DEER: Distribution Divergence-Based Graph Contrast for Partial Label Learning on Graphs

Yiyang Gu, Zihao Chen, Yifang Qin, Zhengyang Mao, Zhiping Xiao, Wei Ju, *Member, IEEE*, Chong Chen, Xian-Sheng Hua, *Fellow, IEEE*, Yifan Wang, Xiao Luo, and Ming Zhang, *Member, IEEE*

***Abstract*—Graph neural networks (GNNs) have emerged as powerful tools for graph classification tasks. However, contemporary graph classification methods are predominantly studied in fully supervised scenarios, while there could be label ambiguity and noise in real-world applications. In this work, we explore the weakly supervised problem of partial label learning on graphs, where each graph sample is assigned a collection of candidate labels. A novel method called Distribution Divergence-based Graph Contrast (DEER) is proposed to address this issue. At the heart of our DEER is to measure the divergence among the underlying semantic distributions in the hidden space and this metric enables the identification of accurate positive graph pairs for effective graph contrastive learning. Specifically, we generate graph representations of augmented graph views that retain semantics and can be regarded as samples from the underlying semantic distributions. We employ a non-parametric metric to measure distribution divergence, which is then combined with pseudo-labeling to generate unbiased and target-oriented graph pairs. Furthermore, we introduce a label-correction method to eliminate noisy candidate labels, updating target labels using posterior distributions in a soft manner. Comprehensive experiments on various benchmarks demonstrate the superiority of our DEER in different settings compared to a range of state-of-the-art baselines.**

***Index Terms*—Graph classification, partial label learning, contrastive learning, distribution divergence.**

Manuscript received 26 January 2024; revised 26 April 2024; accepted 20 May 2024. Date of publication 31 May 2024; date of current version 27 March 2026. This work was supported in part by the National Natural Science Foundation of China NSFC under Grant 62306014 and Grant 62276002 and in part by the China Postdoctoral Science Foundation under Grant 2023M730057. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Bingkun Bao. *(Corresponding authors: Zhiping Xiao; Xiao Luo; Ming Zhang.)*

Yiyang Gu, Yifang Qin, Zhengyang Mao, Wei Ju, and Ming Zhang are with the State Key Laboratory for Multimedia Information Processing, School of Computer Science, PKU-Anker LLM Lab, Peking University, Beijing 100871, China (e-mail: yiyanggu@pku.edu.cn; qinyifang@pku.edu.cn; zhengyang.mao@stu.pku.edu.cn; juwei@pku.edu.cn; mzhang_cs@pku.edu.cn).

Zihao Chen is with the School of Mathematical Sciences, Peking University, Beijing 100871, China (e-mail: g.e.challenger@pku.edu.cn).

Zhiping Xiao and Xiao Luo are with the Department of Computer Science, University of California, Los Angeles CA 90095 USA (e-mail: patricia.xiao@cs.ucla.edu; xiaoluo@cs.ucla.edu).

Chong Chen and Xian-Sheng Hua are with the Terminus Group, Beijing 100027, China (e-mail: chenchong.cz@gmail.com; huaxiansheng@gmail.com).

Yifan Wang is with the School of Information Technology and Management, University of International Business and Economics, Beijing 100029, China (e-mail: yifanwang@uibe.edu.cn).

Digital Object Identifier 10.1109/TMM.2024.3408038

I. Introduction

GRAPH-STRUCTURED data is pervasive on the Internet and has raised widespread research interest in data mining and multimedia communities [1], [2], [3], [4], [5]. Graph classification that endeavors to determine the property of the entire graph is a fundamental problem in machine learning with applications in text mining [6], [7] and social network analysis [8], [9]. Currently, graph neural networks (GNNs) [10], [11], [12], [13], [14] have demonstrated superior performance for this problem. These methods typically begin with embedding topological knowledge based on the neighborhood propagation mechanism [15], which generates effective node representations followed by various graph pooling operations [16], [17], [18], [19] for graph-level representations. These graph-level representations are then fed into a classifier to generate the final label prediction.

Although numerous graph classification methods have been proposed, the majority of them are trained with end-to-end supervision. However, obtaining precise graph labels is prohibitively costly and time-consuming. For example, costly density functional theory (DFT) [20], [21] calculations are required to identify the properties of chemical compounds. A viable option is utilizing different automated devices and/or experts to facilitate the annotation process. But this process could introduce label ambiguity and noise, due to their potentially different comments. Those noisy and ambiguous supervised signals would bring challenges when optimizing neural networks. An example can be found in Fig. 1. Towards this end, in this paper, we investigate a practical weakly-supervised learning problem of partial label learning on graphs, which assigns ambiguous candidate sets rather than a single ground-truth label to each training graph.

Graph contrastive learning has demonstrated powerful capability in a variety of graph machine problems, which learns efficient graph representations by comparing representations of views augmented from the same graph with those from different graphs. Here, we seek to address the problem of partial label learning on graphs by leveraging the strength of graph contrastive learning. However, we face the following challenges: (1) **How can we leverage several weakly-supervised signals to improve graph contrastive learning?** By means of graph augmentation techniques, traditional graph contrastive learning methods generate positive pairs that could not capture the underlying semantics associated with the target categories of these

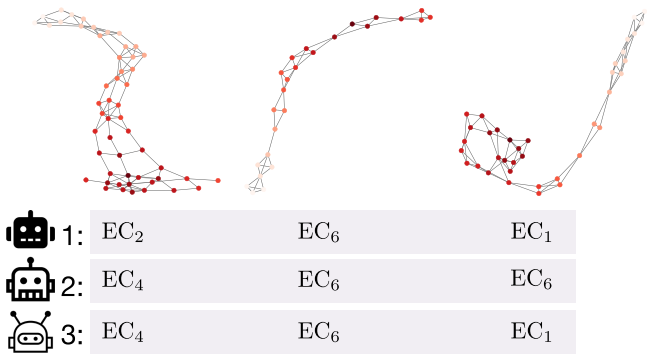


Fig. 1. Illustration of three protein graphs with potential enzyme classes (EC classes) they belong to, which are annotated by three different automated devices. These labels are noisy and ambiguous.

graphs well, and thus fail to learn target-oriented graph representations. In contrast, in our circumstance, we have access to weakly-supervised signals. Therefore, it is anticipated that positive pairs would be generated with semantics preserved utilizing these weak signals. However, it is difficult to precisely identify positive couples only based on weakly-supervised signals, since each graph is linked to a candidate label set rather than a specific label. To tackle this challenge, we try to explore the semantic distribution of graphs in the hidden space and leverage distribution divergence to identify positive pairs for enhanced graph contrastive learning, thereby facilitating the learning of target-oriented graph representations. (2) **How to detect and remove the noisy candidate labels to lessen their impact?** Partial label learning would involve a large number of noisy candidate labels, which can misguide the optimization process of neural networks. As a consequence, we should identify the ground truth label from the candidate pool for each training graph sample. Besides, the complexity and scarcity of graph data make it probable to provide biased and unstable predictions, which can lead to error accumulation during optimization with wrong identification. To this end, we try to leverage posterior label distributions and exponential moving average (EMA) strategy to correct graph labels in a soft manner, which can relieve overfitting and reduce potential error accumulation of wrong identification.

In this paper, we propose a novel method named Distribution Divergence-based Graph Contrast (DEER), which makes the first attempt of partial label learning on graphs. The core of DEER is to identify accurate positive graph pairs by looking into underlying semantic distributions of the graphs in the hidden space, facilitating effective graph contrastive learning. Since deep features from similar graphs should be on a high-dimensional manifold, we characterize the semantics of each graph using its underlying semantic distributions in the hidden space. Then, under the assumption that most graph augmentation procedures maintain semantics information, we produce multiple graph representations by perturbing each graph, which can be thought of as samples from its underlying distribution. The distribution divergence between underlying semantic distributions of two graphs is measured using a non-parametric metric, i.e., maximum mean discrepancy. Compared with pairwise distance in the hidden space, our proposed distribution divergence is unbiased and accurate, which can be combined with pseudo-labeling to generate target-oriented positive pairs. On this basis, we formalize a graph contrastive learning framework to learn discriminative graph representations. In addition, we provide a label-correction method to eliminate noisy candidate labels. We start with uniform prior distributions and then use posterior distributions derived from model predictions to correct noisy candidate labels. The exponential moving average technique is leveraged to produce soft labels, which can alleviate overfitting and error accumulation resulting from overconfident predictions. We conduct experiments on a variety of benchmark graph classification datasets in different settings, and the results verify the effectiveness of the proposed DEER compared with a number of rival approaches. In summary, the contributions of this paper are three-fold:

- *Problem Formulation:* We give rise to a practical problem of partial label learning on graphs, where each graph is associated with an ambiguous candidate label set. To the best of our knowledge, we are the first to formalize the problem on graphs.
- *Unified Framework:* We propose a novel method named DEER for the problem of partial label learning on graphs. DEER utilizes weak signals by exploring the semantics distribution of graphs in the hidden space to enhance the contrastive learning framework. We further tackle the challenges of noisy labels using posterior distributions to correct label noise in a soft manner.
- *Comprehensive Experiments:* Extensive experiments on a variety of benchmark graph classification datasets validate the superiority of the proposed DEER compared with competing baselines in different settings.

The related works are described in Section II. In Section III, we introduce the notations, the problem definition, and the details of our proposed framework DEER, respectively. Section IV provides multifaceted experimental results including performance comparison, ablation studies, sensitivity analysis, and visualization. Finally, we present a comprehensive conclusion in Section V.

II. Related Work

A. Graph Classification

Graph classification is an essential problem in graph machine learning with numerous applications in social network analysis [8], [9] and chemistry [22], [23]. Early efforts attempt to learn graph similarity using graph kernels [24], [25], [26], which divide graphs into substructures. By contrast, graph neural networks (GNNs) [10], [11], [12], [13], [15] eliminate hand-crafted features and learn effective graph representations using end-to-end methods. These methods typically update node representations using the message transmission mechanism [15], and then summarize these representations to generate graph representations with embedded topological information. Graph pooling [17], [18], [27], [28] has proven to be a crucial step in the generation of graph-level representations, and numerous downsampling strategies have been proposed. For instance,

SAGPool [17] uses the self-attention mechanism to select vital nodes from graphs. Despite their promising results, existing methods rely heavily on end-to-end supervised optimization, which necessitates a large number of accurate graph labels. This paper investigates an underexplored yet practical problem of partial label learning on graphs and proposes a novel solution to alleviate the potential label-ambiguity and noise problems.

B. Partial Label Learning

Partial label learning assumes each training sample to be associated with a set of candidate labels, among which only one of them is the ground truth label. Typically, early efforts employ average-based algorithms [29], [30], [31], [32] that treat all candidate labels equally when determining the loss objective. However, incorporating all candidate labels would mislead the optimization of neural networks. To address this issue, a variety of identification-based methods [33], [34], [35], [36], [37] that endeavor to identify ground truth labels from candidate sets are proposed. For instance, PLDA [37] integrates similarity learning and prototypical learning to improve discriminative learning in the feature space. Recently, contrastive learning has been incorporated into partial label learning, which facilitates the generation of high-quality deep features. PiCO [38] learns prototypes in the hidden space for each class, guiding label disambiguation through contrastive learning. Although these approaches have demonstrated superior performance in computer vision, they have not yet been investigated in the graph learning problems. To the best of our knowledge, we are the first to investigate the problem of partial label learning on graphs. Our work is also related to the problem of multimedia tagging where user-generated tags are often noisy and incomplete in the real world, leading to a lack of high-quality learning examples [39]. To tackle the challenge of tag inaccuracy, RankVote [40], a ranking-oriented neighbor voting framework is proposed for effective tag relevance learning. CVTagger [41] integrates multimodal features and temporal dynamics through a hierarchical classification model, which enhances video tagging accuracy. Different from these methods, we try to address the problem of lacking precisely labeled graph data from a perspective of graph partial label learning, and propose a novel and effective framework DEER to reduce the impact of label noise and ambiguity.

