Towards Semi-supervised Universal Graph Classification

Xiao Luo*, Yusheng Zhao*, Yifang Qin*, Wei Ju, and Ming Zhang

Abstract— Graph neural networks have pushed state-of-the-arts in graph classifications recently. Typically, these methods are studied within the context of supervised end-to-end training, which necessities copious task-specific labels. However, in real-world circumstances, labeled data could be limited, and there could be a massive corpus of unlabeled data, even from unknown classes as a complementary. Towards this end, we study the problem of semi-supervised universal graph classification, which not only identifies graph samples which do not belong to known classes, but also classifies the remaining samples into their respective classes. This problem is challenging due to a severe lack of labels and potential class shifts. In this paper, we propose a novel graph neural network framework named UGNN, which makes the best of unlabeled data from the subgraph perspective. To tackle class shifts, we estimate the certainty of unlabeled graphs using multiple subgraphs, which facilities the discovery of unlabeled data from unknown categories. Moreover, we construct semantic prototypes in the embedding space for both known and unknown categories and utilize posterior prototype assignments inferred from the Sinkhorn-Knopp algorithm to learn from abundant unlabeled graphs across different subgraph views. Extensive experiments on six datasets verify the effectiveness of UGNN in different settings.

Index Terms—Graph Neural Network, Semi-supervised Learning, OOD Detection.

1 Introduction

Graphs have garnered growing interest due to their capacity of portraying structured and relational data in a large range of domains. As one of the most prevalent graph machine learning problems, graph classification aims to predict the properties of whole graphs, which has widespread applications in visual and biological systems [1]–[4]. In recent years, graph neural networks (GNNs) have exhibited promising performance in graph classification [5]–[7], which usually follow the paradigm of message passing [8]–[11]. In detail, node representations are iteratively updated by aggregating neighborhood information, followed by a readout operation to generate graph representations. These graph representations can implicitly capture the structural topology in an end-to-end fashion, therefore facilitating downstream classification effectively.

Despite their exceptional performance, GNNs are heavily reliant on copious task-specific labels while learning graph representation. In various real-world settings, however, large-scale data annotations could need a significant number of human resources [7]. To address this, semi-supervised graph classification approaches have been proposed [12]–[14], which use a huge corpus of unlabeled data to enhance the model performance in an efficient manner. These approaches presume that unlabeled graphs have the same distribution as labeled graphs, which would not be true in practice, particularly when labeled graphs make up a tiny portion of the whole dataset. For example, as in Fig. 1, samples from classes '8' and '9' are unavailable in labeled data. To tackle this, in this research we investigate a more realistic problem named **semi-supervised universal graph classification**, where unlabeled data could belong to unknown classes. Here, two tasks must be carried out: (1) identifying graph samples that do not belong to known classes; and (2) categorizing the remaining samples into their respective classes. These out-of-distribution (OOD) graph samples (i.e., data from unknown classes) could be provided to experts, which increases the efficiency of data annotations.

In reality, this realistic graph classification would face the following essential difficulties: (1) **How can these OOD graph examples be detected in unlabeled data?** The essence of this topic is to identify various samples that belong to unknown classes without adequate prior information. Typically, the bulk of conventional OOD contexts in computer vision assume that OOD samples are only engaged during assessment [15]–[17], but our problem focuses on their involvement during training. Even worse, this problem needs to deal with heterogeneous information in a large number of networks, i.e., node characteristics and structural topology, which makes discriminating between in-distribution (ID) and out-of-distribution (OOD) samples more challenging. (2) **How to overcome the label scarcity in the training data?** In reality, labeled data is scarce owing to the prohibitive expense of data annotation, but unlabeled data is abundant. Existing semi-supervised approaches often produce pseudo-labels to help optimize GNNs [13], [14]. Nevertheless, these pseudo-labels could be biased and imbalanced, particularly when OOD graph samples exist. Note that resolving these

- *Xiao Luo is with University of California, Los Angeles, USA. (e-mail: xiaoluo@cs.ucla.edu)*
- *Yusheng Zhao, Wei Ju, Yifang Qin and Ming Zhang are with Peking University, Beijing, China. (e-mail: yusheng.zhao@stu.pku.edu.cn, juwei@pku.edu.cn, qinyifang@pku.edu.cn, mzhang_cs@pku.edu.cn)*
- *Xiao Luo, Yusheng Zhao and Yifang Qin contribute equally to this work.*

- *This paper is partially supported by the National Key Research and Development Program of China with Grant No. 2018AAA0101902 as well as the National Natural Science Foundation of China (NSFC Grant No. 62106008 and No. 62006004).*

Manuscript received April 19, 2005; revised August 26, 2015.

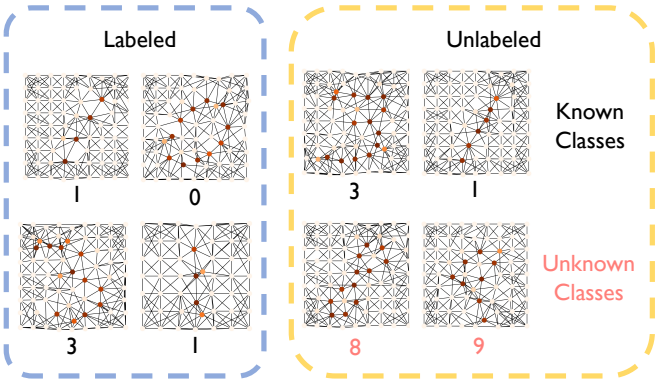


Fig. 1: An illustration of our problem setting. We are given both labeled graphs and unlabeled graphs which could contain samples from unknown classes.

obstacles might be mutually beneficial. On the one hand, the precise identification of OOD graph samples enables the best use of unlabeled ID graphs. On the other hand, exploring sufficient semantics from unlabeled data can assist in the discovery of OOD graph examples.

In this study, we present a novel framework Universal Graph Neural Network (UGNN) that overcomes the aforementioned difficulties from the subgraph perspective. In particular, to produce graph-level representations, we first warmed up the message passing neural network with labeled graph samples. To combat class shift, we adopt a simple yet effective selection strategy, which samples numerous subgraphs to capture both prediction confidence and individual output uncertainty based on the calibration of GNNs. Our strategy computes both the average and the variance of prediction confidence scores among various subgraphs, and then sets an adaptive threshold to distinguish OOD samples from easy to hard. In addition, to make the most of unlabeled data, we construct graph prototypes in the embedding space for both known and unknown classes. Then, a semi-supervised prototype representation learning paradigm is developed, which utilizes the posterior prototype assignments from one subgraph view to supervise the semantics of unlabeled data from another view. The Sinkhorn-Knopp algorithm [18]–[20] is involved to promise balanced and soft posterior distributions. Our OOD sample selection technique and prototype-aware semi-supervised learning paradigm could mutually strengthen each other, enabling optimal use of unlabeled data. To demonstrate the efficacy of our UGNN, we conduct extensive experiments on six benchmark graph classification datasets. The results demonstrate that UGNN outperforms a number of state-of-the-art models in a range of settings. The contributions of our work are summarized as follows:

- **Problem Formalization:** We investigate the problem of semi-supervised universal graph classification, which facilities data annotation efficiency in real-world applications.
- **Novel Methodologies:** We propose a simple yet effective method named UGNN to solve the problem. On the one hand, it captures individual output uncertainties by sampling multiple subgraphs to detect OOD graph samples. On the other hand, a semi-supervised prototype representation learning paradigm employs the posterior prototype assignments to supervise the semantics of unlabeled graphs across two subgraph views.
- **Multifaceted Experiments:** Extensive experiments on six graph classification datasets to validate the efficacy of the proposed UGNN in different settings.

