present a case study to demonstrate how our MA module transfers knowledge in this section. The memory feature is constructed based on different class prototypes with varying attention coefficients, which are crucial to the memory feature's quality. Figure 6 showcases the three memory prototypes receiving the highest attention. The Letter-high dataset consists of graph forms of uppercase letters, with nodes representing end-points and edges corresponding to lines. To classify a sample from the tail class "Letter W," our dynamic memory assembles knowledge from letters "A," "M," and "N," which exhibit high topological correlations with the target tail class. Using the dynamic MA mechanism, our model explicitly provides additional topological and semantic enrichment to the tail class, leading to more robust representation learning.

5.7 Impact of Knowledge-diverse Experts

Here we first conduct an effectiveness and efficiency analysis of the various models in the context of a test-agnostic long-tailed classification task on a large-scale graph dataset. Furthermore, we present visualizations of the contributions of experts to demonstrate how knowledge-diverse experts enhance performance in different scenarios.

5.7.1 Efficiency Analysis. We focus on the efficiency analysis of our framework and several baselines on the large-scale dataset ogbg-ppa. We selected DeeperGCN [31] as the backbone encoder due to the dataset's scale and included the competitive baseline RAHNet in our comparison. Table 7 presents the test-agnostic classification results, as well as the inference time on the test set

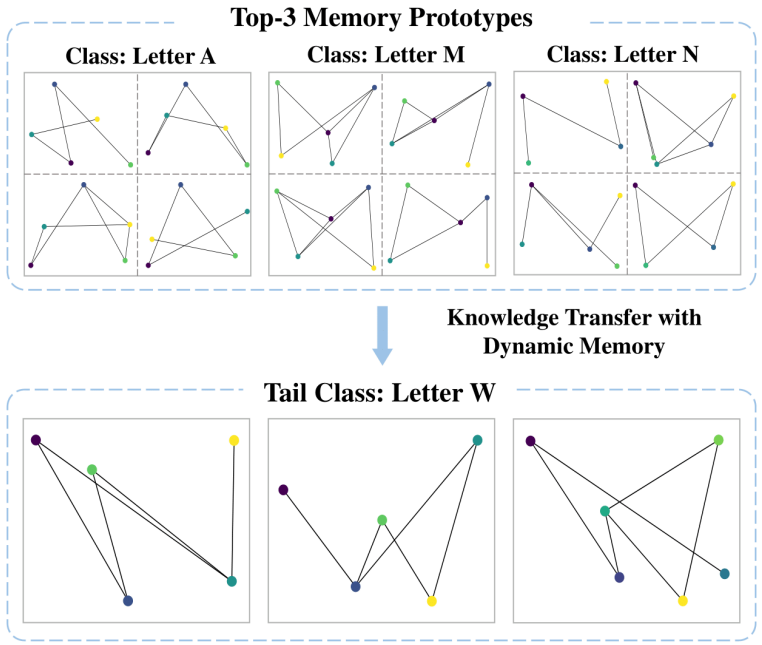


Fig. 6. A case study of top-3 memory prototypes transferred from data-rich classes to the data-poor class.

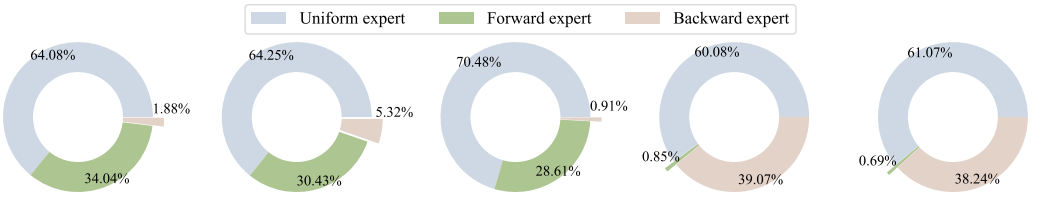
Table 7. Effectiveness and Efficiency Analysis of Different Models on ogbg-ppa Datasets with Different Test Distributions