C. Graph Contrastive Learning

Inspired by the advancements in computer vision [42], [43], [44], graph contrastive learning (GCL) [45], [46], [47], [48], [49] has become a popular topic in graph representation learning, and it can be applied to node-level and graph-level machine learning tasks. A typical GCL procedure is to generate distinct perspectives of graphs via augmentation and then maximize the similarity between their representations in comparison to negative pairs. The effectiveness of GCL has been enhanced by incorporating adversarial learning, clustering, curriculum learning, and different augmentation strategies. For example, CGC [47] uses the counterfactual mechanism to provide challenging negative pairs from original samples. By perturbing both node attributes and node embeddings, GASSL [49] generates hard positive samples using adversarial learning. GRLC [50] enhances the generalization ability of graph representation learning by maximizing the mutual information between the semantic information and the structural information of graphs and designing three effective constraints. UMGRL [51] separates the common and private representations for multiplex graphs and leverages a contrastive loss to preserve the complementarity and filter out the noise in the private information. MGCL [4] employs node-level contrastive learning across modals to refine representations of users and items and mitigate multimodal noise contamination. Despite their remarkable success, employing GCL to the problem of partial label learning on graphs remains challenging, as we have mentioned in the Introduction. In this paper, we connect partial label learning on graphs with graph contrastive learning and utilize weak signals by exploring the semantics distribution of graphs in the hidden space to enhance the contrastive learning framework.

TABLE I
SUMMARY OF KEY NOTATIONS

Notation	Description
G	A graph
$\mathcal{V}$	The node set of a graph
$\mathcal{E}$	The edge set of a graph
$\boldsymbol{x}_v$	The attribute vector of the node v
$\boldsymbol{A}$	The adjacency matrix of a graph
Y_i	The candidate label set of the graph G_i
$\hat{G}_i^{(m)}$	The m-th augmented view for the graph G_i
$\hat{\boldsymbol{z}}_i^{(m)}$	The embedding for the m-th augmented view $\hat{G}_i^{(m)}$
M	The number of augmented views for a graph
Q_j	The underlying semantics distribution of the graph G_j
$\widehat{\text{MMD}}$	The estimated maximum mean discrepancy
α	The threshold percentile for MMD-based selection
$\hat{\boldsymbol{p}}_i[j]$	The predicted probability of the i-th graph belonging to the j-th class
$\tilde{G}_i$	The original or augmented view for a graph in a mini-batch, the subscript i is re-annotated
B	The size of a mini-batch
$\boldsymbol{Y}^{pri}$	The prior label matrix
$\boldsymbol{Y}^{pos}$	The posterior label matrix
$\boldsymbol{Y}^{EMA}$	The corrected label matrix with momentum updating

III. METHODOLOGY

A. Notations and Problem Definition

We first introduce the notations used in our paper. Let $G = (\mathcal{V}, \mathcal{E})$ represent a graph with the node set $\mathcal{V}$ and the edge set $\mathcal{E}$. Each node v is associated with an attribute vector $\boldsymbol{x}_v \in \mathbb{R}^F$ where F denotes the attribute dimension. The edge set can be depicted using an adjacency matrix $\boldsymbol{A} \in \mathbb{R}^{|\mathcal{V}|\times|\mathcal{V}|}$. Table I summarizes the key notations utilized in this paper. Then, we give the problem definition of partial label learning (PLL) on graphs.

Definition 1 (Partial label learning on graphs): Let $\mathcal{Y} = \{1, 2, \cdots, C\}$ denote the label space where C denotes the number of classes. We are given a training set $\mathcal{D} = \{G_i, Y_i\}_{i=1}^n$, where each graph G_i is connected with a candidate set $Y_i \subset \mathcal{Y}$. The ground truth label y_i is in its candidate set but invisible to the learner. Our goal is to learn a graph neural network model,

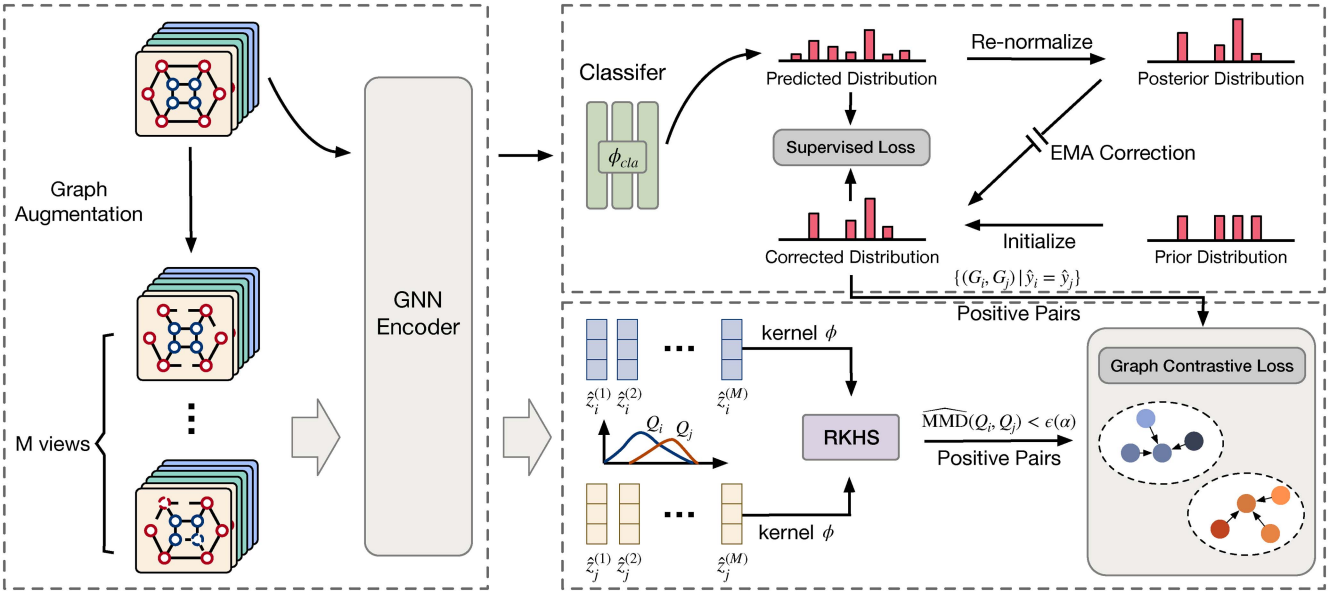


Fig. 2. Illustration of the proposed framework DEER. Our DEER first generates M augmented graphs and feed them into the GNN-based encoder to generate multiple deep features. Then, we leverage distribution divergence between two graphs, and corrected label distributions to generate unbiased and target-oriented positive pairs for effective graph contrastive learning. The labels are corrected using the exponential moving average strategy. The whole framework is optimized using both supervised learning loss and graph contrastive learning loss.

which can precisely predict the labels for graph samples in the test set.

B. Framework Overview

This paper makes the first attempt to study partial label learning on graphs, and proposes a graph contrastive learning framework named DEER. The core of our DEER is to determine accurate positive pairs by exploring underlying semantic distributions of graphs in the hidden space. In particular, we generate samples to explore semantic distributions by perturbing graphs multiple times and then utilize a non-parametric metric to measure the distribution divergence between two graphs. Then, these pairwise signals are incorporated with supervised information to decide positives for a graph contrastive learning framework, which helps learn discriminative and target-oriented graph representations. To further relieve the influence of noisy candidate labels, we start from uniform prior distributions, and utilize the exponential moving average strategy to correct noisy candidate labels with posterior distributions in a soft manner. More details can be seen in Fig. 2.

C. GNN-Based Encoder

To begin, we introduce our GNN-based encoder f_{enc}. GNNs have shown powerful capability of extracting crucial information in graphs following the scheme of message passing [15]. In particular, let $\boldsymbol{h}_v^{(l)}$ denote the representation of $v \in G$ at layer l, and the propagation rule can be formulated as:

$$\boldsymbol{h}_v^{(l)} = \text{COM}\left(\boldsymbol{h}_v^{(l-1)}, \text{AGG}\left(\left\{\boldsymbol{h}_u^{(l-1)}\right\}_{u\in\mathcal{N}(v)}\right)\right), \quad (1)$$

where $\mathcal{N}(v)$ denotes the neighbors of node v. COM and AGG denote the combination and aggregation operations, respectively. Finally, we utilize summation as our readout function, which can generate a graph-level representation $\boldsymbol{z}$. In formulation,

$$\boldsymbol{z} = f_{enc}(G) = \sum_{v\in G} \boldsymbol{h}_v^{(L)}, \quad (2)$$

where $\boldsymbol{h}_v^{(L)}$ denotes the node representation at the last layer and L denotes the number of layers. Finally, we utilize a multi-layer perception (MLP) ϕ_{cla} to map representations into label distributions, i.e.,

$$\hat{\boldsymbol{p}} = \phi_{cla}(\boldsymbol{z}), \quad (3)$$

in which $\hat{\boldsymbol{p}} \in [0,1]^C$ and ϕ_{cla} adopts a softmax activation function in the output layer.

D. Distribution Divergence for Positive Pairs

This paper aims to develop a graph contrastive learning framework for partial label learning on graphs. The most crucial part of graph contrastive learning is to determine the positives. Previous methods [45], [52] usually consider different views of each training graph, which cannot make the best of cross-graph semantics. A straightforward solution to this dilemma is to measure the pairwise distance of deep features, which could be inaccurate still due to the following reason. Note that deep features from similar graphs should lay on a high-dimensional manifold. As a consequence, when deep features are around the manifold boundary, serious biases would be introduced when computing pairwise distance. Here, we propose a novel method which leverages different graph augmentations to explore the semantic distribution

behind each graph and then measure the distribution divergence to determine positive pairs.

To begin, we first measure the underlying semantic distribution behind each graph. Since we cannot get access to accurate graph labels, we turn to graph augmentations which would not change the semantics of graphs. In particular, four common graph augmentation strategies [52] are leveraged here: (1) *Edge perturbation*, which adds or deletes several edges at random in the graph; (2) *Node dropping*, which selects nodes at random and eradicates them from the graph, along with their connected edges; (3) *Attribute masking*, which selects a subset of nodes and masks part of their attributes; (4) *Subgraph*, which leverages random walk to select a subgraph from the graph. Here, we generate M augmented views for each graph followed by the GNN-based encoder, which can be considered as samples from its underlying semantic distribution.

Then, we calculate the semantic distance between two graphs from the perspective of distributions. Since the prior of semantics distribution in the hidden space is not available, we utilize a non-parametric metric, maximum mean discrepancy (MMD) [53], [54], [55] to calculate the distribution divergence. In formulation, given a group of M augmented views for a graph sample G_i, i.e., $\{\hat{G}_i^{(1)}, \ldots, \hat{G}_i^{(M)}\}$, we generate their embeddings in the hidden space using $\hat{\boldsymbol{z}}_i^{(m)} = f_{enc}(\hat{G}_i^{(m)})$, which results in M deep features viewed as samples from the semantics distribution Q_i. Similarly, for graph G_j, we can also sample M deep features from its underlying semantics distribution Q_j. Then, the MMD between two distributions is formulated as:

$$\mathrm{MMD}(Q_i, Q_j) = \sup_{\|f\|_{\mathcal{H}} \leq 1} \left(\mathbf{E}_{\hat{\boldsymbol{z}}_i \sim Q_i}[f(\hat{\boldsymbol{z}}_i)] - \mathbf{E}_{\hat{\boldsymbol{z}}_j \sim Q_j}[f(\hat{\boldsymbol{z}}_j)] \right), \tag{4}$$

where f is a function from the unit ball in the reproducing kernel Hilbert space (RKHS) [56], [57]. This essentially measures the largest possible discrepancy between the expectations over these distributions for any function f that is bounded by 1 in the RKHS norm. The estimated MMD can be formulated as:

$$\widehat{\mathrm{MMD}}(Q_i, Q_j) = \left\| \frac{1}{M} \sum_{m=1}^{M} \phi\left(\hat{\boldsymbol{z}}_i^{(m)}\right) - \frac{1}{M} \sum_{m'=1}^{M} \phi\left(\hat{\boldsymbol{z}}_j^{(m')}\right) \right\|_{\mathcal{H}}, \tag{5}$$

where ϕ is a kernel function mapping deep features into the RKHS. The terms inside the norm represent the average mappings of samples from each distribution into the RKHS. In practice, we calculate $\widehat{\mathrm{MMD}}$ using the Gaussian kernel [58], [59], which offers a smooth and infinitely differentiable mapping, benefiting for analyzing the features in a high-dimensional space. Finally, we introduce a threshold ϵ and consider the graph pairs with the distribution divergence smaller than ϵ as positives. In formulation, the set of positives can be written as:

$$\mathcal{P} = \{(G_i, G_j) | \widehat{\mathrm{MMD}}(Q_i, Q_j) < \epsilon(\alpha)\}, \tag{6}$$

where $\epsilon(\alpha)$ is determined by the α-percentile of all pairwise $\widehat{\mathrm{MMD}}$. Moreover, we involve supervised information positive pairs, which can further distill potential false positives. In particular, we first generate pseudo-labels for each sample as follows:

$$\hat{y}_i = \mathrm{argmax}_{j \in Y_i} \hat{\boldsymbol{p}}_i[j], \tag{7}$$

where $\hat{\boldsymbol{p}}_i[j]$ denotes the probability of i-th sample belonging to j-th class. We further require that positive pairs should have the same pseudo-labels. Therefore, the distilled set of positive pairs is:

$$\hat{\mathcal{P}} = \mathcal{P} \cap \{(G_i, G_j) | \hat{y}_i = \hat{y}_j\}. \tag{8}$$

With the incorporation of distribution divergence and pseudo-labels, we can generate unbiased and target-oriented positive pairs, which would be utilized for effective graph contrastive learning.