The related works are introduced in Section 2. In Section 3 and Section 4, we describe the prior knowledge and the details of our UGNN, respectively. Section 5 offers extensive experimental results including quantitative comparisons, ablation studies, parameter sensitivity and visualization. In the end, we give a conclusion in Section 6.

2 Related Work

2.1 Graph Neural Networks

Graph neural networks (GNNs) have shown outstanding performance in relational data representation learning [21], which has been extensively adopted in a number of applications, including node classification [22]–[24], link prediction [25]–[27], and anomaly detection [28]–[30]. Early efforts [31]–[33] usually utilize spectral GNNs based on the spectral graph theory, which begin with transferring graph signals into the embedding space, followed by spectral filters deduced from the graph Laplacian. Recent spatial methods have become the mainstream due to their lower computational complexity [8], [34], [35]. Typically, they adhere to the message passing paradigm, in which each node receives data from its neighbors, followed by an aggregation process that continually updates the node representations. GNNs have also been used regularly for graph classification. Typically, these methods use graph pooling functions to summarize node representations into graph-level representations [3], [4]. For example, SAG Pooling [36] utilizes the attention technique to preserve important nodes in a hierarchical fashion. Despite their promising performance, these methods are data-hungry whereas real-world applications often consist of limited labeled data and massive unlabeled data containing OOD graph samples. Towards this end, we investigate the semi-supervised universal graph classification problem, which not only identifies graph samples that do not belong to known classes but also classifies the remaining samples into their respective classes.

2.2 Semi-supervised Graph Classification

Semi-supervised learning has received increasing attention in recent years. Pseudo-labeling is a popular technique which predicts the label distribution of unlabeled examples and selects confident samples for further guidance. For example, FlexMatch [37] introduces class-specific adaptive thresholds to decide confident samples inspired by curriculum learning. Ada-CM [38] further utilizes contrastive learning to explore unconfident samples in the unlabeled set. Another line of semi-supervised learning is consistency learning. FixMatch [39] is a simple method which combines semi-supervised learning and consistency learning, achieving superior performance in this field. The objective of semi-supervised graph classification is to predict graph properties using both labeled data and unlabeled data, which

accounts for real-world label scarcity [12]–[14], [40], [41]. Typically, existing methods combine graph neural networks with semi-supervised learning techniques. Early attempts often use pseudo-labeling approaches [12], which annotate unlabeled graphs using the classification model itself, and then add samples along with highly confident predictions to the training set. Unfortunately, these approaches may produce overconfident and skewed pseudo-labels, which could lead to an accumulation of errors during subsequent optimization. Considering the complexity of learning graph-level representations, recent approaches often use the multi-task learning framework where the teacher model attempts to learn discriminative graph representations, whereas the student model concentrates on the classification task [13], [14]. However, these methods do not consider the realistic problem of potential OOD graph samples, which brings challenges in the sufficient exploration of unlabeled data. To tackle this, our UGNN employs a sample selection strategy, which captures prediction confidence as well as individual output uncertainty from the subgraph perspective.

2.3 Out-of-distribution Detection

Out-of-distribution (OOD) detection has been widely utilized in a variety of real-world applications [42]. Typically, this problem is addressed in two contexts, resulting in supervised methods and unsupervised methods. With access to identified OOD samples during optimization, supervised algorithms [15]–[17] typically not only reduce the cross-entropy loss for ID samples, but also enforce the uniformity of the prediction distributions for OOD samples. In practical applications, it is difficult to locate OOD samples in advance. To circumvent this issue, unsupervised approaches do not use OOD samples during training [43]–[45]. They deploy post-hoc detectors based on distance measurements such as Mahalanobis distance [46] after training classification models using ID samples. Despite the impressive performance of OOD detection in computer vision, its application to graphs remains underexplored. In contrast, we offer a novel graph neural network named UGNN that thoroughly both explores subgraphs to find OOD graph samples and learns from unlabeled graphs.

3 Preliminary Knowledge

3.1 Problem Formalization

To begin with, we formally introduce the notations and the problem definition. Here a graph containing n nodes is denoted as $G = (\boldsymbol{A}, \boldsymbol{X})$, where $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ represents the adjacent matrix, $\boldsymbol{X} \in \mathbb{R}^{n\times d}$ represents the node attribute matrix and d is the attribute dimension. In the problem of semi-supervised universal graph classification, we are given a training dataset $\mathcal{D}$ containing labeled graphs $\mathcal{D}^l = \{G_1^l, G_2^l, \cdots, G_{N^l}^l\}$ and unlabeled graphs $\mathcal{D}^u = \{G_1^u, G_2^u, \cdots, G_{N^u}^u\}$, where G_i^l and G_j^u represents the i-th labeled sample and the j-th unlabeled sample, respectively. The label set of the labeled data and the whole training data are denoted as C^l and C, respectively. $y_i^l \in C^l$ denotes the label of G_i^l. Due to the potential label shifts, we have $C^l \subseteq C$.

Our aim is to (1) identify graph samples which do not belong to known classes, i.e., $\mathcal{S} = \{G_j^u | y_j^u \in C/C^l\}$ and (2) classify the remaining samples, i.e., $\mathcal{D}^u/\mathcal{S}$ into their corresponding classes in C^l.

3.2 Message Passing Neural Networks

We briefly introduce message passing neural networks, which are widely utilized to generate graph-level representations [8], [34], [35]. They usually utilize the neighborhood aggregation mechanism to explore topological information in an implicit fashion. The updating formulation at the k-th layer in a given graph G is written as:

$$\begin{aligned} \boldsymbol{v}_{N(v_i)}^{(k)} &= \mathrm{AGGREGATE}^{(k)}\left(\left\{\boldsymbol{v}_i^{(k-1)} : j \in \mathcal{N}(i)\right\}\right) \\ \boldsymbol{v}_i^{(k)} &= \mathrm{COMBINE}^{(k)}\left(\boldsymbol{v}_i^{(k-1)}, \boldsymbol{v}_{N(v_i)}^{(k)}\right) \end{aligned} \tag{1}$$

where $\boldsymbol{v}_i^{(k)}$ is the representation of node v_i at the k-th layer. $\mathrm{AGGREGATE}^{(k)}(\cdot)$ and $\mathrm{COMBINE}^{(k)}(\cdot)$ represent the aggregation and combination function at the k-th layer, respectively. Finally, a global pooling function is utilized to summarize all the node representations at the last layer, resulting in a graph-level representation:

$$\boldsymbol{z} = \mathrm{GP}\left(\left\{\boldsymbol{v}_i^{(K)}\right\}_{i=1}^{n}\right), \tag{2}$$

where $\mathrm{GP}(\cdot)$ represents a global pooling function.