Model	Forward-LT			Uniform	Backward-LT			Inference time (s)
	50	20	10	1	10	20	50	
DeeperGCN	65.56	59.11	56.20	50.21	49.72	46.07	44.31	**7.6674**
RAHNet	71.70	66.23	62.51	56.93	56.06	53.74	50.25	13.5495
KDEX w/o KDE and HEA	69.03	63.40	60.11	55.73	54.80	51.29	48.01	8.3258
KDEX w/o HEA	72.92	67.57	63.39	58.04	57.87	53.44	50.46	10.9492
KDEX	**73.81**	**68.46**	**64.57**	**58.13**	**59.55**	**54.02**	**51.46**	9.6053

The top performances are highlighted in bold. HEA, hierarchical expert aggregation; KDE, knowledge-diverse experts.

using a single RTX 3090 GPU. Our results show that KDEX achieves the best performance with limited computational overhead compared to vanilla DeeperGCN. KDEX is also more efficient than RAHNet, which uses a more complex retrieval framework. Among the variants of KDEX, the use of knowledge-diverse experts improves classification performance on forward and backward long-tailed test distributions, indicating that training experts in diverse areas enhance the model's adaptability to different test distributions. Additionally, hierarchical expert aggregation effectively improves KDEX's efficiency by dynamically reducing unnecessary expert engagement.

5.7.2 Visualization of Expert Engagement. We further visualize the average contribution of the uniform, forward, and backward experts in various test class distributions. When the first-level router is deactivated, the uniform expert takes 100% of the contribution. And when the first-level is activated, the uniform expert takes 50% of the contribution, while the contributions of the



(a) Foward-LT 10 (b) Foward-LT 5 (c) Uniform (d) Backward-LT 5 (e) Backward-LT 10

Fig. 7. Contribution of the uniform, forward, and backward experts in various test class distributions.

forward and backward experts are decided by the second-level router. Figure 7 shows that expert engagement varies significantly across test distributions. It is important to highlight that our hierarchical routing module offers a reduction in computational overhead compared to SADE. For example, in the uniform distribution, the uniform expert accounts for over 70% of the average contribution. This finding suggests that the first-level router reduces expert engagement to enhance efficiency when the uniform expert is capable of making confident decisions. Moreover, the forward expert takes higher contributions in the forward long-tailed test distributions, while the backward expert contributes more in the backward long-tailed test distributions. This implies that the second-level router effectively adjusts the gating weights of forward and backward experts to cater to their respective proficient classes.

6 Conclusion

This article addresses the relatively unexplored issue of long-tailed graph-level classification. GNNs tend to perform poorly on long-tailed training distributions, and existing approaches to long-tailed learning often struggle with learning good representations for the tail and handling unknown test distributions. To mitigate the above limitations, we propose a knowledge-diverse experts framework. KDEX incorporates the dynamic MA for representation learning, which facilitates the knowledge transfer from data-abundant classes to the data-scarce tail classes. Moreover, we employ a knowledge-diverse learning strategy to train experts with unique expertise for different distributions and aggregate them with hierarchical routers for better performance and efficiency. Extensive experiments on various benchmarks demonstrate that KDEX consistently outperforms the competitive baselines in classification accuracy for both standard and test-agnostic long-tailed graph classification scenarios. The primary limitation of this study is its focus on a limited range of test distributions. Future efforts aim to seek methods that obviate the need for self-supervised training on test samples for effective adaptation across more complex test distributions.

References

[1] Tara Baldacchino, Elizabeth J. Cross, Keith Worden, and Jennifer Rowson. 2016. Variational Bayesian mixture of experts models and sensitivity analysis for nonlinear dynamical systems. *Mechanical Systems and Signal Processing* 66 (2016), 178–200.

[2] Noam Barda, Gal Yona, Guy N. Rothblum, Philip Greenland, Morton Leibowitz, Ran Balicer, Eitan Bachmat, and Noa Dagan. 2021. Addressing bias in prediction models by improving subpopulation calibration. *Journal of the American Medical Informatics Association* 28, 3 (2021), 549–558.