E. Graph Contrastive Learning for Partial Label Learning

After obtaining positive samples, we can introduce our graph contrastive learning for PLL on graphs. Graph contrastive learning [45] has been proven effective in generating effective graph representations in unsupervised scenarios, which enforces positive pairs to be close compared with negative pairs. In our DEER, we adapt this technique into our weakly-supervised scenario.

In particular, given a mini-batch of $\{(G_i, Y_i)\}_{i=1}^{B}$, we involve in both original and augmented views for each graph, resulting in $2B$ graphs in total. We re-annotate these graphs as $\{(\tilde{G}_i, \tilde{Y}_i)\}_{i=1}^{2B}$ and generate positive pairs as follows:

$$\tilde{\mathcal{P}}_{bat} = \{(\tilde{G}_i, \tilde{G}_j) | (G_{k_i}, G_{k_j}) \in \hat{\mathcal{P}} \text{ or } G_{k_i} = G_{k_j}\}, \tag{9}$$

where k_i and k_j denote the original indices of $\tilde{G}_i$ and $\tilde{G}_j$. The negative samples are all the other samples besides themselves. In formulation, we introduce an MLP head $g(\cdot)$ and the loss objective for graph contrastive learning is written as:

$$\mathcal{L}_{GCL} = -\sum_{i=1}^{2B} \frac{1}{|\Pi(i)|} \sum_{j \in \Pi(i)} \log \frac{\exp\left(g(\tilde{\boldsymbol{z}}_i) \cdot g(\tilde{\boldsymbol{z}}_j)/\tau\right)}{\sum_{j' \neq i} \exp\left(g(\tilde{\boldsymbol{z}}_i) \cdot g(\tilde{\boldsymbol{z}}_{j'})/\tau\right)}, \tag{10}$$

where $\Pi(i) = \{j | (\tilde{G}_i, \tilde{G}_j) \in \tilde{\mathcal{P}}_{bat}\}$ denotes the index of positives for $\tilde{G}_i$ and τ is a temperature parameter set to 0.07 following [43].

Compared with classic graph contrastive learning, our DEER leverages both underlying semantic distributions and pseudo-labeling to generate positive samples, which can generate more discriminative and target-oriented graph representations.

F. Exponential Moving Average (EMA)-Based Label Correction

To further relieve the impact of noisy samples, we aim to identify true labels from candidate sets. Here, we start from a uniform prior distribution, and then leverage posterior distributions and an exponential moving average (EMA) strategy, to correct graph labels in a soft manner. This process can relieve overfitting and reduce potential error accumulation of wrong identification.

In detail, we first initialize the label matrix, which considers all candidate labels as equal. In particular, we have the prior

Algorithm 1: Training Algorithm of DEER

Input: Training dataset $\mathcal{D} = \{G_i, Y_i\}_{i=1}^n$; Divergence percentile α; Number of augmentation M.
Output: Parameters for f_{enc} and ϕ_{cla}.
Initialize the soft label matrix $\boldsymbol{Y}^{pri}$; // Eq. 11
Warm up the neural networks ;
while *not convergence* **do**
  Sample a mini-batch of $\{(G_i, Y_i)\}_{i=1}^B$ from $\mathcal{D}$;
  Generate M augmented views $\{\hat{G}_i^{(1)}, \cdots, \hat{G}_i^{(M)}\}$ for each graph G_i ;
  Calculate the distribution divergence $\widehat{\text{MMD}}$ among graphs in the mini-batch ; // Eq. 5
  Generate the set of positive pairs $\hat{\mathcal{P}}$; // Eq. 8
  Perform graph contrastive learning with $\hat{\mathcal{P}}$ to get loss objective $\mathcal{L}_{GCL}$; // Eq. 10
  Perform EMA-based label correction by updating the soft label matrix $\boldsymbol{Y}^{EMA}$; // Eq. 13
  Supervise the model with $\boldsymbol{Y}^{EMA}$ to get loss objective $\mathcal{L}_{SUP}$; // Eq. 14
  Optimization via $\mathcal{L} = \mathcal{L}_{SUP} + \mathcal{L}_{GCL}$;

matrix $\boldsymbol{Y}^{pri} \in [0,1]^{n \times C}$ as follows:

$$\boldsymbol{Y}_{ic}^{pri} = \begin{cases} 1/|Y_i|, & \text{if} \quad c \in Y_i \\ 0, & \text{otherwise.} \end{cases} \tag{11}$$

Clearly, uniform probabilities for all candidate labels are not optimal. A feasible solution is to update the label matrix using these predicted distributions. Here, we clear probabilities outside the candidate set and re-normalize them for posterior distributions as follows:

$$\boldsymbol{Y}_{ic}^{pos} = \begin{cases} \frac{\hat{\boldsymbol{p}}_i[c]}{\sum_{c' \in Y_i} \hat{\boldsymbol{p}}_i[c']} & \text{if} \quad c \in Y_i \\ 0 & \text{otherwise,} \end{cases} \tag{12}$$

where $\boldsymbol{Y}_{ic}^{pos}$ denotes the posterior probability of the graph sample G_i belonging to class c.

To reduce the impact of the wrong prediction, we employ the exponential moving average (EMA) strategy as follows:

$$\begin{aligned} &\boldsymbol{Y}^{EMA} \leftarrow \boldsymbol{Y}^{pri}, \\ &\boldsymbol{Y}^{EMA} \leftarrow \eta \boldsymbol{Y}^{EMA} + (1-\eta)\boldsymbol{Y}^{pos}, \end{aligned} \tag{13}$$

where $\boldsymbol{Y}^{EMA}$ denotes the corrected label matrix with momentum updating and η is a momentum coefficient set to 0.99 following [43]. Note that we employ a soft label matrix updated by the EMA strategy, which not only acts like label smoothing to relieve overfitting, but also reduces the impact of error accumulation with false pseudo-labels. The corrected label distributions can be utilized to supervise the optimization of the neural network using the least squared classification loss as follows:

$$\mathcal{L}_{SUP} = \frac{1}{B} \sum_{i=1}^{B} ||\hat{\boldsymbol{p}}_i - \boldsymbol{y}_i^{EMA}||_2^2, \tag{14}$$

where $\boldsymbol{y}_i^{EMA}$ is the i-th row vector in $\boldsymbol{Y}^{EMA}$.

In summary, the final loss objective can be written as:

$$\mathcal{L} = \mathcal{L}_{SUP} + \mathcal{L}_{GCL}. \tag{15}$$

The whole framework is optimized using the mini-batch stochastic gradient descent (SGD). We first warm up the network and the learning procedure is shown in Algorithm 1.

G. Theoretical Analysis

In this section, we provide a theoretical analysis to demonstrate how DEER works. First, we show that MMD-based distribution divergence with our graph augmentation strategies can help filter out true positive pairs from similar distributions. Then, we prove that our contrastive loss for partial label learning converges to the standard limit of contrastive loss.

Definition 2 (Generalized random dot product graph [60]): We say an graph G with adjacency matrix $\boldsymbol{A} \in \{0,1\}^{n \times n}$ is a generalized random dot product graph on n vertices with distribution F_ξ, sparsity factor α_n, and latent positions $\boldsymbol{\Xi}$ if $\boldsymbol{A}$ is symmetric, and the entries $\boldsymbol{A}_{ij}$ $(1 \le i < j \le n)$ are conditionally independent given $\boldsymbol{\Xi}$ and Bernoulli random variables with

$$\mathbb{P}(\boldsymbol{A}_{ij} = 1 | \boldsymbol{\Xi}) = \alpha_n \xi_i^T \boldsymbol{I}_{p,q} \xi_j,$$

where $\boldsymbol{I}_{p,q} = \text{diag}(\boldsymbol{I}_p, -\boldsymbol{I}_q)$, $\xi_1, \ldots, \xi_n \sim F_\xi$ are i.i.d., $\boldsymbol{\Xi} = (\xi_1, \ldots, \xi_n)^T$, and F_ξ is a distribution function with support $\Omega \subset \mathbb{R}^{p+q}$ satisfying $\boldsymbol{x}^T \boldsymbol{I}_{p,q} \boldsymbol{y} \in [0,1]$ for all $\boldsymbol{x}, \boldsymbol{y} \in \Omega$. For simplicity, we write $(G, \boldsymbol{\Xi}) \sim \text{GRDPG}(F_\xi, n, \alpha_n)$.

Definition 3: Let F_ξ and F_ζ be two distributions satisfying conditions in Definition 2. We say F_ξ and F_ζ are equal up to indefinite orthogonal transformation if there exists a matrix $\boldsymbol{Q}$ satisfying $\boldsymbol{Q}\boldsymbol{I}_{p,q}\boldsymbol{Q}^T = \boldsymbol{I}_{p,q}$ such that $F_\xi = F_\zeta \circ \boldsymbol{Q}$. In this case, we write $F_\xi \simeq F_\zeta$.

Theorem 1: Suppose $(G_1, \boldsymbol{\Xi}_1) \sim \text{GRDPG}(F_1, N_1)$ and $(G_2, \boldsymbol{\Xi}_2) \sim \text{GRDPG}(F_2, N_2)$ are two independent graphs, where $\boldsymbol{\Xi}_1$ and $\boldsymbol{\Xi}_2$ are unobservable. Set $k(\cdot,\cdot)$ be the kernel function corresponding to feature map ϕ for RKHS $\mathcal{H}$. Define the mean embedding of a distribution function F as

$$\mu[F, \alpha] := \mathbb{E}_{G \sim \text{GRDPG}(F,N,\alpha)}\, k(\cdot, \boldsymbol{z}),$$

where $\boldsymbol{z} = f_{enc}(G)$. Further assume there exists an encoder function f_{enc} such that $\mu[F_1, \alpha_1] = \mu[F_2, \alpha_2]$ if and only if $F_1 \simeq F_2$ and $\alpha_1 = \alpha_2$.

If samples in (5) are α-mixing, then, for graph augmentation strategies (1) Edge perturbation, (2) Node dropping and (4) Subgraph, as $M \to \infty$,

$$\widehat{\text{MMD}}(Q_1, Q_2) \to \begin{cases} 0 & F_1 \simeq F_2; \\ c > 0 & \text{otherwise.} \end{cases} \tag{16}$$

Remark 1: In Theorem 1, we only consider graphs without node attributes for notation simplicity. But the theorem still holds under graph augmentation strategy (3) Attribute masking if the graphs have attributes $\boldsymbol{x}_v$ whose distribution only depends on $\boldsymbol{\Xi}$.