4 Methodology

This paper proposes a novel graph neural network framework named UGNN for semi-supervised universal graph classification. The core of our UGNN is to utilize subgraphs to sufficiently explore the semantics in unlabeled graphs. We first warm up our GNN-based encoder using labeled data to generate graph representations. To overcome label shift, we employ a sample selection strategy, which calculates confidence scores from the distribution viewpoint by sampling multiple subgraphs. To make the most of unlabeled data, we measure graph prototypes for both known and unknown classes, which can yield balanced and reliable prototype assignments by solving an optimization problem. Then, a semi-supervised graph prototype representation learning paradigm is presented, which utilizes the posterior prototype assignments from one subgraph view to supervise the semantics of unlabeled data from another view. More details can be illustrated in Fig. 2.

4.1 Graph Representation Learning

Our model needs to extract topological information from both labeled and unlabeled graphs for the classification task with potential label shifts. Therefore, learning effective graph representations is crucial for our problem. Towards this end, we leverage a message passing neural network to encode graphs into low-dimensional embeddings. In addition, a hierarchical graph pooling structure is adopted to explore local substructures in the graph.

In detail, given a graph sample $G = (\boldsymbol{A}, \boldsymbol{X})$, we first utilize the message passing neural network in Equation 1 to extract topological information, resulting in discriminative node representations, i.e., $\{\boldsymbol{v}_i^{(K)}\}$. Then, we follow the paradigm of TopK-based pooling by utilizing the attention mechanism to identify crucial nodes which will be kept.

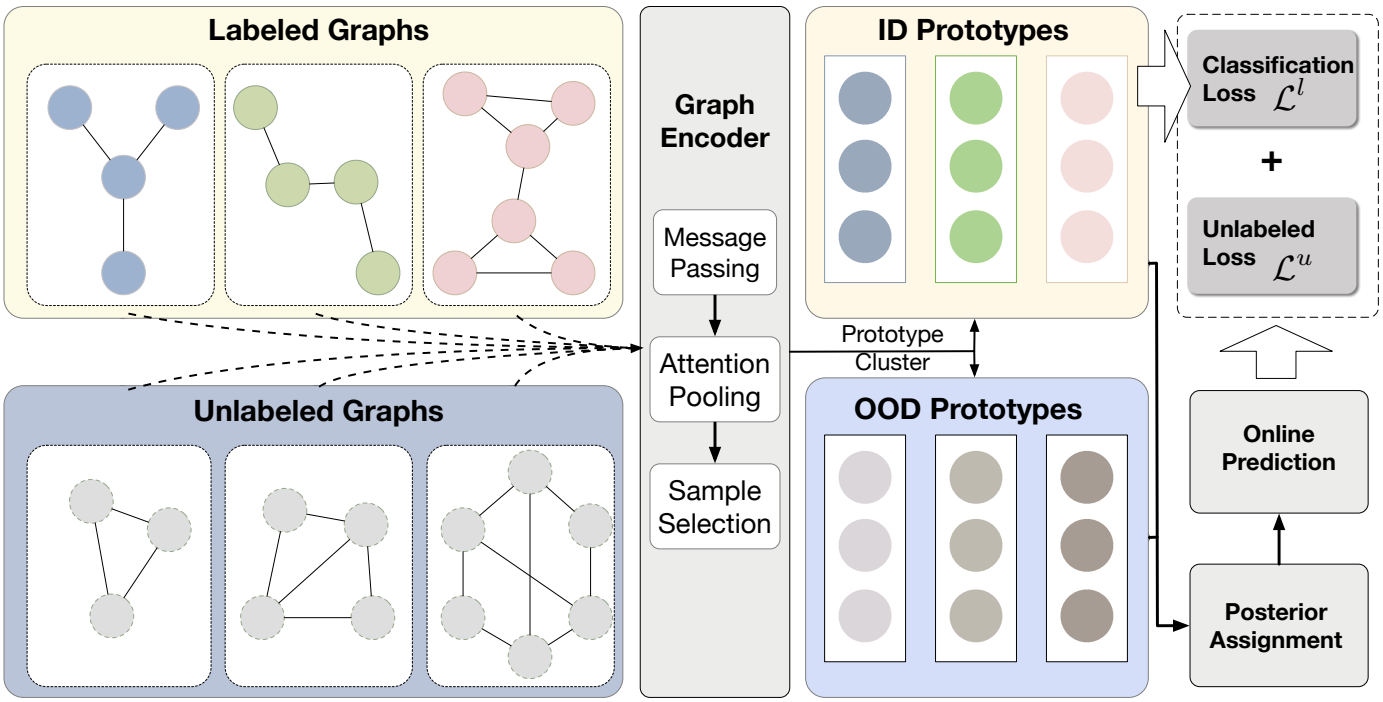


Fig. 2: Illustration of the proposed framework UGNN. Our UGNN is first warmed up using labeled data and then identifies OOD samples based on multiple subgraphs from each graph. Besides, UGNN constructs graph prototypes and compares the posterior prototype assignments with online prediction across different views.

Here, we utilize a different encoder to produce an importance score vector $\boldsymbol{S} \in \mathbb{R}^{n\times 1}$ for nodes in the graph. The top $\lceil \rho n \rceil$ nodes will be kept by comparing the values in $\boldsymbol{S}$. Let idx denote the index of kept nodes, and we derive pooled graph with the adjacent matrix $\tilde{\boldsymbol{A}}$ and the hidden embedding matrix $\tilde{\boldsymbol{H}}$ as follows:

$$\tilde{\boldsymbol{V}} = \boldsymbol{V}_{idx,:} \odot \boldsymbol{S}_{idx}, \tilde{\boldsymbol{A}} = \boldsymbol{A}_{idx,idx}, \tag{3}$$

where $\boldsymbol{V}_{idx,:}$ denotes stacked node representation matrix and $\odot$ represents the broadcasted Hadamard product [36], [47]. $\boldsymbol{A}_{idx,idx}$ represents the row-wise and column-wise indexed matrix of $\boldsymbol{A}$. Then, the pooled graph is fed to the message passing neural network again, followed by the readout function as in Equation 2, producing a graph-level representation $\Phi(G)$. In this way, we summarize node semantics embedded in a hierarchical fashion, maximizing the model capacity for graph representation learning.

Finally, we add an MLP classifier $\psi(\cdot)$ on top of the representations, which produces label distribution for each graph sample. Here the graph classification network is warmed up by merely labeled graph samples. Given a batch of labeled graphs $\mathcal{B}^l \subset \mathcal{D}^l$, the cross-entropy is employed as the classification loss as follows:

$$\mathcal{L}^l = \frac{1}{|\mathcal{B}^l|} \sum_{G_i^l \in \mathcal{B}^l} -\log(\boldsymbol{y}_i^l)^T \psi(\Phi(G_i^l)) \tag{4}$$

where $\boldsymbol{y}_i^l$ denotes the one-hot vector of y_i^l.