[3] Kaidi Cao, Colin Wei, Adrien Gaidon, Nikos Arechiga, and Tengyu Ma. 2019. Learning imbalanced datasets with label-distribution-aware margin loss. In *Proceedings of the Advances in Neural Information Processing Systems*, Vol. 32.

[4] Nitesh V. Chawla. 2003. C4.5 and imbalanced data sets: investigating the effect of sampling method, probabilistic estimate, and decision tree structure. In *Proceedings of the ICML*, Vol. 3, 66.

[5] 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.

[6] Yin Cui, Menglin Jia, Tsung-Yi Lin, Yang Song, and Serge Belongie. 2019. Class-balanced loss based on effective number of samples. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 9268–9277.

[7] Yin Cui, Yang Song, Chen Sun, Andrew Howard, and Serge Belongie. 2018. Large scale fine-grained categorization and domain-specific transfer learning. In *Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 4109–4118. DOI: https://doi.org/10.1109/CVPR.2018.00432

[8] Allen B. Downey. 2001. Evidence for long-tailed distributions in the internet. In *Proceedings of the 1st ACM SIGCOMM Workshop on Internet Measurement*, 229–241.

[9] Vijay Prakash Dwivedi, Chaitanya K. Joshi, Thomas Laurent, Yoshua Bengio, and Xavier Bresson. 2020. Benchmarking graph neural networks. arXiv:2003.00982. Retrieved from https://arxiv.org/abs/2003.00982

[10] Zheng Fang, Qingqing Long, Guojie Song, and Kunqing Xie. 2021. Spatial-temporal graph ODE networks for traffic flow forecasting. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 364–373.

[11] William Fedus, Barret Zoph, and Noam Shazeer. 2022. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research* 23, 120 (2022), 1–39. Retrieved from http://jmlr.org/papers/v23/21-0998.html

[12] Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. 2017. Neural message passing for quantum chemistry. In *Proceedings of the International Conference on Machine Learning*. PMLR, 1263–1272.

[13] Hao Guo and Song Wang. 2021. Long-tailed multi-label visual recognition by collaborative training on uniform and re-balanced samplings. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 15089–15098.

[14] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *Proceedings of the Advances in Neural Information Processing Systems,* Vol. 30.

[15] Fenyu Hu, W. Liping, L. Qiang, Shu Wu, Liang Wang, and Tieniu Tan. 2022. GraphDIVE: Graph classification by mixture of diverse experts. In *Proceedings of the 31st International Joint Conference on Artificial Intelligence*.

[16] Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. 2020. Open graph benchmark: Datasets for machine learning on graphs. arXiv:2005.00687. Retrieved from https://arxiv.org/abs/2005.00687

[17] Zhenhua Huang, Yinhao Tang, and Yunwen Chen. 2022. A graph neural network-based node classification model on class-imbalanced graph data. *Knowledge-Based Systems* 244 (2022), 108538.

[18] Robert A. Jacobs, Michael I. Jordan, Steven J. Nowlan, and Geoffrey E. Hinton. 1991. Adaptive mixtures of local experts. *Neural Computation* 3, 1 (1991), 79–87. DOI: https://doi.org/10.1162/neco.1991.3.1.79

[19] Alistair E. W. Johnson, David J. Stone, Leo A. Celi, and Tom J. Pollard. 2018. The MIMIC code repository: Enabling reproducibility in critical care research. *Journal of the American Medical Informatics Association* 25, 1 (2018), 32–39.

[20] Michael I. Jordan and Robert A. Jacobs. 1994. Hierarchical mixtures of experts and the EM algorithm. *Neural Computation* 6, 2 (1994), 181–214. DOI: https://doi.org/10.1162/neco.1994.6.2.181

[21] Wei Ju, Zhengyang Mao, Siyu Yi, Yifang Qin, Yiyang Gu, Zhiping Xiao, Yifan Wang, Xiao Luo, and Ming Zhang. 2024. Hypergraph-enhanced dual semi-supervised graph classification. arXiv:2405.04773. Retrieved from https://arxiv.org/abs/2405.04773

[22] Wei Ju, Yifang Qin, Ziyue Qiao, Xiao Luo, Yifan Wang, Yanjie Fu, and Ming Zhang. 2022. Kernel-based substructure exploration for next POI recommendation. In *Proceedings of the IEEE International Conference on Data Mining (ICDM '22)*. IEEE, 221–230.