Proof: First, we consider (1) Edge perturbation. Without loss of generality, we assume each edge is deleted independently with probability $\epsilon > 0$. Formally, let $(G, \boldsymbol{\Xi}) \sim \text{GRDPG}(F, N, \alpha)$ and

δ_{ij} $(1 \leq i < j \leq n)$ are independent Bernoulli random variables with parameter $1-\epsilon$. Suppose δ_{ij} $(1 \leq i < j \leq n)$ are independent from graph G. Further, define the adjacency matrix of the permuted graph $\tilde{G}$ as $\tilde{\boldsymbol{A}}$ with $\tilde{\boldsymbol{A}}_{ij} = \boldsymbol{A}_{ij}\delta_{ij}$ $(1 \leq i < j \leq n)$. Therefore,

$$\begin{aligned}\mathbb{P}(\tilde{\boldsymbol{A}}_{ij} = 1|\boldsymbol{\Xi}) &= \mathbb{E}\left\{\mathbb{P}(\tilde{\boldsymbol{A}}_{ij} = 1|\boldsymbol{\Xi}, \delta_{ij})\right\} \\ &= (1-\epsilon)\mathbb{P}(\boldsymbol{A}_{ij} = 1|\boldsymbol{\Xi}, \delta_{ij} = 1) \\ &= (1-\epsilon)\alpha \cdot \xi_i^T \boldsymbol{I}_{p,q} \xi_j.\end{aligned}$$

Hence, we have

$$(\tilde{G}, \boldsymbol{\Xi}) \sim \text{GRDPG}(F, N, (1-\epsilon)\alpha). \tag{17}$$

Therefore, using standard theory for V-statistics [61], [62], we have

$$\begin{aligned}\widehat{\text{MMD}}(Q_1, Q_2) \to \|\mu[F_1, (1-\epsilon)\alpha] - \mu[F_2, (1-\epsilon)\alpha]\|_{\mathcal{H}}^2 \\ = \begin{cases} 0 & F_1 \simeq F_2; \\ c > 0 & \text{otherwise.} \end{cases}\end{aligned} \tag{18}$$

Similarly, we consider (2) Node dropping. We assume each node is dropped independently with probability $\epsilon > 0$. Formally, let δ_i $(1 \leq i \leq n)$ are independent Bernoulli random variables with parameter $1-\epsilon$. Suppose δ_{ij} $(1 \leq i < j \leq n)$ are independent from graph G. Further, define the adjacency matrix of the permuted graph $\tilde{G}$ as $\tilde{\boldsymbol{A}}$ with $\tilde{\boldsymbol{A}}_{ij} = \boldsymbol{A}_{ij}\delta_i\delta_j$ $(1 \leq i < j \leq n)$. Therefore,

$$\begin{aligned}\mathbb{P}(\tilde{\boldsymbol{A}}_{ij} = 1|\boldsymbol{\Xi}) &= \mathbb{E}\left\{\mathbb{P}(\tilde{\boldsymbol{A}}_{ij} = 1|\boldsymbol{\Xi}, \delta_{ij})\right\} \\ &= (1-\epsilon)^2\mathbb{P}(\boldsymbol{A}_{ij} = 1|\boldsymbol{\Xi}, \delta_i = 1, \delta_j = 1) \\ &= (1-\epsilon)^2\alpha \cdot \xi_i^T \boldsymbol{I}_{p,q} \xi_j.\end{aligned}$$

Hence, we have

$$(\tilde{G}, \boldsymbol{\Xi}) \sim \text{GRDPG}(F, N, (1-\epsilon)^2\alpha). \tag{19}$$

So, similarly, (16) still holds.

For (4) Subgraph, we only need to marginalize out those unobserved nodes, so

$$(\tilde{G}, \boldsymbol{\Xi}) \sim \text{GRDPG}(F, \tilde{N}, \alpha). \tag{20}$$

Therefore, (16) still holds.

Theorem 2: Define

$$\tilde{\mathcal{L}}_{GCL} = \mathbb{E}_{G_i}\left\{\frac{1}{|\Pi(i)|}\sum_{j\in\Pi(i)} - \log\frac{\exp\left(g(\tilde{\boldsymbol{z}}_i)\cdot g(\tilde{\boldsymbol{z}}_j)/\tau\right)}{\sum_{j'\neq i}\exp\left(g(\tilde{\boldsymbol{z}}_i)\cdot g(\tilde{\boldsymbol{z}}_{j'})/\tau\right)}\right\} \tag{21}$$

Under the assumption of Theorem 1 and for any fixed $\tau > 0$, we have

$$\begin{aligned}\lim_{M\to\infty}\lim_{B\to\infty}\tilde{\mathcal{L}}_{GCL} - \log(2B) \\ = -\frac{1}{\tau}\mathbb{E}_{(G_1,G_2)\sim p_{pos}}\left[g(\tilde{\boldsymbol{z}}_1)\cdot g(\tilde{\boldsymbol{z}}_2)\right] \\ + \mathbb{E}_{G\sim p_{data}}\left[\log\mathbb{E}_{G^-\sim p_{data}}\left\{\exp\left(g(\tilde{\boldsymbol{z}})\cdot g(\tilde{\boldsymbol{z}}^-)/\tau\right)\right\}\right]\end{aligned} \tag{22}$$

almost surely.

Remark 2: (22) is the same as the standard asymptotic function of contrastive loss [63]. The first term can be interpreted as the alignment loss and the second term indicates the uniformity.

Proof: From (21), we know that as $B \to \infty$,

$$\tilde{\mathcal{L}}_{GCL} = \mathbb{E}_{(G_i,G_j)\sim\tilde{p}_{bat},G_{j'}\sim p_{data}}\left\{\log\frac{\sum_{j'\neq i}\exp\left(g(\tilde{\boldsymbol{z}}_i)\cdot g(\tilde{\boldsymbol{z}}_{j'})/\tau\right)}{\exp\left(g(\tilde{\boldsymbol{z}}_i)\cdot g(\tilde{\boldsymbol{z}}_j)/\tau\right)}\right\},$$

where $\tilde{p}_{bat}$ is the distribution induced by $\tilde{\mathcal{P}}_{bat}$ in (9) and p_{data} is the data distribution. So, by strong law of large numbers and the Continuous Mapping Theorem,

$$\begin{aligned}\lim_{B\to\infty}\log\left\{\frac{1}{2B}\exp\left(g(\tilde{\boldsymbol{z}}_i)\cdot g(\tilde{\boldsymbol{z}}_j)/\tau\right)\right. \\ \left. + \frac{1}{2B}\sum_{j'\neq i, j'\neq j}\exp\left(g(\tilde{\boldsymbol{z}}_i)\cdot g(\tilde{\boldsymbol{z}}_{j'})/\tau\right)\right\}\end{aligned} \tag{23}$$

$$= \log\mathbb{E}_{G_{j'}\sim p_{data}}\left\{\exp\left(g(\tilde{\boldsymbol{z}}_i)\cdot g(\tilde{\boldsymbol{z}}_{j'})/\tau\right)\right\}. \tag{24}$$

Further, from Theorem 1, we know that as $M \to \infty$, $\tilde{p}_{bat}$ weakly converges to p_{pos}, where p_{pos} are distributions induced by all true positive pairs. Hence, combining (24), we know that

$$\begin{aligned}\lim_{M\to\infty}\lim_{B\to\infty}\tilde{\mathcal{L}}_{GCL} - \log(2B) \\ = -\frac{1}{\tau}\mathbb{E}_{(G_1,G_2)\sim p_{pos}}\left[g(\tilde{\boldsymbol{z}}_1)\cdot g(\tilde{\boldsymbol{z}}_2)\right] \\ + \mathbb{E}_{G\sim p_{data}}\left[\log\mathbb{E}_{G^-\sim p_{data}}\left\{\exp\left(g(\tilde{\boldsymbol{z}})\cdot g(\tilde{\boldsymbol{z}}^-)/\tau\right)\right\}\right]\end{aligned} \tag{25}$$

almost surely.

H. Computational Efficiency Analysis

Suppose n is the number of graphs in a dataset, $|\mathcal{V}|$ and $|\mathcal{E}|$ are the average number of nodes and the average number of edges in the graphs respectively, M is the number of augmented views for each graph, and B is the batch size. The computational complexity of GraphSAGE [12] for each graph is $O(|\mathcal{E}|)$. The complexity of calculating the distribution divergence between all graph pairs in each batch is $O(B^2\,M^2)$. Besides, the computational complexity of the graph contrastive loss $\mathcal{L}_{GCL}$ and the least squared classification loss $\mathcal{L}_{SUP}$ for each batch is $O(B^2)$ and $O(B)$ respectively. Therefore, the total computational complexity of the DEER is $O(n|\mathcal{E}|) + O(\lfloor\frac{n}{B}\rfloor(B^2\,M^2 + B^2 + B)) = O(n(|\mathcal{E}| + M^2B))$. As seen in the following sensitivity analysis, the performance of the model approaches saturation when the number of augmented views for each graph M is small, thus M can be regarded as a constant term and has limited impact on the computational efficiency.

IV. Experiments

In this section, we conduct extensive experiments on various datasets to illustrate the effectiveness of the proposed DEER.

Our focus revolves around addressing several research questions (RQs) to ensure comprehensive analysis:

- *RQ 1:* How does our proposed framework DEER perform on partial label graph learning compared with other state-of-the-art methods?
- *RQ 2:* What is the effect of the key components of DEER (i.e. EMA-based label correction, graph contrastive learning for PLL, and distribution divergence for positive pairs)?
- *RQ 3:* Is the proposed model sensitive to hyper-parameters including the threshold percentile for MMD-based positive pair selection and the number of augmented views for each graph?
- *RQ 4:* What is the effect of different GNN encoders on the performance of DEER?
- *RQ 5:* What is the effect of graph augmentation strategies on the performance of DEER?
- *RQ 6:* Are there any visualization of learned graph-level representations, or any case study to demonstrate the effectiveness of DEER for partial label graph learning?

These research questions will be answered in details in Sections IV-B, IV-C, IV-D, IV-E and IV-F, respectively. Our answers will be justified by experimental results.

A. Experimental Setup

1) Datasets: In order to verify the effectiveness of our proposed framework, we evaluate the DEER on five widely used graph benchmark datasets from various domains: a) Synthetic: COLORS-3; b) Bioinformatics: ENZYMES; c) Computer vision: Letter-High, CIFAR10, and COIL-DEL.

- *COLORS-3:* COLORS-3 dataset [64] contains a large number of randomly generated graphs where each node is colored in one of three hues: red, green, or blue. The goal is to capture the number of nodes with a green color in each graph.
- *ENZYMES:* ENZYMES dataset [65] is a bioinformatics dataset comprising 600 protein tertiary structures obtained from the BRENDA enzyme database. Nodes in each graph represent secondary structure elements (SSEs). The dataset includes 6 Enzyme Commission top-level enzyme classes (EC classes).
- *Letter-High:* Letter-High dataset [66] consists of 15 capital letters (A, E, F, H, I, K, L, M, N, T, V, W, X, Y, Z) represented as graphs. Each letter is converted into a prototype graph, where lines are represented as undirected edges, and the endpoints of these lines are represented as nodes.
- *CIFAR10:* CIFAR10 dataset [67] is constructed from a vision dataset with the same name. Super-pixels of images are extracted to construct nodes, and a kNN graph is utilized to characterize the relationships between the super-pixels.
- *COIL-DEL:* COIL-DEL dataset [66] is created from a vision dataset by applying Harris corner detection [68] and Delaunay Triangulation on images. The resulting triangulation is transformed into an undirected graph, where nodes correspond to the endpoints of the lines and edges correspond to the connected lines.

Following the commonly-used partial label setting in computer vision [38], [69], we generate the candidate label set by making $C-1$ independent decisions for negative labels $\overline{y} \neq y$, where each negative label has a flipping probability $q = P(\overline{y} \in Y | \overline{y} \neq y)$ to turn to a false positive, and then enter the candidate set. A higher value of q corresponds to a greater degree of label ambiguity in the partially labeled data. We consider $q \in \{0.1, 0.3, 0.5\}$ for ENZYMES, Letter-High, CIFAR10, and COLORS-3 datasets and adopt $q \in \{0.02, 0.05, 0.1\}$ for COIL-DEL, since COIL-DEL has much more categories and would form larger candidate sets with the same flipping probability.

2) Evaluation Setting: We split the whole dataset into the training, validation, and test sets in the ratio $16:1:3$ for ENZYMES, Letter-High, COIL-DEL, and COLORS-3. While for CIFAR10, we adopt the standard splits with 45,000 training, 5,000 validation, and 10,000 test graphs following [67]. The models are optimized by the training set where each graph is associated with a candidate label set, and then evaluated by the accuracy of predicting the ground-truth labels for graph samples in the test set. Classification accuracy is utilized as our evaluation metric.