4.2 Subgraph-based OOD detection

The first aim of our problem is to detect OOD samples in unlabeled data. Intuitively, graph samples with shaped predicted label distributions are more likely to be ID samples. However, the serious label scarcity of labeled graph data could bring in biased and overconfident predictions. To tackle this, we incorporate multiple subgraphs sampled from the graph input into our OOD detection from the view of both confidence and model calibration.

In particular, we first introduce two perturbation strategies to generate subgraphs [40], [48] as follows: (1) Edge deletion: we randomly remove several edges from a graph obeying an i.i.d uniform distribution. (2) Node deletion: several nodes are removed at random, along with their connected edges. Then, a subgraph set $S = \{\tilde{G}_1, \tilde{G}_2, \cdots, \tilde{G}_I\}$ can be generated for each graph G, where I represents the number of subgraphs. We can derive the confidence score for each subgraph which is defined as the largest probability among the prediction distribution:

$$s_r = \phi(f(\tilde{G}_r)), \tag{5}$$

where $\phi : \mathbb{R}^d \rightarrow \mathbb{N}$ return the index of the maximum value. Here we measure the distribution of confidence score using a normal distribution, i.e., $N(\mu, \sigma^2)$. The mean and the variance among the subgraph set are measured, i.e., $\hat{\mu} = \frac{1}{I}\sum_{i=1}^{I} s_i$ and $\hat{\sigma}^2 = \frac{1}{I}\sum_{i=1}^{I}(s_i - \mu)^2$. In this part, we adopt a hybrid strategy to determine the OOD samples based on both the mean and variance of the confidence distributions.

To begin with, samples with high confidence are more likely to belong to ID samples. Here we utilize the mean of the confidence scores to release the sample biases. Intuitively, high variances for augmented views imply the prediction is not stable. In other words, their augmented views could be dissimilar to samples in the predicted class

to generate a contradiction. Therefore, these samples could come from unseen classes, and we cannot identify them as ID samples. From a different view, compared with OOD samples, ID samples can be more resistant to the noise attack since they are correctly classified. Therefore, we turn to model calibration. In particular, the expected calibration error is directly related to the variance of the prediction empirically [49]. Therefore, we get more emphasis on the samples with low-variance confidence, which are resistant to the potential noise attack. Taking both factors into consideration, the final score for OOD detection is formulated as:

$$\tilde{s} = \hat{\mu} - \hat{\sigma}. \tag{6}$$

A higher score indicates a small probability of being OOD samples. Therefore, the set of ID samples is inferred as follows:

$$\mathcal{D}^{u,id} = \{G | \tilde{s}(G) > \delta\} \tag{7}$$

where δ is a threshold to decide the portion of ID samples. Inspired by curriculum learning, we gradually raise the threshold to identify the OOD samples from easy to hard. For the t-th iteration, we have $\delta_t = \frac{t}{T}\delta$, where T is the total number of iterations.

4.3 Prototype-aware Semi-supervised Learning

In addition, to overcome the label scarcity in the training data, we introduce a novel prototype-aware semi-supervised learning framework, which combines learning discriminative graph representations with semi-supervised learning.

Prototype Initialization. To begin with, we initialize prototypes for both ID samples and OOD samples. On the one hand, we take the average of graph representations from every known category in the embedding space. On the other hand, we cluster the graph representations of OOD samples selected before.

In detail, the prototype representation for the c-th category is defined as:

$$\boldsymbol{h}_c = \frac{\sum_{i=1}^{N^l} \mathbf{1}_{\boldsymbol{y}_i^l = c} \boldsymbol{z}_i}{\sum_{i=1}^{N^l} \mathbf{1}_{\boldsymbol{y}_i^l = c}}, \tag{8}$$

Then, the graph representations of OOD samples are clustered into R parts, and the clustering center is denoted as $\boldsymbol{h}_{C+1}, \cdots, \boldsymbol{h}_{C+R}$. These prototypes from ID and OOD samples jointly enhance semi-supervised learning for discriminative graph representations.

Semi-supervised Learning. To learn from unlabeled data, we first generate their posterior prototype assignments to provide additional knowledge. To prevent generating overconfident distributions, prototype assignments are also inferred from the view of subgraphs, which would guide the semantic learning for unlabeled data.

In detail, given a batch of unlabeled graphs $\{G_j^u\}_{j=1}^{B^u}$, we define matrix $\boldsymbol{Q} \in \mathbb{R}^{E \times B^u} = Eq\left(y = c \mid G_j^u\right)$ where $E = C + R$. We first stack the prototype representations and a batch of subgraph representations into the matrices $\boldsymbol{H} \in \mathbb{R}^{D \times E}$ and $\tilde{\boldsymbol{Z}} \in \mathbb{R}^{D \times B^u}$. To obtain accurate and balanced posterior prototype assignments, we maximize the following objective as follows:

$$\begin{aligned} \mathcal{E} = &\left\langle \mathbf{H}^T \tilde{\boldsymbol{Z}}, \boldsymbol{Q} \right\rangle + \epsilon \mathcal{H}(\boldsymbol{Q}) + < \boldsymbol{f}, \boldsymbol{Q}\mathbf{1}_{B^u} - \frac{1}{E}\mathbf{1}_E > \\ &+ \left\langle \boldsymbol{g}, \boldsymbol{Q}^\top \mathbf{1}_E - \frac{1}{B^u}\mathbf{1}_{B^u} \right\rangle \end{aligned} \tag{9}$$

where $< \mathbf{H}^T \tilde{\boldsymbol{Z}}, \boldsymbol{Q} >$ returns the trace of $\tilde{\boldsymbol{Z}}^T \mathbf{H} \boldsymbol{Q}$ and $\mathcal{H}(\boldsymbol{Q}) = -\sum_{i,j} \boldsymbol{Q}_{i,j} \log \boldsymbol{Q}_{i,j}$ denotes the entropy of the matrix. $\boldsymbol{f} \in \mathbf{R}^{B^u \times 1}$ and $\boldsymbol{g} \in \mathbf{R}^{E \times 1}$ are two adaptive Lagrange multipliers. In Equation 9, the first term maximizes the similarity between $\boldsymbol{Q}$ and $\mathbf{H}^T \tilde{\boldsymbol{Z}}$, which denotes the expected similarity scores between graph representations are their representations. The second term is a regularization term to maximize the entropy of $\boldsymbol{Q}$, which encourages the diversity of the posterior prototype assignment. ϵ is a temperature parameter to control the diversity. The last two terms are Lagrange constraints, which aim to produce $\boldsymbol{Q}$ with both row and column normalization to produce balanced posterior distributions [18], [19].