[23] Wei Ju, Siyu Yi, Yifan Wang, Qingqing Long, Junyu Luo, Zhiping Xiao, and Ming Zhang. 2024. A survey of data-efficient graph learning. arXiv:2402.00447. Retrieved from https://arxiv.org/abs/2402.00447

[24] Wei Ju, Siyu Yi, Yifan Wang, Zhiping Xiao, Zhengyang Mao, Hourun Li, Yiyang Gu, Yifang Qin, Nan Yin, Senzhang Wang, et al. 2024. A survey of graph neural networks in real world: Imbalance, noise, privacy and OOD challenges. arXiv:2403.04468. Retrieved from https://arxiv.org/abs/2403.04468

[25] Wei Ju, Yusheng Zhao, Yifang Qin, Siyu Yi, Jingyang Yuan, Zhiping Xiao, Xiao Luo, Xiting Yan, and Ming Zhang. 2024. Cool: A conjoint perspective on spatio-temporal graph neural network for traffic forecasting. *Information Fusion* 107 (2024), 102341.

[26] Hisashi Kashima, Koji Tsuda, and Akihiro Inokuchi. 2003. Marginalized kernels between labeled graphs. In *Proceedings of International Conference on Machine Learning*, 321–328.

[27] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. 2020. Supervised contrastive learning. In *Proceedings of the Advances in Neural Information Processing Systems,* Vol. 33, 18661–18673.

[28] Thomas N. Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. arXiv:1609.02907. Retrieved from https://arxiv.org/abs/1609.02907

[29] Junhyun Lee, Inyeop Lee, and Jaewoo Kang. 2019. Self-attention graph pooling. In *Proceedings of the International Conference on Machine Learning*. PMLR, 3734–3743.

[30] Bolian Li, Zongbo Han, Haining Li, Huazhu Fu, and Changqing Zhang. 2022. Trustworthy long-tailed classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 6970–6979.

[31] Guohao Li, Chenxin Xiong, Ali Thabet, and Bernard Ghanem. 2020. DeeperGCN: All you need to train deeper GCNs. arXiv:2006.07739. Retrieved from https://arxiv.org/abs/2006.07739

[32] Jia Li, Yongfeng Huang, Heng Chang, and Yu Rong. 2022. Semi-supervised hierarchical graph classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45, 5 (2022), 6265–6276.

[33] Shijun Li, Wenqiang Lei, Qingyun Wu, Xiangnan He, Peng Jiang, and Tat-Seng Chua. 2021. Seamlessly unifying attributes and items: Conversational recommendation for cold-start users. *ACM Transactions on Information Systems* 39, 4 (Aug. 2021), Article 40, 29 pages. DOI: https://doi.org/10.1145/3446427

[34] Ziwei Liu, Zhongqi Miao, Xiaohang Zhan, Jiayun Wang, Boqing Gong, and Stella X. Yu. 2019. Large-scale long-tailed recognition in an open world. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2537–2546.

[35] Zemin Liu, Trung-Kien Nguyen, and Yuan Fang. 2021. Tail-GNN: Tail-node graph neural networks. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 1109–1119.

[36] Qingqing Long, Yuchen Yan, Wentao Cui, Wei Ju, Zhihong Zhu, Yuanchun Zhou, Xuezhi Wang, and Meng Xiao. 2024. MOAT: Graph prompting for 3D molecular graphs. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, 1586–1596.

[37] Dongsheng Luo, Wei Cheng, Wenchao Yu, Bo Zong, Jingchao Ni, Haifeng Chen, and Xiang Zhang. 2021. Learning to drop: Robust graph neural network via topological denoising. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, 779–787.