3) Baseline Methods: The proposed DEER is evaluated against a variety of competing baselines, which include graph neural network methods, graph pooling methods, graph augmentation methods, unsupervised graph learning methods, weakly-supervised graph learning methods, and partial label learning methods in computer vision.

Graph Neural Networks: We take four different types of graph convolutional layers into comparison:

- *Graph Convolutional Network (GCN)* [10], which aggregates information from neighboring nodes with spectrum-based graph convolutional operations.
- *Graph Attention Network (GAT)* [11], which employs the attention mechanism to adaptively attend to neighboring nodes during information aggregation.
- *Graph Isomorphic Network (GIN)* [13], which leverages multi-layer perceptrons (MLPs) to approximate injective aggregation functions, theoretically showing its strong discriminative power equal to the WL test.
- *GraphSAGE* [12], which samples and aggregates information from the local neighborhood of each node, enabling it to scale to large graphs.

Graph Pooling Methods: We adopt four graph pooling methods as baselines. Specifically, we use:

- *TopK Pooling (TopKPool)* [27], which assesses the significance of nodes by learning a projection vector, subsequently applying the TopK operation to select the most crucial subset of nodes for constructing the coarsened graph.
- *Self-Attention Graph Pooling (SAGPool)* [17], which utilizes the self-attention scores generated by graph convolution to capture both node features and graph topology.
- *Edge Pooling (EdgePool)* [28], which leverages the concept of edge contraction to obtain a localized and sparse hard pooling transform.
- *Adaptive Structure Aware Pooling (ASAP)* [18], which also employs the self-attention mechanism but learns soft cluster assignments for each node to perform pooling.

Graph Augmentation Methods: We compare our proposed framework with a novel mixup-like graph augmentation method called *Graph Transplant* [70], which operates by identifying the sub-structure as a mix unit and exploits node saliency to select meaningful subgraphs and adaptively determine the labels.

Unsupervised Graph Learning Methods: We compare our DEER with three representative graph contrastive learning methods:

- *GraphCL* [52], which proposes several graph augmentation strategies with different underlying priors, and leverages contrastive learning with these augmentations to learn robust representations of graphs without supervise.
- *SimGRACE* [71], a simple framework for graph contrastive learning that utilizes perturbations in GNN encoders, rather than manual and complex augmentations, to generate semantically consistent views for graphs.
- *GraphACL* [72], a novel unsupervised graph classification framework, which improves graph contrastive learning by learning hard negative samples adaptively in an adversarial manner.

Weakly-supervised Graph Learning Methods: We adopt **TGNN** [73], a recent semi-supervised graph-level classification framework for comparison. TGNN jointly leverages a message passing module and a graph kernel module to fully explore graph structural characteristics in both implicit and explicit manners.

The above baseline methods are trained by a commonly-used cross-entropy loss with the prior label distribution $\boldsymbol{Y}^{pri}$ (11) by default. Besides, we optimize GraphACL with this weakly-supervised loss and its proposed contrastive loss jointly. TGNN is trained with both this weakly-supervised loss and its consistency loss to make full use of the partially labeled data, in both weakly-supervised and unsupervised ways. There is a lack of partial label learning methods on graphs, so we make great efforts to find some comparable ones.

Partial label learning methods in computer vision: We also compare our DEER with **PiCO** [38], a state-of-the-art PLL method in computer vision. PiCO guides label disambiguation by contrastive learning for distinguishable representations and prototype learning in the hidden space. We equip PiCO with a GNN-based encoder and adapt it to the setting of PLL on graphs.

4) Implementation Details: Our proposed DEER employs GraphSAGE as the GNN-based encoder by default. The number of GraphSAGE layers is 2, and the hidden dimension is set to 512 for all the datasets. The batch size is set to 128. Besides, the number of augmented views is set to 6, and the percentile α for MMD-based positive selection is 0.7 as the default. For optimization, we adopt Adam [74] as the optimizer with the initial learning rate of 0.001. To get relatively stable predicted distributions for EMA-based label correction and initial features for MMD-based positive pair selection, we warm up the model for 20 epochs, where it only uses prior label distributions as the target and graph augmentations as the positive views. We implement the proposed DEER with PyTorch and conduct experiments with an NVIDIA GeForce GTX 1080 Ti.

B. Performance Comparison (RQ 1)

We record the classification accuracy of our DEER and the aforementioned baseline methods on four graph classification benchmarks with respect to the degree of label ambiguity q. Observing the results reported in Table II, we can make the following conclusions:

- Overall, compared with other baseline methods, the proposed DEER achieves the state-of-the-art performance on all datasets with different settings of partial labels. Particularly, DEER averagely outperforms the best baseline method with more than 6.7%, 6%, 6.5% and 5% performance gain on ENZYME, Letter-High, CIFAR10 and COIL-DEL datasets respectively. The superiority of DEER demonstrates the effectiveness of the proposed framework on graph classification when there are only partial labels available.
- Among all categories of baseline methods, Graph Transplant generally outperforms others on Letter-High and CIFAR10, mainly due to its ability to learn superior representations through the use of mixup augmentation; PiCO performs best on COIL-DEL, which may benefit from better distinguishability and disambiguation provided by contrastive learning and prototype learning in the scenario of many classes; TGNN shows superior performance on COLORS-3, which may owe to the complementary graph semantic patterns captured by its two modules. Among different types of graph neural network models, GraphSAGE performs better than others on Letter-High and COIL-DEL datasets, while GAT shows superior performance on the CIFAR10 dataset. Besides, among the four types of pooling methods, EdgePool exhibits relatively stable performance across all datasets.
- When the flipping ratio q of the negative labels increases, all methods suffer from information loss and the ambiguity brought by larger candidate sets. In general, when testing on datasets with more classes (e.g. Letter-High and COIL-DEL), the performance of models will drop dramatically with the increase of the flipping probability q, which is a reasonable result since the models are confronted with more confusing labels under this situation. Nevertheless, our DEER is less affected by this issue, manifesting its impressive robustness to ambiguous labels, which may owe to the label correction mechanism and the rich semantics from contrasting the positive pairs.

Accuracy curves: We record the accuracy of the DEER's prediction on the training set and the test set as well as the accuracy of the label matrix of the training set during the training process. As shown in Fig. 3, after the first 20 epochs' warm-up with the prior label distributions, the accuracy of the label matrix increases steadily, owing to the correction from the posterior distributions $\boldsymbol{Y}^{pos}$. Note that when the model is not fully trained yet, the accuracy of the posterior distributions $\boldsymbol{Y}^{pos}$ is much higher than that of the raw prediction of the model due to the re-normalization with the candidate prior, which helps exclude non-candidate classes. Thus the label accuracy improves rapidly and exceeds the training accuracy near the 20^{th} epoch. With the

TABLE II
THE CLASSIFICATION ACCURACY (MEAN%±STD% OVER 5 RUNS) ON FIVE GRAPH BENCHMARK DATASETS (PART 1/2)

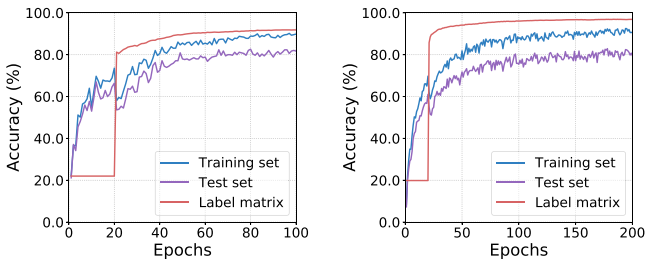
Dataset	ENZYMES			Letter-High			CIFAR10		
Method	$q=0.1$	$q=0.3$	$q=0.5$	$q=0.1$	$q=0.3$	$q=0.5$	$q=0.1$	$q=0.3$	$q=0.5$
GCN	61.33±2.85	48.44±2.06	40.22±2.93	50.09±0.70	44.00±1.08	35.94±1.82	47.18±1.09	43.68±0.68	41.35±0.65
GAT	58.22±3.03	49.11±2.93	34.67±3.87	73.39±1.41	61.33±3.48	53.04±3.06	57.56±0.65	52.93±1.22	48.54±0.46
GIN	59.78±4.58	47.11±4.59	34.22±1.78	55.83±4.28	50.43±1.92	35.59±3.75	47.29±0.61	43.91±0.45	41.24±0.52
GraphSAGE	60.89±1.09	47.33±3.03	39.33±3.11	78.20±1.17	70.96±1.48	60.35±1.83	57.22±0.67	51.92±0.26	47.44±0.83
TopKPool	53.11±4.12	44.22±2.76	36.00±4.80	67.07±1.60	55.25±2.74	43.83±5.21	55.26±0.85	48.97±1.24	42.87±1.31
SAGPool	56.89±5.37	46.67±2.53	37.11±5.00	67.42±1.91	55.71±4.71	39.30±5.49	54.23±0.53	50.01±0.68	45.16±0.36
EdgePool	58.67±2.67	51.11±3.06	33.33±1.99	70.49±3.29	64.17±2.44	55.36±2.16	55.09±0.61	50.17±0.64	45.90±0.44
ASAP	60.89±2.67	44.44±3.06	31.56±3.34	71.25±1.44	65.04±1.22	52.75±4.41	54.56±0.66	50.10±0.63	44.81±1.57
Graph Transplant	61.56±2.86	51.78±2.39	43.78±3.41	80.75±0.60	74.84±1.44	66.78±1.86	56.87±1.28	53.79±1.11	48.95±1.47
PiCO	61.08±6.67	46.88±2.76	35.78±3.02	81.27±1.60	73.56±1.71	64.63±4.35	57.70±0.82	53.47±1.14	46.04±1.20
TGNN	62.44±3.01	53.33±3.51	42.22±4.39	78.55±0.78	70.43±0.97	59.83±1.32	OOM	OOM	OOM
GraphCL	61.78±1.51	54.22±5.14	39.78±5.09	78.43±0.85	72.00±2.01	62.49±2.00	57.62±0.56	53.57±0.87	48.10±0.61
SimGRACE	62.22±4.56	53.11±2.67	38.67±3.01	76.64±1.35	70.14±1.91	59.01±1.61	57.45±0.22	51.07±0.39	46.59±1.22
GraphACL	58.22±1.51	54.44±2.33	44.89±4.75	81.04±1.01	69.80±1.01	57.68±2.85	57.65±0.21	53.25±0.62	47.86±0.65
DEER (Ours)	**67.11±1.66**	**58.22±2.18**	**47.56±2.57**	**83.48±0.92**	**80.12±1.26**	**72.29±1.54**	**61.45±0.40**	**57.08±0.57**	**52.48±0.72**

The best results are shown in boldface and the second best results are underlined. $q = P(\overline{y} \in Y | \overline{y} \neq y)$ reflects the degree of label ambiguity. OOM represents out-of-memory. It shows that our DEER outperforms the rest.

TABLE III
THE CLASSIFICATION ACCURACY (MEAN%±STD% OVER 5 RUNS) ON FIVE GRAPH BENCHMARK DATASETS (PART 2/2)

Dataset	COIL-DEL			COLORS-3		
Method	$q=0.02$	$q=0.05$	$q=0.1$	$q=0.1$	$q=0.3$	$q=0.5$
GCN	60.77±1.71	50.43±1.07	41.63±1.74	90.34±0.06	74.87±0.25	60.67±1.64
GAT	69.11±2.86	59.77±1.97	46.63±1.54	89.33±1.42	71.83±0.22	62.56±3.54
GIN	55.94±1.69	46.23±0.88	37.29±1.04	63.01±1.45	48.17±0.44	41.00±2.75
GraphSAGE	71.40±2.15	58.91±1.92	49.23±1.90	91.70±2.18	71.24±3.00	56.63±5.81
TopKPool	55.80±4.86	44.83±2.19	34.63±2.08	82.35±1.36	56.69±3.58	33.49±1.74
SAGPool	52.94±2.59	41.89±4.28	30.17±1.85	76.99±4.39	59.91±4.14	24.62±0.32
EdgePool	68.74±1.85	56.74±3.98	45.89±1.30	87.47±0.41	76.96±0.13	62.31±1.23
ASAP	59.03±3.09	46.20±4.08	34.94±3.02	77.84±1.26	70.11±0.54	62.47±0.98
Graph Transplant	80.09±0.75	66.57±1.60	57.11±1.03	85.48±0.89	74.66±1.30	62.72±2.37
PiCO	84.88±1.09	76.25±1.66	63.69±1.42	65.68±1.07	53.99±0.92	34.74±1.77
TGNN	70.49±0.87	62.28±1.05	50.20±1.17	93.16±1.55	75.84±1.81	63.95±2.18
GraphCL	78.83±1.06	69.94±2.31	60.17±2.96	92.71±1.61	72.89±1.24	61.55±1.01
SimGRACE	72.66±1.52	60.51±2.31	50.97±3.95	88.51±0.44	72.10±3.16	55.02±1.10
GraphACL	80.66±0.41	71.40±0.95	60.29±3.04	92.05±0.50	73.84±2.12	63.57±1.39
DEER (Ours)	**87.86±1.41**	**79.94±1.20**	**68.03±1.11**	**96.23±2.94**	**88.13±2.08**	**66.81±2.82**

The best results are shown in boldface and the second best results are underlined.