To maximize Equation 9, we first calculate its gradient with respect to every element $\boldsymbol{Q}_{ij}$ as follows:

$$\frac{\partial \mathcal{E}}{\partial (\boldsymbol{Q}_{ij})} = 2[\mathbf{H}^T \tilde{\boldsymbol{Z}}]_{ij} - \epsilon \log (\boldsymbol{Q})_{ij} + \boldsymbol{f}_i + \boldsymbol{g}_j \tag{10}$$

Therefore, the closed solution is written as:

$$\mathbf{Q}^* = \mathrm{Diag}(\mathbf{u}) \exp\left(\frac{2\mathbf{H}^T \tilde{\boldsymbol{Z}}}{\epsilon}\right) \mathrm{Diag}(\mathbf{v}) \tag{11}$$

where $\mathbf{u} = \mathrm{diag}\left(\exp\left(\frac{\boldsymbol{f}}{\epsilon}\right)\right)$ and $\mathbf{v} = \mathrm{diag}\left(\exp\left(\frac{\boldsymbol{g}}{\epsilon}\right)\right)$. In practice, we recall the constraints embedded in Equation 9, i.e., $\boldsymbol{Q}\mathbf{1}_{B^u} - \frac{1}{E}\mathbf{1}_E = 0$ and $\boldsymbol{Q}^\top \mathbf{1}_E - \frac{1}{B^u}\mathbf{1}_{B^u} = 0$ and utilize the iterative Sinkhorn-Knopp algorithm [18]–[20] to solve this, which repeatedly conducts row normalization and column normalization to $\exp\left(\frac{2\mathbf{H}^T \tilde{\boldsymbol{Z}}}{\epsilon}\right)$. Preliminary studies indicate that using three iterations would achieve incredible performance with less computational expense, and that soft target assignments have a higher performance than one-hot ones.

Then, these prototype assignments are viewed as guidance to learn from unlabeled data. In particular, the cross-entropy loss objective is written as:

$$\mathcal{L}^u = -\frac{1}{B^u} \sum_{j=1}^{B^u} \sum_{c=1}^{E} q\left(y = c \mid G_j^u\right) \log p\left(y = c \mid G_j^u\right) \tag{12}$$

where online predictions are calculated by

$$p\left(y = c \mid G_j^u\right) = \frac{\exp\left(\tilde{\tilde{\boldsymbol{z}}}_j^{u\top} \boldsymbol{h}_c / \tau\right)}{\sum_{c'=1}^{E} \exp\left(\tilde{\tilde{\boldsymbol{z}}}_j^{u\top} \boldsymbol{h}_c / \tau\right)} \tag{13}$$

Here $\tilde{\tilde{\boldsymbol{z}}}_j^u$ is the other subgraph representations of G_j^u and τ is the temperature parameter. In this section, two different perturbations are involved for posterior prototype assignments and online predictions, which helps to capture the invariant semantics in unlabeled graphs with less bias.

Prototype Update. Here, we update the prototype representations as the training process of semi-supervised learning. In particular, we aggregate all the graph representations which are close to each prototype using momentum update.

TABLE 1: Statistics of the datasets used in the experiments.

Dataset	# Graphs	# Classes	Avg. # Nodes	Avg. # Edges	# Known	# Unknown	Node feature	Low Ratio	High Ratio
COIL-DEL	3900	100	21.54	54.24	80	20	Coordinates	50%	80%
Letter-High	2250	15	4.67	4.50	10	5	Coordinates	20%	40%
MNIST	55,000	10	70.6	564.5	7	3	Pixel (Gray) + Coordinates	1%	3%
CIFAR10	45,000	10	117.6	941.2	7	3	Pixel (RGB) + Coordinates	30%	70%
REDDIT-MULTI-12K	11929	11	391.41	456.89	7	4	-	30%	70%
COLORS-3	10500	11	61.31	91.03	7	4	Colors	30%	80%

Algorithm 1 Training Algorithm of UGNN

Require: Labeled graphs $\mathcal{D}^l$; Unlabeled graphs $\mathcal{D}^u$;
Ensure: Identify OOD graph samples $\mathcal{S}$ and generate the prediction for $\mathcal{D}^u/\mathcal{S}$;
Warm up the network using $\mathcal{D}^l$;
for $t = 1, 2, \cdots, T$ **do**
  Generate $\mathcal{D}^{u,id}$ using Equation 7;
  Initialize graph prototype representations using Equation 8;
  repeat
    Construct a mini-batch by sampling graphs from $\mathcal{D}^l$ and $\mathcal{D}^u$;
    Generate the posterior distribution using the Sinkhorn-Knopp algorithm;
    Calculate the final loss using Equation 15;
    Update the network parameters through back propagation;
    Update the prototype representations using Equation 14;
  until convergence
end for

Formally, we first generate the pseudo-labels for each unlabeled graph, i.e., $p_j = argmax_c\{\boldsymbol{h}_j^T \boldsymbol{z}_c\}$ and the updated prototypes are derived as:

$$\boldsymbol{h}_c \leftarrow \eta \boldsymbol{h}_c + (1-\eta)\frac{\sum_{j=1}^{N^u} \mathbf{1}_{p_j=c} \boldsymbol{z}_j^u}{\sum_{j=1}^{N^u} \mathbf{1}_{p_j=c}} \tag{14}$$

where η is a momentum coefficient.

4.4 Optimization

In a nutshell, the overall training loss of our UGNN is summarized into:

$$\mathcal{L} = \mathcal{L}^l + \mathcal{L}^u \tag{15}$$

We optimize the whole framework using mini-batch stochastic gradient descent. For every cycle, we update the set of ID unlabeled samples using curriculum learning and the number of total cycles is T. The detailed algorithm is summarized in Algorithm 1.

5 Experiments

In this section, exhaustive experiments are conducted on several datasets to demonstrate the effectiveness of the proposed UGNN. Particularly, we are interested in several research questions (RQs):

- **RQ 1**: What is the overall performance of UGNN compared to baseline methods?
- **RQ 2**: What is the influence of Subgraph-based OOD Detection and Prototype-aware Semi-supervised Learning in the proposed task?
- **RQ 3**: Do the proposed model sensitive to hyperparameters like the number of clusters, the presumed number of OOD samples and the temperature in prototype-aware semi-supervised learning?
- **RQ 4**: Are there any visualization of classification results and learned representations to show the effectiveness of UGNN?

5.1 Experimental Setup

5.1.1 Datasets

In the experiments, we use six public graph datasets: COIL-DEL, Letter-high, MNIST, CIFAR10, REDDIT-MULTI-12K and COLORS-3. The detailed statistics of the datasets are listed in Table 1.

COIL-DEL. COIL-DEL dataset [50] is constructed by applying Harris corner detection and Delaunay Triangulation on images. The result of triangulation is converted to a graph, where nodes and edges represent ending points and lines.

Letter-high. Letter-high dataset [50] involves graph representations of 15 capital letters (*i.e.* A, E, F, H, I, K, L, M, N, T, V, W, X, Y, Z). In the graph, nodes represent the endpoints of the drawing and (undirected) edges correspond to lines. The letters are highly distorted, which makes the task challenging.

MNIST. MNIST dataset [51] is constructed by extracting super-pixels (small regions of homogeneous intensity in the images) of the image. After the super-pixel extraction, a k-nearest-neighbor graph is then constructed to represent the image.

CIFAR10. CIFAR10 dataset [51] is constructed in the same way as MNIST. The difference between the two datasets is that CIFAR10 contains larger graphs with richer semantic meanings, which makes the classification task more challenging.