[38] Junyu Luo, Yiyang Gu, Xiao Luo, Wei Ju, Zhiping Xiao, Yusheng Zhao, Jingyang Yuan, and Ming Zhang. 2024. GALA: Graph diffusion-based alignment with Jigsaw for source-free domain adaptation. *IEEE Transactions on Pattern Analysis & Machine Intelligence* 01 (2024), 1–14.

[39] Junyu Luo, Zhiping Xiao, Yifan Wang, Xiao Luo, Jingyang Yuan, Wei Ju, Langechuan Liu, and Ming Zhang. 2024. Rank and align: Towards effective source-free graph domain adaptation. arXiv:2408.12185. Retrieved from https://arxiv.org/abs/2408.12185

[40] Xiao Luo, Yiyang Gu, Huiyu Jiang, Hang Zhou, Jinsheng Huang, Wei Ju, Zhiping Xiao, Ming Zhang, and Yizhou Sun. [2024]. PGODE: Towards high-quality system dynamics modeling. In *Proceedings of the 41st International Conference on Machine Learning*.

[41] Xiao Luo, Jingyang Yuan, Zijie Huang, Huiyu Jiang, Yifang Qin, Wei Ju, Ming Zhang, and Yizhou Sun. 2023. Hope: High-order graph ODE for modeling interacting dynamics. In *Proceedings of the International Conference on Machine Learning*. PMLR, 23124–23139.

[42] Xiao Luo, Yusheng Zhao, Zhengyang Mao, Yifang Qin, Wei Ju, Ming Zhang, and Yizhou Sun. 2023. RIGNN: A rationale perspective for semi-supervised open-world graph classification. *Transactions on Machine Learning Research* (2023).

[43] Xiao Luo, Yusheng Zhao, Yifang Qin, Wei Ju, and Ming Zhang. 2023. Towards semi-supervised universal graph classification. *IEEE Transactions on Knowledge and Data Engineering* 36, 1 (2023), 416–428.

[44] Yanbiao Ma, Licheng Jiao, Fang Liu, Shuyuan Yang, Xu Liu, and Puhua Chen. 2023. Feature distribution representation learning based on knowledge transfer for long-tailed classification. *IEEE Transactions on Multimedia* 26 (2023), 2772–2784.

[45] Zhengyang Mao, Wei Ju, Yifang Qin, Xiao Luo, and Ming Zhang. 2023. RAHNet: Retrieval augmented hybrid network for long-tailed graph classification. In *Proceedings of the 31st ACM International Conference on Multimedia*, 3817–3826.

[46] Aditya Krishna Menon, Sadeep Jayasumana, Ankit Singh Rawat, Himanshu Jain, Andreas Veit, and Sanjiv Kumar. 2020. Long-tail learning via logit adjustment. arXiv:2007.07314. Retrieved from https://arxiv.org/abs/2007.07314

[47] Marion Neumann, Roman Garnett, Christian Bauckhage, and Kristian Kersting. 2016. Propagation kernels: Efficient graph kernels from propagated information. *Machine Learning* 102, 2 (2016), 209–245.

[48] Mark E. J. Newman. 2005. Power laws, Pareto distributions and Zipf's law. *Contemporary Physics* 46, 5 (2005), 323–351.

[49] Tianhao Qi, Hongtao Xie, Pandeng Li, Jiannan Ge, and Yongdong Zhang. 2023. Balanced classification: A unified framework for long-tailed object detection. *IEEE Transactions on Multimedia* 26 (2023), 3088–3101.

[50] Yifang Qin, Wei Ju, Hongjun Wu, Xiao Luo, and Ming Zhang. 2024. Learning graph ode for continuous-time sequential recommendation. *IEEE Transactions on Knowledge and Data Engineering* 36 (2024), 3224–3236.

[51] Yifang Qin, Hongjun Wu, Wei Ju, Xiao Luo, and Ming Zhang. 2023. A diffusion model for POI recommendation. *ACM Transactions on Information Systems* 42, 2 (2023), 1–27.