(a) Letter-High (q=0.3) (b) COIL-DEL (q=0.05)

Fig. 3. Accuracy curves of our proposed DEER on Letter-High and COIL-DEL. The first 20 epochs are under warm-up with the prior label distributions.

accurate guide of the continuously disambiguated label matrix, the proposed approach achieves superior performance gradually, justifying the benefits of EMA-based label correction.

C. Ablation Studies (RQ 2)

We investigate the effectiveness of key components in our DEER from three aspects: EMA-based label correction, graph contrastive learning, and MMD-based distribution divergence for positive pairs. The ablation study results are summarized in Table IV.

- *Effect of EMA-based label correction:* The baseline method in Table IV is denoted as M_1, whereas M_2 and M_3 improve upon it by incorporating straightforward and EMA-based label correction (LC) to identify the true labels from candidate sets, respectively. The straightforward LC strategy disambiguates the noisy candidate labels by the one-hot pseudo labels derived from the model predictions. It can be observed that both LC strategies improve the performance, and the EMA-based LC leads to a more notable increase than the straightforward LC in accuracy across all datasets with various settings of q. These findings

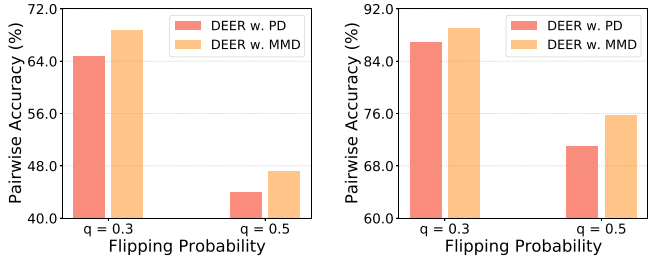
TABLE IV
ABLATION STUDY ON ENZYMES AND LETTER-HIGH

	Correlations			ENZYMES			Letter-High		
	EMALC	Contr	MMD	$q = 0.1$	$q = 0.3$	$q = 0.5$	$q = 0.1$	$q = 0.3$	$q = 0.5$
M_1				60.89	47.33	39.33	78.20	70.96	60.35
M_2	SFLC			61.33	52.44	40.22	78.72	71.71	61.57
M_3	√			62.22	54.67	44.22	80.58	73.04	66.67
M_4	√	√		65.56	56.00	45.78	82.03	78.84	69.74
M_5	√	√	PD	66.22	56.89	46.44	82.61	79.13	70.43
M_6 (full model)	√	√	√	**67.11**	**58.22**	**47.56**	**83.48**	**80.12**	**72.29**

EMALC refers to EMA-based label correction, SFLC refers to straightforward label correction, contr refers to graph contrastive learning, MMD corresponds to MMD-based distribution divergence for positive pairs, and PD corresponds to positive pair selection based on the pairwise distance of deep features.

suggest that excluding noisy candidate labels is vital for partial label learning, and our EMA-based label correction is effective in mitigating the influence of noisy labels.

- *Effect of graph contrastive learning:* In Table IV, M_4 is an improvement over M_3 by integrating graph contrastive learning, which encourages positive pairs to be closer to each other than negative pairs. It is worth noting that in M_4, positive pairs are constructed using only pseudo-labels. Comparing the results of M_3 and M_4, it is evident that graph contrastive learning plays a vital role in enhancing the representations learning on graphs and achieving better accuracy by a significant margin on both datasets.
- *Effect of distribution divergence for positive pairs:* Compared to M_4, M_5 further utilizes the pairwise distance (PD) of deep features to determine the positive pairs for graph contrastive learning, while M_6 employs the MMD-based distribution divergence for positive pair identification. As shown in Table IV, the inclusion of both metrics (PD or MMD) for positive pair selection improves the performance, and M_6 further outperforms M_5 on both datasets. This is due to the fact that weak supervised signals and noisy label candidates make it insufficient to identify effective positive pairs solely using the pseudo-labels. And our proposed MMD-based distribution divergence module effectively overcomes this limitation by characterizing the underlying semantic distributions in the hidden space, enabling it to identify accurate positive graph pairs. Moreover, it benefits the graph contrastive learning framework, allowing for more discriminative graph representations. We further compare the accuracy of positive pairs selected by these two strategies in Fig. 4. As we can see, the positive pairs identified by the MMD-based distribution divergence have higher accuracy, demonstrating that our proposed metric is more unbiased and accurate under label ambiguity.



(a) ENZYMES (b) Letter-High

Fig. 4. Accuracy of positive pairs w.r.t. different selection strategies. MMD corresponds to MMD-based distribution divergence for positive pair selection, while PD corresponds to positive pair selection based on the pairwise distance of deep features. (a) ENZYMES. (b) Letter-High.

D. Sensitivity Analysis (RQ 3)

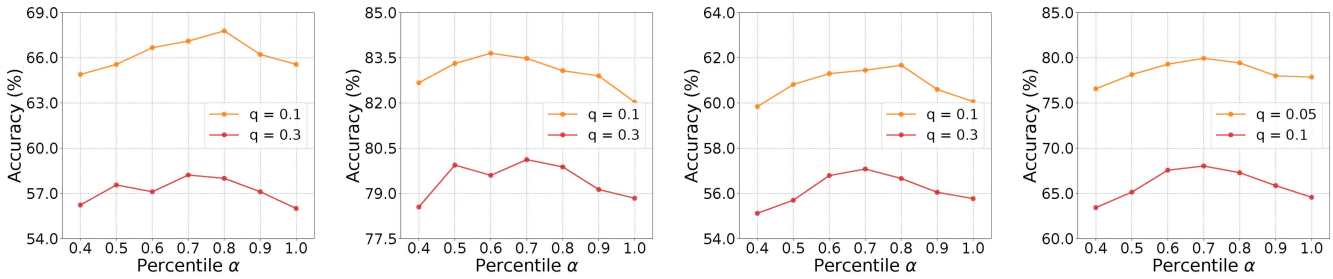
We also investigate the sensitivity of our proposed DEER to the hyper-parameters. Specifically, we look into the effect of varying threshold percentiles for MMD-based positive pair selection and numbers of augmented views for each graph.

1) Analysis of Percentile for MMD-Based Selection: We first vary the threshold percentile α for MMD-based positive pair selection. Note that positive pairs are determined by both distribution divergence and pseudo labels, thus the MMD metric can be regarded as a filter of the target-oriented positive pairs generated by pseudo labels, where percentile α is the filtering ratio. As shown in Fig. 5, the performance tends to improve gradually when the percentile α decreases from 1.0 to 0.7. It suggests that filtering out false positive pairs (i.e. whose semantic distributions are far away in the hidden space) is beneficial to graph contrastive learning. This finding justifies the effectiveness of positive pair selection based on semantic distribution divergence. Nevertheless, the performance could be harmed by the overly-small percentile α, due to the potential loss of true positive pairs.

2) Analysis of the Number of Augmented Views: Next, we investigate the influence of the number of augmented views M for each graph. Fig. 6 shows the results on four graph datasets. We can see that the performance of our DEER improves gradually with the increase of the augmented view number until saturation. It illustrates that sampling more semantic-preserved augmentations for each graph is advantageous to capture the semantic distribution behind the graph, and alleviate the biases of pairwise distances caused by the manifold boundary in the feature space. Moreover, we find that six augmented views are sufficient for measuring underlying semantic distribution divergence among graphs generally.

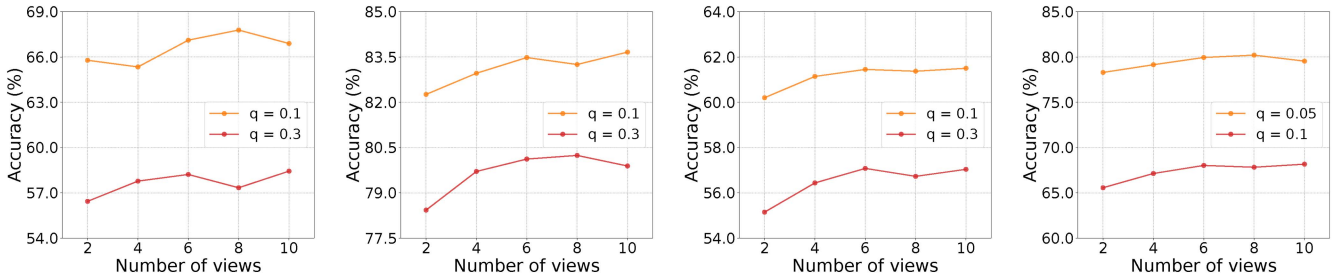
E. Effect of Different GNN Encoders (RQ 4)

We further examine the effect of different GNN encoders. Specifically, we equip our DEER with three widely used GNN



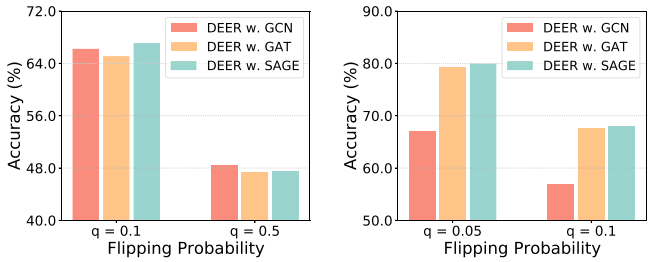
(a) ENZYMES (b) Letter-High (c) CIFAR10 (d) COIL-DEL

Fig. 5. Analysis of the threshold percentile α for MMD-based positive pair selection.



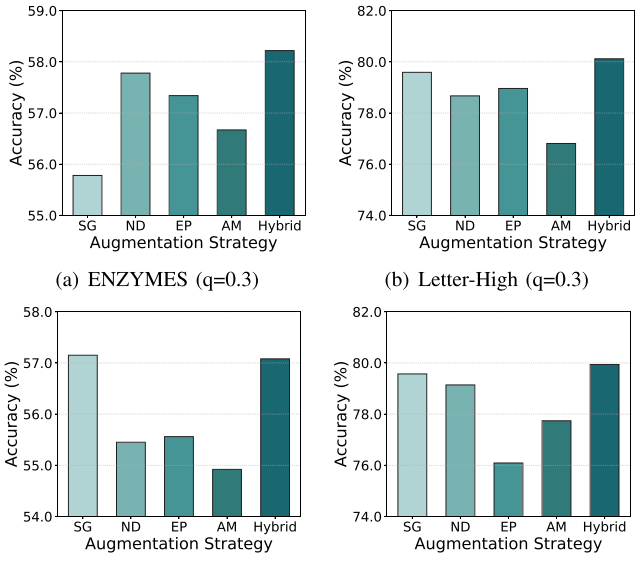
(a) ENZYMES (b) Letter-High (c) CIFAR10 (d) COIL-DEL

Fig. 6. Analysis of the number of augmented views M for each graph.



(a) ENZYMES (b) COIL-DEL

Fig. 7. Performance comparison w.r.t. different GNN encoders. SAGE corresponds to GraphSAGE.