REDDIT-MULTI-12K. REDDIT-MULTI-12K dataset [33] contains graphs where nodes denote users and edges denote comments. The aim is to classify graphs into different communities.

COLORS-3. COLORS-3 dataset [52] contains random graphs where each node shows one color from red, green and blue and we aim to capture the number of green nodes in each graph.

5.1.2 Evaluation Setting

We split the classes into known classes and unknown classes, and the number of each is shown in Table 1. In the semi-supervised universal graph classification task, we only use *some* of the labels in the known classes. All instances in the unknown classes are *unlabeled*. In order to measure the models' performance in different application scenarios, we adopt two settings with different labeling ratios (*i.e.* High Ratio and Low Ratio) and high labeling ratio/low labeling ratio is between 1.6 and 3. For example, on the MNIST dataset, we use 1% of the labels and 3% of the labels as high labeling ratio and low labeling ratio, respectively. More details can be found in Table 1.

As for the evaluation metric, we use the classification accuracy as the default. In the proposed semi-supervised universal graph classification setting, a graph is classified correctly if and only if (i) the graph belongs to the source classes and the model predicts the correct label, or (ii) the graph belongs to the novel classes and the model detects it as the novel instance.

5.1.3 Baseline Methods

We compare the proposed UGNN with a wealth of baselines, ranging from kernel-based methods to graph neural networks and semi-supervised graph classification methods. The detailed baseline methods are described as follows:

Graph Kernel Methods. In the experiments, we take three graph kernels for comparison, including:

- Weisfeiler-Lehman (WL) Kernel [53] that utilizes the Weisfeiler-Lehman algorithm to generate features of nodes that are compared across graphs.
- Shortest-Path (SP) Kernel [54] that decomposes graphs into shortest paths and compares pairs of shortest paths based on their lengths.
- Graphlet Kernel [55] that counts the graphlets in the input graphs and generates features according to their occurrence.

We use labeled data to fit a Support Vector Machine (SVM) with graph kernels and then make predictions with this SVM classifier.

Graph Convolutional Neural Networks. We also adopt a variety of graph convolutional layers, including Graph Convolutional Network (GCN) [56], GraphSAGE [57], Graph Isomorphic Network (GIN) [58]. When applying different graph convolutions, we use TopK Pooling [59] as the default graph pooling method.

Graph Pooling Methods. As for graph pooling methods, we select three graph pooling methods as baselines. Specifically, we use:

- TopK Pooling [59] that is described in Sec 4.1.
- Self-Attention Graph Pooling (SAG Pooling) [36] that is based on self-attention to jointly consider both node features and graph topology.
- Adaptive Structure Aware Pooling (ASAP) [60] that utilizes self-attention and learns soft cluster assignments for each node to pool the graph.

When comparing different graph pooling methods, we keep graph convolution method as the default GIN.

Semi-supervised Graph Classification Methods. Contrastive learning is widely used in graph classification in semi-supervised settings. Therefore, we select three graph contrastive learning methods and one knowledge distillation method as our baselines.

- InfoGraph [61] that incorporates a teacher encoder and a student encoder trained in supervised and self-supervised manners, respectively. Their knowledge is transferred by maximizing the mutual information.
- GraphCL [62] that adopts graph augmentations and normalized temperature-scaled cross entropy loss (NT-xent) to learn generalizable, transferable and robust representations using contrastive learning.
- GLA [63] that designs label-invariant augmentations in the representation space, and achieves promising results in semi-supervised graph classification task.
- RGCL [64] that utilizes invariant rationale discovery to separate the graph into two parts. These two part would be fed into the contrastive learning framework to learn effective graph representations.

5.1.4 Implementation Details

For graph representation learning, we use GIN convolution [58] as default. In subgraph-based OOD detection, we obtain the subgraphs by randomly deleting 20% of the nodes and the corresponding edges. We extract a total of 3 subgraphs from the original graph (*i.e.* $I = 3$) to empirically compute the mean and variance. For prototype-aware semi-supervised learning, the OOD samples are clustered into 3 parts (*i.e.* $R = 3$). We set ϵ in Eq. 9 to 0.05 and the softmax temperature τ in Eq. 13 to 0.1 following [18]. The momentum coefficient η is set to 0.99 as in [65]. During optimization, we also use an additional supervised contrastive loss function in [66] to help the training. The proposed UGNN is implemented with PyTorch and can be trained on an NVIDIA RTX GPU. We train the model for 100 epochs in which the first 50 epochs are used to warm-up the model with labeled data. We use Adam optimizer [67] with a learning rate of 0.001 and the batch size is set to 256.

5.2 Main Results (RQ 1)

The classification accuracy and F1 score of UGNN in comparison with various baseline methods are shown in Table 2 and Table 3. From the results, we have several observations.

Firstly, the proposed UGNN achieves a consistent improvement compared to all baseline methods in both low labeling ratio (Low Ratio columns in the table) and high labeling ratio (High Ratio columns in the table) scenarios on all four datasets. The significant improvement demonstrates the effectiveness of the proposed Subgraph-based OOD Detection and Prototype-aware Semi-supervised Learning. More specifically, we attribute the performance gain to the following aspects: (i) UGNN is better at detecting OOD samples. While baseline methods use prediction confidence as the metric for classifying known and unknown categories, our model tackles this problem from a subgraph perspective, which yields more robust OOD detection. (ii) The proposed method provides consistent classification benefiting from learning both known and unknown prototypes,

TABLE 2: Classification accuracy of different labeling ratios on six datasets. Our UGNN achieves the best performance significantly.