[52] Liang Qu, Huaisheng Zhu, Ruiqi Zheng, Yuhui Shi, and Hongzhi Yin. 2021. ImGAGN: Imbalanced network embedding via generative adversarial graph networks. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 1390–1398.

[53] Kaspar Riesen and Horst Bunke. 2008. IAM graph database repository for graph based pattern recognition and machine learning. In *Proceedings of the Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR '08) and Structural and Syntactic Pattern Recognition (SSPR '08)*. Springer, 287–297.

[54] Ida Schomburg, Antje Chang, Christian Ebeling, Marion Gremse, Christian Heldt, Gregor Huhn, and Dietmar Schomburg. 2004. BRENDA, the enzyme database: Updates and major new developments. *Nucleic Acids Research* 32, suppl_1 (2004), D431–D433.

[55] Noam Shazeer, *Azalia Mirhoseini, *Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. 2017. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *Proceedings of the International Conference on Learning Representations*. Retrieved from https://openreview.net/forum?id=B1ckMDqlg

[56] Nino Shervashidze, Pascal Schweitzer, Erik Jan Van Leeuwen, Kurt Mehlhorn, and Karsten M. Borgwardt. 2011. Weisfeiler-lehman graph kernels. *Journal of Machine Learning Research* 12, 9 (2011), 2539–2561.

[57] Nino Shervashidze, S. V. N. Vishwanathan, Tobias Petri, Kurt Mehlhorn, and Karsten Borgwardt. 2009. Efficient graphlet kernels for large graph comparison. In *Proceedings of International Conference on Artificial Intelligence and Statistics*, 488–495.

[58] Jaeyun Song, Joonhyung Park, and Eunho Yang. 2022. TAM: Topology-aware margin loss for class-imbalanced node classification. In *Proceedings of the International Conference on Machine Learning*. PMLR, 20369–20383.

[59] Yongduo Sui, Xiang Wang, Jiancan Wu, Min Lin, Xiangnan He, and Tat-Seng Chua. 2022. Causal attention for interpretable and generalizable graph classification. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 1696–1705.

[60] Jingru Tan, Changbao Wang, Buyu Li, Quanquan Li, Wanli Ouyang, Changqing Yin, and Junjie Yan. 2020. Equalization loss for long-tailed object recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 11662–11671.

[61] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph attention networks. In *Proceedings of the International Conference on Learning Representations*.

[62] 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*, 3045–3056.

[63] Xudong Wang, Long Lian, Zhongqi Miao, Ziwei Liu, and Stella X. Yu. 2020. Long-tailed recognition by routing diverse distribution-aware experts. arXiv:2010.01809. Retrieved from https://arxiv.org/abs/2010.01809

[64] Yifan Wang, Xiao Luo, Chong Chen, Xian-Sheng Hua, Ming Zhang, and Wei Ju. 2024. DisenSemi: Semi-supervised graph classification via disentangled representation learning. *IEEE Transactions on Neural Networks and Learning Systems* (2024).

[65] Yu Wang, Yuying Zhao, Neil Shah, and Tyler Derr. 2022. Imbalanced graph classification via graph-of-graph neural networks. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, 2067–2076.

[66] Shirley Wu, Kaidi Cao, Bruno Ribeiro, James Zou, and Jure Leskovec. 2023. Graphmetro: Mitigating complex distribution shifts in gnns via mixture of aligned experts. arXiv:2312.04693. Retrieved from https://arxiv.org/abs/2312.04693

[67] Zhengzhuo Xu, Zenghao Chai, Chengyin Xu, Chun Yuan, and Haiqin Yang. 2023. Towards effective collaborative learning in long-tailed recognition. *IEEE Transactions on Multimedia* 26 (2023), 3754–3764.

[68] Pinar Yanardag and S. V .N. Vishwanathan. 2015. Deep graph kernels. In *Proceedings of the 21st ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1365–1374.