(a) ENZYMES (q=0.3) (b) Letter-High (q=0.3)

(c) CIFAR10 (q=0.3) (d) COIL-DEL (q=0.05)

Fig. 8. Effect of augmentation strategies. SG denotes subgraph, ND denotes node dropping, EP denotes edge perturbation, AM denotes attribute masking, and Hybrid means random selection of these four strategies.

architectures, i.e., GCN [10], GAT [11], GraphSAGE [12], and compare their performance on ENZYMES and COIL-DEL. The results are depicted in Fig. 7. As we can see, GraphSAGE performs well in most cases, illustrating its strong robustness under label ambiguity. Therefore, we employ GraphSAGE as the GNN-based encoder of our DEER for partial label learning on graphs. Besides, it can be observed that GCN performs much worse than the other two encoders on COIL-DEL, which suggests the significance of adopting an appropriate GNN encoder against label noise.

F. Effect of Graph Augmentation (RQ 5)

As suggested by previous literatures [45], [52], [75], augmentation strategies are important to graph contrastive learning. Thus, we examine the influence of various graph augmentation strategies on our proposed DEER in this section. Fig. 8 shows the results on ENZYMES and Letter-High when $q = 0.3$. According to the plots, we can draw the following observations: (i) Various augmentation strategies have different effects on different datasets. The potential reason is that the degree of semantic preservation through various augmentation strategies is influenced by the intrinsic characteristics of graph topologies and node features in different datasets. (ii) The hybrid application of

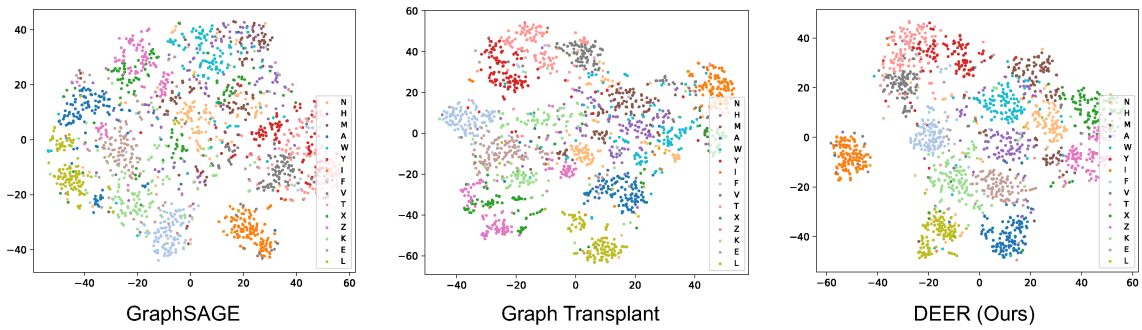


Fig. 9. Visualization of learned features on Letter-High (q=0.3) using t-SNE.

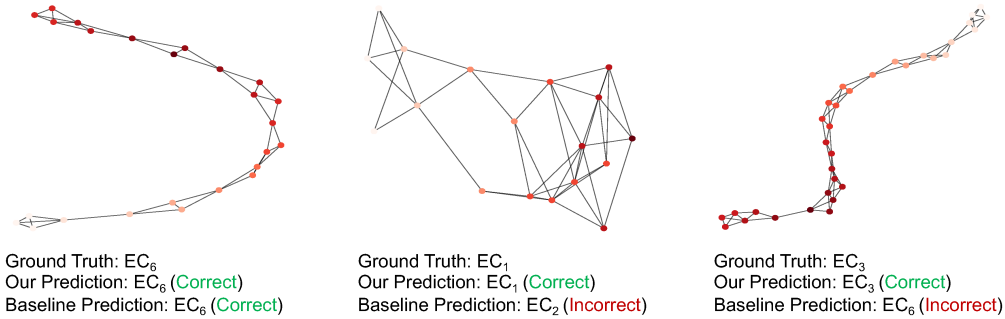


Fig. 10. Visualization of three cases on ENZYMES. Our proposed DEER successfully outputs correct prediction in all three cases while the baseline TGNN fails two of them.

the four augmentation strategies by random selection performs well on both datasets, demonstrating its effectiveness and robustness in capturing underlying semantic distributions behind graphs. This phenomenon may owe to the comprehensive exploration of diverse semantic aspects of graphs.

G. Visualization (RQ 6)

To intuitively illustrate the effectiveness of the proposed DEER in the feature space under the partial label setting, we visualize the learned graph-level features of GraphSAGE, the best baseline Graph Transplant and our DEER with t-SNE algorithm [76] in Fig. 9. The learned features of different classes are marked with corresponding colors. From the figure we can observe that: (i) Generally, the cluster structures of features are highly related to the model performance. Specifically, the clearer the cluster structures are, the better performance the method can achieve. (ii) Our proposed DEER can clarify different graph classes even though the full ground truth is not given. In comparison, the other two baseline methods fail to distinguish some similar classes (e.g. Z and X) due to the lack of some clearer supervised signals. It demonstrates the effectiveness of the idea of supervising the model with the accurate positive pairs identified by the distribution divergence measurement, and the clear target labels disambiguated by the posterior label distributions.

In addition, we conduct case studies to analyze the benefits of our proposed DEER. In particular, three examples from ENZYMES and their predictions from our DEER and TGNN are illustrated in Fig. 10. From the results, we can observe that TGNN fails two cases while our proposed DEER generates correct predictions in all three cases. The latter two cases are more complicated than the first one. Particularly, the third case is similar to the first one in the structure, which may be the potential reason for TGNN classifying them into the same category wrongly. However, our DEER still captures the differences between these two cases and classifies them correctly, demonstrating that our DEER can handle graph-structured hard examples with partial label learning.

V. Conclusion

This paper studies an underexplored but practical problem of partial label learning on graphs, where each graph is assigned a collection of candidate labels. Here we propose a novel approach named Distribution Divergence-based Graph Contrast (DEER) to solve this challenging problem. Our DEER leverages graph augmentation to infer underlying semantic distributions in the hidden space, and measure their divergence to identify positive pairs for effective graph contrastive learning. In order to remove noisy candidate labels, we adopt a label correction strategy, which starts from prior distributions and then updates target labels using the exponential moving average method.

Extensive experiments on various benchmark datasets show the effectiveness of our proposed DEER in comparison to various competing approaches. In the future work, we will explore the following topics: (1) extending our proposed framework DEER to more weakly supervised scenarios such as semi-supervised learning, learning with label noise and few-shot learning; (2) applying our proposed DEER to various crucial

scientific applications such as drug property predictions and protein function analysis; (3) trying more advanced graph machine techniques such as graph kernels to make our approach more effective.

ACKNOWLEDGMENT

The authors are grateful to the anonymous reviewers for critically reading the manuscript and for giving important suggestions to improve their paper.

REFERENCES

[1] Q. Wang et al., "DualGNN: Dual graph neural network for multimedia recommendation," *IEEE Trans. Multimedia*, vol. 25, pp. 1074–1084, 2023.
[2] Z. Li et al., "Consensus graph learning for multi-view clustering," *IEEE Trans. Multimedia*, vol. 24, pp. 2461–2472, 2022.
[3] J. Yuan et al., "Self-supervised graph neural network for multi-source domain adaptation," in *Proc. 30th ACM Int. Conf. Multimedia*, 2022, pp. 3907–3916.
[4] K. Liu et al., "Multimodal graph contrastive learning for multimedia-based recommendation," *IEEE Trans. Multimedia*, vol. 25, pp. 9343–9355, 2023.
[5] X. Zhou et al., "Inductive graph transformer for delivery time estimation," in *Proc. 16th ACM Int. Conf. Web Search Data Mining*, 2023, pp. 679–687.
[6] F. Rousseau, E. Kiagias, and M. Vazirgiannis, "Text categorization as a graph classification problem," in *Proc. 53rd Annu. Meeting Assoc. Comput. Linguistics 7th Int. Joint Conf. Natural Lang. Process.* 2015, pp. 1702–1712.
[7] F. Zhao, C. Li, Z. Wu, S. Xing, and X. Dai, "Learning from different text-image pairs: A relation-enhanced graph convolutional network for multimodal NER," in *Proc. 30th ACM Int. Conf. Multimedia*, 2022, pp. 3983–3992.
[8] J. Yoo, S. Shim, and U. Kang, "Model-agnostic augmentation for accurate graph classification," in *Proc. ACM Web Conf.* 2022, pp. 1281–1291.
[9] W. Ju et al., "KGNN: Harnessing kernel-based networks for semi-supervised graph classification," in *Proc. 15th ACM Int. Conf. Web Search Data Mining*, 2022, pp. 421–429.
[10] M. Welling and T. N. Kipf, "Semi-supervised classification with graph convolutional networks," in *Proc. Int. Conf. Learn. Representations*, 2017.
[11] P. Veličković et al., "Graph attention networks," in *Proc. Int. Conf. Learn. Representations*, 2018.
[12] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 1025–1035.
[13] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How powerful are graph neural networks ?," in *Proc. Int. Conf. Learn. Representations*, 2018.
[14] W. Ju et al., "A comprehensive survey on deep graph representation learning," *Neural Netw.*, vol. 173, 2024, Art. no. 106207.
[15] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, "Neural message passing for quantum chemistry," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 1263–1272.
[16] Z. Ying et al., "Hierarchical graph representation learning with differentiable pooling," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 31, 2018, pp. 4805–4815.
[17] J. Lee, I. Lee, and J. Kang, "Self-attention graph pooling," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 3734–3743.
[18] E. Ranjan, S. Sanyal, and P. Talukdar, "ASAP: Adaptive structure aware pooling for learning hierarchical graph representations," in *Proc. AAAI Conf. Artif. Intell.*, 2020, vol. 34, no. 4, pp. 5470–5477.
[19] X. Gao, W. Dai, C. Li, H. Xiong, and P. Frossard, "iPool–information-based pooling in hierarchical graph neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 9, pp. 5032–5044, Sep. 2022.
[20] H. Chermette, "Chemical reactivity indexes in density functional theory," *J. Comput. Chem.*, vol. 20, no. 1, pp. 129–154, 1999.
[21] A. D. Becke, "Perspective: Fifty years of density-functional theory in chemical physics," *J. Chem. Phys.*, vol. 140, no. 18, 2014, Art. no. 18A301.
[22] X. Fang et al., "Geometry-enhanced molecular representation learning for property prediction," *Nature Mach. Intell.*, vol. 4, no. 2, pp. 127–134, 2022.
[23] Z. Zhang, Q. Liu, H. Wang, C. Lu, and C.-K. Lee, "Motif-based graph self-supervised learning for molecular property prediction," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 34, pp. 15870–15882, 2021.
[24] K. M. Borgwardt and H.-P. Kriegel, "Shortest-path kernels on graphs," in *Proc. IEEE 15th Int. Conf. Data Mining*, 2005, pp. 1–8.
[25] N. Shervashidze, S. Vishwanathan, T. Petri, K. Mehlhorn, and K. Borgwardt, "Efficient graphlet kernels for large graph comparison," in *Proc. Artif. Intell. Statist.*, 2009, pp. 488–495.
[26] N. Shervashidze, P. Schweitzer, E. J. Van Leeuwen, K. Mehlhorn, and K. M. Borgwardt, "Weisfeiler-lehman graph kernels," *J. Mach. Learn. Res.*, vol. 12, no. 9, pp. 2539–2561, 2011.
[27] H. Gao and S. Ji, "Graph U-nets," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 2083–2092.
[28] F. Diehl, "Edge contraction pooling for graph neural networks," 2019, *arXiv:1905.10990*.
[29] E. Hüllermeier and J. Beringer, "Learning from ambiguously labeled examples," in *Proc. Adv. Intell. Data Anal. VI: 6th Int. Symp. Intell. Data Anal.*, 2005, pp. 168–179.
[30] T. Cour, B. Sapp, and B. Taskar, "Learning from partial labels," *J. Mach. Learn. Res.*, vol. 12, pp. 1501–1536, 2011.
[31] M.-L. Zhang and F. Yu, "Solving the partial label learning problem: An instance-based approach," in *Proc. 24th Int. Conf. Artif. Intell.*, 2015, pp. 4048–4054.
[32] C. Gong et al., "A regularization approach for instance-based superset label learning," *IEEE Trans. Cybern.*, vol. 48, no. 3, pp. 967–978, Mar. 2018.
[33] Y. Zhou, J. He, and H. Gu, "Partial label learning via Gaussian processes," *IEEE Trans. Cybern.*, vol. 47, no. 12, pp. 4443–4450, Dec. 2017.
[34] L. Feng and B. An, "Partial label learning by semantic difference maximization," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, 2019, pp. 2294–2300.
[35] Y. Yan and Y. Guo, "Partial label learning with batch label correction," in *Proc. AAAI Conf. Artif. Intell.*, 2020, vol. 34, no. 4, pp. 6575–6582.
[36] G. Lyu, S. Feng, T. Wang, and C. Lang, "A self-paced regularization framework for partial-label learning," *IEEE Trans. Cybern.*, vol. 52, no. 2, pp. 899–911, Feb. 2022.
[37] W. Wang and M.-L. Zhang, "Partial label learning with discrimination augmentation," in *Proc. Int. ACM SIGKDD Conf. Knowl. Discov. Data Mining*, 2022, pp. 1920–1928.
[38] H. Wang et al., "PICO: Contrastive label disambiguation for partial label learning," in *Proc. Int. Conf. Learn. Representations*, 2022.
[39] J. Shen, M. Wang, S. Yan, and X.-S. Hua, "Multimedia tagging: Past, present and future," in *Proc. 19th ACM Int. Conf. Multimedia*, 2011, pp. 639–640.
[40] C. Cui, J. Shen, J. Ma, and T. Lian, "Social tag relevance learning via ranking-oriented neighbor voting," *Multimedia Tools Appl.*, vol. 76, pp. 8831–8857, 2017.
[41] J. Shen, M. Wang, and T.-S. Chua, "Accurate online video tagging via probabilistic hybrid modeling," *Multimedia Syst.*, vol. 22, pp. 99–113, 2016.
[42] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 1597–1607.
[43] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 9729–9738.
[44] X. Luo et al., "A statistical approach to mining semantic similarity for deep unsupervised hashing," in *Proc. 29th ACM Int. Conf. Multimedia*, 2021, pp. 4306–4314.
[45] Y. You, T. Chen, Y. Shen, and Z. Wang, "Graph contrastive learning automated," in *Proc. Int. Conf. Mach. Learn.*, 2021, pp. 12121–12132.
[46] S. Suresh, P. Li, C. Hao, and J. Neville, "Adversarial graph augmentation to improve graph contrastive learning," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 34, 2021, pp. 15920–15933.
[47] H. Yang et al., "Generating counterfactual hard negative samples for graph contrastive learning," in *Proc. ACM Web Conf. 2023. ACM*, 2023, pp. 621–629.
[48] W. Ju et al., "Unsupervised graph-level representation learning with hierarchical contrasts," *Neural Netw.*, vol. 158, pp. 359–368, 2023.
[49] L. Yang, L. Zhang, and W. Yang, "Graph adversarial self-supervised learning," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 34, pp. 14887–14899, 2021.
[50] L. Peng et al., "GRLC: Graph representation learning with constraints," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 35, no. 6, pp. 8609–8622, 2024, doi: 10.1109/TNNLS.2022.3230979.
[51] Y. Mo et al., "Disentangled multiplex graph representation learning," in *Proc. Int. Conf. Mach. Learn.*, 2023, pp. 24983–25005.
[52] Y. You et al., "Graph contrastive learning with augmentations," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, pp. 5812–5823, 2020.
[53] A. Gretton, K. Borgwardt, M. Rasch, B. Schölkopf, and A. Smola, "A kernel method for the two-sample-problem," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 19, 2006, pp. 513–520.