Model	COIL-DEL		Letter-High		MNIST		CIFAR10		REDDIT-MULTI-12K		COLORS-3	
	Low Ratio	High Ratio	Low Ratio	High Ratio	Low Ratio	High Ratio	Low Ratio	High Ratio	Low Ratio	High Ratio	Low Ratio	High Ratio
WL Kernel	0.068±0.000	0.087±0.001	0.444±0.001	0.511±0.000	0.142±0.001	0.178±0.002	0.133±0.001	0.139±0.002	0.159±0.000	0.164±0.001	0.104±0.001	0.106±0.000
SP Kernel	0.039±0.004	0.053±0.002	0.131±0.002	0.142±0.004	0.142±0.001	0.144±0.006	0.139±0.003	0.133±0.004	0.156±0.006	0.165±0.005	0.097±0.003	0.102±0.007
Graphlet Kernel	0.035±0.002	0.049±0.001	0.129±0.002	0.156±0.004	0.118±0.001	0.139±0.001	0.101±0.000	0.110±0.000	0.114±0.002	0.129±0.003	0.090±0.001	0.095±0.004
GCN	0.226±0.025	0.335±0.012	0.329±0.039	0.504±0.018	0.200±0.004	0.370±0.005	0.309±0.010	0.365±0.008	0.312±0.006	0.334±0.005	0.319±0.007	0.383±0.008
GraphSAGE	0.377±0.032	0.397±0.015	0.409±0.007	0.582±0.004	0.196±0.002	0.420±0.019	0.329±0.006	0.351±0.002	0.244±0.004	0.262±0.004	0.313±0.016	0.340±0.019
GIN	0.412±0.016	0.445±0.013	0.489±0.006	0.571±0.005	0.297±0.008	0.626±0.002	0.272±0.019	0.333±0.007	0.347±0.003	0.3583±0.004	0.311±0.006	0.335±0.008
ASAP	0.553±0.006	0.601±0.023	0.567±0.017	0.609±0.006	0.482±0.006	0.541±0.004	0.386±0.009	0.395±0.004	0.335±0.008	0.367±0.012	0.321±0.003	0.343±0.009
SAG Pooling	0.410±0.008	0.485±0.040	0.462±0.058	0.522±0.003	0.425±0.001	0.632±0.003	0.364±0.015	0.388±0.021	0.328±0.006	0.347±0.008	0.391±0.005	0.402±0.010
InfoGraph	0.524±0.006	0.547±0.019	0.514±0.012	0.550±0.011	0.519±0.015	0.632±0.008	0.362±0.004	0.404±0.007	0.320±0.011	0.335±0.021	0.379±0.007	0.390±0.009
GraphCL	0.563±0.013	0.606±0.024	0.562±0.006	0.635±0.005	0.498±0.015	0.700±0.013	0.373±0.006	0.410±0.008	0.327±0.004	0.360±0.005	0.382±0.007	0.399±0.004
GLA	0.565±0.005	0.610±0.013	0.602±0.016	0.631±0.016	0.483±0.015	0.705±0.014	0.383±0.004	0.412±0.009	0.341±0.009	0.368±0.002	0.365±0.008	0.378±0.005
RGCL	0.572±0.004	0.608±0.012	0.587±0.008	**0.697**±0.012	0.475±0.013	0.714±0.011	0.356±0.009	0.406±0.005	0.345±0.012	0.366±0.007	0.390±0.008	0.409±0.004
UGNN (ours)	**0.594**±0.021	**0.630**±0.007	**0.640**±0.010	0.660±0.007	**0.585**±0.004	**0.730**±0.012	**0.397**±0.004	**0.420**±0.003	**0.377**±0.006	**0.383**±0.008	**0.415**±0.003	**0.434**±0.005

TABLE 3: F1 score of OOD detection on six datasets. Our UGNN achieves the best performance significantly.

Model	COIL-DEL		Letter-High		MNIST		CIFAR10		REDDIT-MULTI-12K		COLORS-3	
	Low Ratio	High Ratio	Low Ratio	High Ratio	Low Ratio	High Ratio	Low Ratio	High Ratio	Low Ratio	High Ratio	Low Ratio	High Ratio
GCN	0.190±0.047	0.214±0.011	0.301±0.054	0.342±0.021	0.241±0.004	0.379±0.014	0.216±0.005	0.238±0.007	0.322±0.004	0.358±0.010	0.429±0.008	0.559±0.005
GraphSAGE	0.207±0.052	0.216±0.054	0.389±0.011	0.342±0.032	0.185±0.012	0.194±0.006	0.273±0.012	0.289±0.009	0.300±0.002	0.322±0.006	0.497±0.017	0.620±0.013
GIN	0.255±0.011	0.274±0.005	0.424±0.011	0.417±0.009	0.213±0.005	0.404±0.004	0.237±0.009	0.255±0.011	0.341±0.002	0.361±0.004	0.542±0.004	0.562±0.014
ASAP	0.270±0.013	0.317±0.024	0.462±0.016	0.518±0.008	0.427±0.006	0.462±0.003	0.251±0.004	0.265±0.006	0.310±0.006	0.321±0.010	0.387±0.005	0.416±0.011
SAG Pooling	0.222±0.016	0.250±0.012	0.516±0.027	0.404±0.008	0.401±0.002	0.412±0.002	0.245±0.012	0.286±0.024	0.352±0.012	0.364±0.015	0.597±0.009	0.633±0.005
InfoGraph	0.232±0.014	0.250±0.016	0.459±0.006	0.511±0.007	0.443±0.019	0.506±0.015	0.264±0.004	0.265±0.006	0.317±0.012	0.365±0.015	0.547±0.016	0.562±0.019
GraphCL	0.257±0.013	0.314±0.016	0.524±0.008	0.517±0.017	0.446±0.016	0.498±0.022	0.260±0.002	0.288±0.006	0.320±0.011	0.377±0.007	0.541±0.016	0.632±0.009
GLA	0.281±0.012	0.296±0.041	0.520±0.029	0.526±0.005	0.440±0.008	0.509±0.073	0.291±0.009	0.294±0.004	0.356±0.003	0.383±0.004	0.570±0.009	0.577±0.012
RGCL	0.282±0.003	0.305±0.018	0.502±0.033	0.520±0.015	0.439±0.011	0.510±0.007	0.285±0.014	0.237±0.009	0.326±0.008	0.365±0.010	0.562±0.013	0.603±0.004
UGNN (ours)	**0.319**±0.088	**0.352**±0.025	**0.567**±0.015	**0.563**±0.004	**0.458**±0.005	**0.535**±0.006	**0.313**±0.006	**0.297**±0.007	**0.428**±0.014	**0.443**±0.008	**0.608**±0.007	**0.654**±0.004

whereas baseline methods focus mainly on known categories. (iii) The two components benefit each other. Better out-of-distribution detection provides more accurate information for semi-supervised learning with prototypes. Conversely, more consistent and robust representations helps with finding OOD samples.

Secondly, the proposed UGNN experiences more significant improvements in the low labeling ratio cases (*e.g.* **8.96**% absolute improvement on MNIST with low labeling ratio, compared to **2.59**% absolute improvement with high labeling ratio). This shows that our UGNN is more helpful in the face of label scarcity, which is more common in real-world scenarios.

Thirdly, traditional graph kernel based methods [53]–[55] generally perform worse than graph neural network methods. One exception can be found in the Letter-high dataset, in which Weisfeiler-Lehman kernel achieves comparable accuracy with GNN-based methods. A possible explanation is that the Letter-high dataset contains smaller graphs (4.67 nodes in a graph on average) and relatively simple structures. Graph Neural Networks (GNNs) [36], [56]–[60] boost the performance in graph classification via learning deep representations of graphs as well as utilizing node attributes. Despite their performance gain, they lack the ability to utilize unlabeled data and fall short in detecting OOD samples. Semi-supervised graph classification methods [13], [62]–[64] make use of unlabeled data and thus improve the accuracy. However, they are still weak when dealing with graphs in unknown classes. In contrast, our model detects OOD samples based on subgraphs and learns the prototypes for both known and unknown classes in the semi-supervised setting, which further boosts the performance.

Finally, the overall performance, as well as the model's improvement, of MNIST is higher than CIFAR10, given that we use less data in MNIST than in CIFAR10. The reason is two-fold. Firstly, even for their counterparts in the computer vision domain, CIFAR10 is harder than MNIST. More importantly, the two graph datasets are constructed using super-pixels, which inevitably causes information loss with regard to the detail in the original image. Since the recognition of hand-written digits relies more on the global shape (which is less affected), it is reasonable that graph neural networks perform better on the MNIST dataset.