[69] Junwei Yang, Hanwen Xu, Srbuhi Mirzoyan, Tong Chen, Zixuan Liu, Wei Ju, Luchen Liu, Ming Zhang, and Sheng Wang. 2023. Poisoning scientific knowledge using large language models. *bioRxiv* (2023). DOI: https://doi.org/10.1101/2023.11.06.565928

[70] 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* 9, 6 (2023), 1683–1696.

[71] Zhitao Ying, Jiaxuan You, Christopher Morris, Xiang Ren, Will Hamilton, and Jure Leskovec. 2018. Hierarchical graph representation learning with differentiable pooling. In *Proceedings of the Advances in Neural Information Processing Systems,* Vol. 31.

[72] Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. 2020. Graph contrastive learning with augmentations. In *Proceedings of the Advances in Neural Information Processing Systems,* Vol. 33, 5812–5823.

[73] Shuo Yu, Huafei Huang, Minh N. Dao, and Feng Xia. 2022. Graph augmentation learning. In *Companion Proceedings of the Web Conference 2022*, 1063–1072.

[74] Jingyang Yuan, Xiao Luo, Yifang Qin, Zhengyang Mao, Wei Ju, and Ming Zhang. 2023. Alex: Towards effective graph transfer learning with noisy labels. In *Proceedings of the 31st ACM International Conference on Multimedia*, 3647–3656.

[75] Jingyang Yuan, Gongbo Sun, Zhiping Xiao, Hang Zhou, Xiao Luo, Junyu Luo, Yusheng Zhao, Wei Ju, and Ming Zhang. 2024. EGODE: An event-attended graph ODE framework for modeling rigid dynamics. In *Proceedings of the 38th Annual Conference on Neural Information Processing Systems*.

[76] Sukwon Yun, Kibum Kim, Kanghoon Yoon, and Chanyoung Park. 2022. LTE4G: Long-tail experts for graph neural networks. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, 2434–2443.

[77] Chunhui Zhang, Chao Huang, Yijun Tian, Qianlong Wen, Zhongyu Ouyang, Youhuan Li, Yanfang Ye, and Chuxu Zhang. 2022. Diving into unified data-model sparsity for class-imbalanced graph representation learning. arXiv:2210.00162. Retrieved from https://arxiv.org/abs/2210.00162

[78] Han Zhang, Yiding Li, and Xuelong Li. 2023. Constrained bipartite graph learning for imbalanced multi-modal retrieval. *IEEE Transactions on Multimedia* 26 (2023), 4502–4514.

[79] Yifan Zhang, Bryan Hooi, Lanqing Hong, and Jiashi Feng. 2022. Self-supervised aggregation of diverse experts for test-agnostic long-tailed recognition. In *Proceedings of the Advances in Neural Information Processing Systems,* Vol. 35, 34077–34090.

[80] Yifan Zhang, Bingyi Kang, Bryan Hooi, Shuicheng Yan, and Jiashi Feng. 2021. Deep long-tailed learning: A survey. arXiv:2110.04596. Retrieved from https://arxiv.org/abs/2110.04596

[81] Tianxiang Zhao, Xiang Zhang, and Suhang Wang. 2021. Graphsmote: Imbalanced node classification on graphs with graph neural networks. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, 833–841.

[82] Yusheng Zhao, Xiao Luo, Wei Ju, Chong Chen, Xian-Sheng Hua, and Ming Zhang. 2023. Dynamic hypergraph structure learning for traffic flow forecasting. In *Proceedings of the IEEE 39th International Conference on Data Engineering (ICDE '23)*. IEEE, 2303–2316.

[83] Cheng Zheng, Bo Zong, Wei Cheng, Dongjin Song, Jingchao Ni, Wenchao Yu, Haifeng Chen, and Wei Wang. 2020. Robust graph representation learning via neural sparsification. In *Proceedings of the International Conference on Machine Learning*. PMLR, 11458–11468.

[84] Boyan Zhou, Quan Cui, Xiu-Shen Wei, and Zhao-Min Chen. 2020. BBN: Bilateral-branch network with cumulative learning for long-tailed visual recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 9719–9728.

Received 24 May 2024; revised 19 August 2024; accepted 10 November 2024