[54] D. Sejdinovic, B. Sriperumbudur, A. Gretton, and K. Fukumizu, "Equivalence of distance-based and RKHS-based statistics in hypothesis testing," *Ann. Statist.*, vol. 41, pp. 2263–2291, 2013.
[55] H. Yan et al., "Mind the class weight bias: Weighted maximum mean discrepancy for unsupervised domain adaptation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 2272–2281.
[56] A. Berlinet and C. Thomas-Agnan, *Reproducing Kernel Hilbert Spaces in Probability and Statistics*. Berlin, Germany: Springer, 2011.
[57] B.K. Sriperumbudur, K. Fukumizu, and G. R. Lanckriet, "Universality, characteristic kernels and RKHS embedding of measures,," *J. Mach. Learn. Res.*, vol. 12, no. 7, 2011, pp. 2389–2410.
[58] M. Long, Y. Cao, J. Wang, and M. Jordan, "Learning transferable features with deep adaptation networks," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 97–105.
[59] I. O. Tolstikhin, B.K. Sriperumbudur, and B. Schölkopf, "Minimax estimation of maximum mean discrepancy with radial kernels," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 29, 2016, pp. 1938–1946.
[60] V. Solanki, P. Rubin-Delanchy, and I. Gallagher, "Persistent homology of graph embeddings," 2019, *arXiv:1912.10238*.
[61] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. Smola, "A kernel two-sample test," *J. Mach. Learn. Res.*, vol. 13, no. 1, pp. 723–773, 2012.
[62] F. Han, "An exponential inequality for u-statistics under mixing conditions," *J. Theor. Probability*, vol. 31, no. 1, pp. 556–578, 2018.
[63] T. Wang and P. Isola, "Understanding contrastive representation learning through alignment and uniformity on the hypersphere," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 9929–9939.
[64] B. Knyazev, G. W. Taylor, and M. Amer, "Understanding attention and generalization in graph neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, 2019, pp. 4202–4212.
[65] I. Schomburg et al., "BRENDA, the enzyme database: Updates and major new developments," *Nucleic acids Res.*, vol. 32, no. suppl_1, pp. D431–D433, 2004.
[66] K. Riesen and H. Bunke, "IAM graph database repository for graph based pattern recognition and machine learning," in *Proc. Joint IAPR Int. Workshops Stat. Techn. Pattern Recognit. Struct. Syntactic Pattern Recognit.*, 2008, pp. 287–297.
[67] V. P. Dwivedi, C. K. Joshi, T. Laurent, Y. Bengio, and X. Bresson, "Benchmarking graph neural networks," *J. Mach. Learn. Res.*, vol. 24, no. 43, pp. 1–48, 2023.
[68] C. Harris et al., "A combined corner and edge detector," in *Proc. Alvey Vis. Conf.*, 1988, vol. 15, no. 50, pp. 10–5244.
[69] S. He, L. Feng, F. Lv, W. Li, and G. Yang, "Partial label learning with semantic label representations," in *Proc. 28th ACM SIGKDD Conf. Knowl. Discov. Data Mining*, 2022, pp. 545–553.
[70] J. Park, H. Shim, and E. Yang, "Graph transplant: Node saliency-guided graph mixup with local structure preservation," in *Proc. AAAI Conf. Artif. Intell.*, vol. 36, no. 7, 2022, pp. 7966–7974.
[71] J. Xia, L. Wu, J. Chen, B. Hu, and S. Z. Li, "Simgrace: A simple framework for graph contrastive learning without data augmentation," in *Proc. ACM Web Conf.*, 2022, pp. 1070–1079.
[72] X. Luo et al., "Self-supervised graph-level representation learning with adversarial contrastive learning," *ACM Trans. Knowl. Discov. Data*, vol. 18, no. 2, pp. 1–23, 2023.
[73] W. Ju et al., "TGNN: A joint semi-supervised framework for graph-level classification," in *Proc. 31th Int. Joint Conf. Artif. Intell.*, 2022, pp. 2122–2128.
[74] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. Int. Conf. Learn. Representations*, 2015.
[75] Y. Zhu et al., "Graph contrastive learning with adaptive augmentation," in *Proc. Web Conf.*, 2021, pp. 2069–2080.
[76] L. Van der Maaten and G. Hinton, "Visualizing data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, no. 11, pp. 2579–2605, 2008.

Yiyang Gu received the B.S. degree in computer science from Peking University, Beijing, China, in 2021, where he is currently working toward the Ph.D. degree in computer science. His research interests include data mining, graph machine learning, self-supervised learning, and bioinformatics.

Zihao Chen received the B.S. degree in statistics in 2022 from Peking University, Beijing, China, where he is currently working toward the Ph.D. degree with the School of Mathematical Sciences. His research interests include biostatistics, bioinformatics, and omics data analysis.

Yifang Qin recieved the B.S. degree in computer science from Peking University, Beijing, China, in 2023. He is currently working toward the Ph.D. degree in computer science. His research interests include graph representation learning and recommender systems.

Zhengyang Mao is currently working toward the master's degree with the School of Computer Science, Peking University, Beijing, China. His research interests include graph representation learning and long-tailed learning.

Zhiping Xiao received the B.S. degree from Peking University, Beijing, China, in 2016, the M.S. degree in computer science from the University of California at Berkely, Berkeley, CA, USA, in 2018, and the Ph.D. degree in computer science from the University of California at Los Angeles, Los Angeles, CA, USA, in 2024. She is currently a Research Scientist with Computer Science Department, the University of California at Los Angeles. Her research interests include graph representation learning, graph neural networks, and applications such as social network analysis and computational biomedicine.

Wei Ju (Member, IEEE) received the B.S. degree in mathematics from Sichuan University, Chengdu, China, in 2017, and the Ph.D. degree in computer science from Peking University, Beijing, China, in 2022. He is currently a Postdoc Research Fellow in computer science with Peking University. He has authored or coauthored more than 40 papers in top-tier venues. His research interests include the area of machine learning on graphs including graph representation learning and graph neural networks, and interdisciplinary applications such as knowledge graphs, bioinformatics, drug discovery and recommender systems. He was the recipient of the best paper finalist in IEEE ICDM 2022.

Chong Chen received B.S. degree in mathematics and applied mathematics and the Ph.D. degree in statistics from Peking University, Beijing, China, in 2013 and 2019, respectively. He is currently a Research Scientist with Terminus Group, Beijing. His research interests include statistics, machine learning, and computer vision.

Xian-Sheng Hua (Fellow, IEEE) received the B.S. and Ph.D. degrees in applied mathematics from Peking University, Beijing, China, in 1996 and 2001, respectively. In 2001, he joined Microsoft Research Asia, Beijing, as a Researcher. Since 2013. he has been a Senior Researcher with Microsoft Research Redmond, Redmond, WA, USA. In 2015, he became a Researcher and the Senior Director with the Alibaba Group, Hangzhou, China. He has authored or coauthored more than 250 research papers and has filed over 90 patents. His research interests include the areas of multimedia search, advertising, understanding, and mining, pattern recognition, and machine learning. Dr. Hua is an ACM Distinguished Scientist. He was on the Technical Directions Board of the IEEE Signal Processing Society. He was the recipient of the MIT Technology Review Innovators Under 35 Asia Pacific (MIT35). He was the Program Co-Chair for the ACM Multimedia 2012, IEEE International Conference on Multimedia and Expo (ICME) 2012, and IEEE ICME 2013.

Yifan Wang received the B.S. degree in software engineering and the M.S. degree in software engineering from Northeastern University, Shenyang, China, in 2014 and 2017 respectively, and the Ph.D. degree in computer science from Peking University, Beijing, China, in 2023. He is currently an Assistant Professor with the School of Information Technology and Management, University of International Business and Economics, Beijing. His research interests include graph representation learning, graph neural networks, disentangled representation learning, and corresponding applications, such as drug discovery and recommender systems.

Xiao Luo received the B.S. degree in mathematics from Nanjing University, Nanjing, China, in 2017, and the Ph.D. degree from the School of Mathematical Sciences, Peking University, Beijing, China. He is currently a Postdoctoral Researcher with the Department of Computer Science, University of California, Los Angeles, CA, USA. His research interests includes machine learning on graphs, image retrieval, statistical models, and bioinformatics.

Ming Zhang (Member, IEEE) received the B.S., M.S., and Ph.D. degrees in computer science from Peking University, Beijing, China, in 1988, 1991, and 2005, respectively. She is currently a Full Professor with the School of Computer Science, Peking University. She has authored or coauthored more than 200 research papers on text mining and machine learning in the top journals and conferences. She is a leading author of several textbooks on data structures and algorithms in Chinese and the corresponding course is awarded as the National Elaborate Course, National Boutique Resource Sharing Course, National Fine-designed Online Course, and National First Class Undergraduate Course by Ministry of Education China. Prof. Zhang is a Member of the Advisory Committee of the Ministry of Education in China. She is one of the 15 members of the ACM/IEEE CC2020 Steering Committee. She was the recipient of the Best Paper Award of the International Conference on Machine Learning 2014 and Best Paper Nominee of World Wide Web 2016.