5.3 Ablation Studies (RQ 2)

In this subsection, we perform ablation studies to verify the effectiveness of Subgraph-based OOD Detection and Prototype-aware Semi-supervised Learning in UGNN. The ablated results are listed in Table 4, where we compare the prediction accuracy of our model and several variants in both low labeling ratio (Low Ratio) and high labeling ratio (High Ratio) scenarios on all the datasets. Moreover, in order to understand the effects of these components in the face of OOD samples, we also report the classification accuracy of unknown classes. Specifically, we compare UGNN to the model (i) without Subgraph-based OOD Detection (w/o S), (ii) without Prototype-aware Semi-supervised Learning (w/o P), (iii) without prototypes of known classes (w/o KP), and (iv) without prototypes of unknown classes (w/o UP).

As can be seen from the results, removing each component results in performance drop in both low labeling ratio and high labeling ratio cases for both overall performance and OOD detection. This demonstrates the effectiveness of Subgraph-based OOD Detection and Prototype-aware Semi-supervised Learning. Concretely, we have several observations listed as follows:

- Without effective OOD detection, the model fails to achieve satisfactory accuracy. As we can see from the second line of Table 4, removing subgraph-based OOD detection causes catastrophic deterioration in classifica-

TABLE 4: Ablation studies on all the datasets. Our full model achieves the best performance consistently.

Experiment	MNIST-low		MNIST-high		COIL-DEL-low		COIL-DEL-high		CIFAR10-low		CIFAR10-high	
	Overall	Unknown	Overall	Unknown	Overall	Unknown	Overall	Unknown	Overall	Unknown	Overall	Unknown
UGNN	**0.585**	**0.485**	**0.730**	**0.585**	**0.594**	**0.313**	**0.630**	**0.337**	**0.397**	**0.278**	**0.420**	**0.290**
w/o S	0.525	0.381	0.646	0.500	0.507	0.216	0.572	0.356	0.383	0.261	0.377	0.293
w/o P	0.488	0.407	0.694	0.554	0.424	0.252	0.503	0.255	0.367	0.266	0.353	0.286
w/o KP	0.490	0.472	0.723	0.561	0.467	0.268	0.525	0.313	0.349	0.247	0.359	0.251
w/o UP	0.530	0.455	0.713	0.567	0.475	0.262	0.549	0.342	0.340	0.238	0.371	0.247

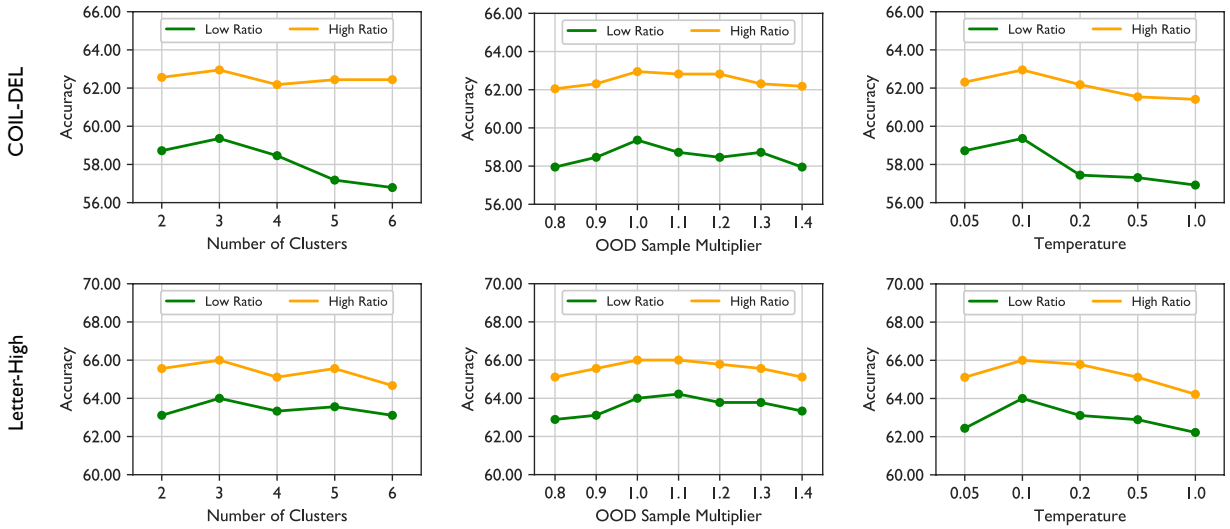


Fig. 3: The parameter sensitivity experiments of UGNN in both low labeling ratio and high labeling ratio cases on two datasets (*i.e.* COIL-DEL and Letter-high). The first column studies the number of novel clusters in Prototype-aware Semi-supervised Learning (*i.e.* the number of prototypes of unknown classes R). The middle column shows the influence of the presumed number of OOD samples. The last column focuses on the temperature in Eq. 13.

tion accuracy of unknown classes (*i.e.* 48.47%→38.13% and 58.53%→49.97% on MNIST).

- Without prototype-aware semi-supervised learning (w/o P, third line), the prediction accuracy falls. Furthermore, it is worth noting that this component is more important when the label is scarce, which is demonstrated by a more significant decline in the low labeling ratio case (*e.g.* 9.68% overall accuracy decline in the low ratio case compared to 3.64% in high ratio case on MNIST). This result is reasonable in that semi-supervised learning aims to utilize unlabeled data, which plays a more important role when labels are scarce.
- Removing either prototypes of known classes (w/o KP, fourth line) or prototypes of unknown classes (w/o UP, fifth line) also hinders the performance, but to a less extent compared to removing all the components (w/o P, third line). This is especially true when the model is provided with more labels. The mitigated performance drop suggests that the prototypes under the semi-supervised learning framework have enough representation power to partially supplant each other.

5.4 Parameter Sensitivity (RQ 3)

In this subsection, we explore the model's sensitivity to hyperparameters. Specifically, we focus on three hyperparameters: (i) the number of clusters (prototypes of unknown classes R) in Prototype-aware Semi-supervised Learning, (ii) the number of presumed OOD samples, and (iii) the temperature in Eq. 13. Note that in the previous experiments, we use the ground truth number of OOD samples and the goal of the experiment (ii) is to show that the model is not sensitive to the presumed number of OOD samples. For mathematical convenience, we introduce a coefficient α called OOD sample multiplier, and instead of choosing top K samples, we select top αK samples. The experimental results are shown in Fig. 3.

As can be seen from the results, although the performance fluctuates around $R \in [4, 5]$ in some cases, fixing the number of clusters R to 3 achieves the best performance. In particular, too few clusters ($R < 3$) would weaken the benefit of prototype learning on OOD samples while too many clusters ($R > 3$) would make the model overfitting after saturation. Moreover, while the actual number of unknown classes varies (*i.e.* 20 unknown classes for the COIL-DEL dataset and 5 unknown classes for the Letter-High dataset), assuming a moderate number of unknown prototypes (clusters) is generally beneficial for the model. One explanation is that the model uses K-means algorithm to initialize the unknown prototypes, which often yield balanced results with three clusters. Another possible reason for this phenomenon relates to the nature of OOD discovery. The model has little knowledge of out-of-distribution

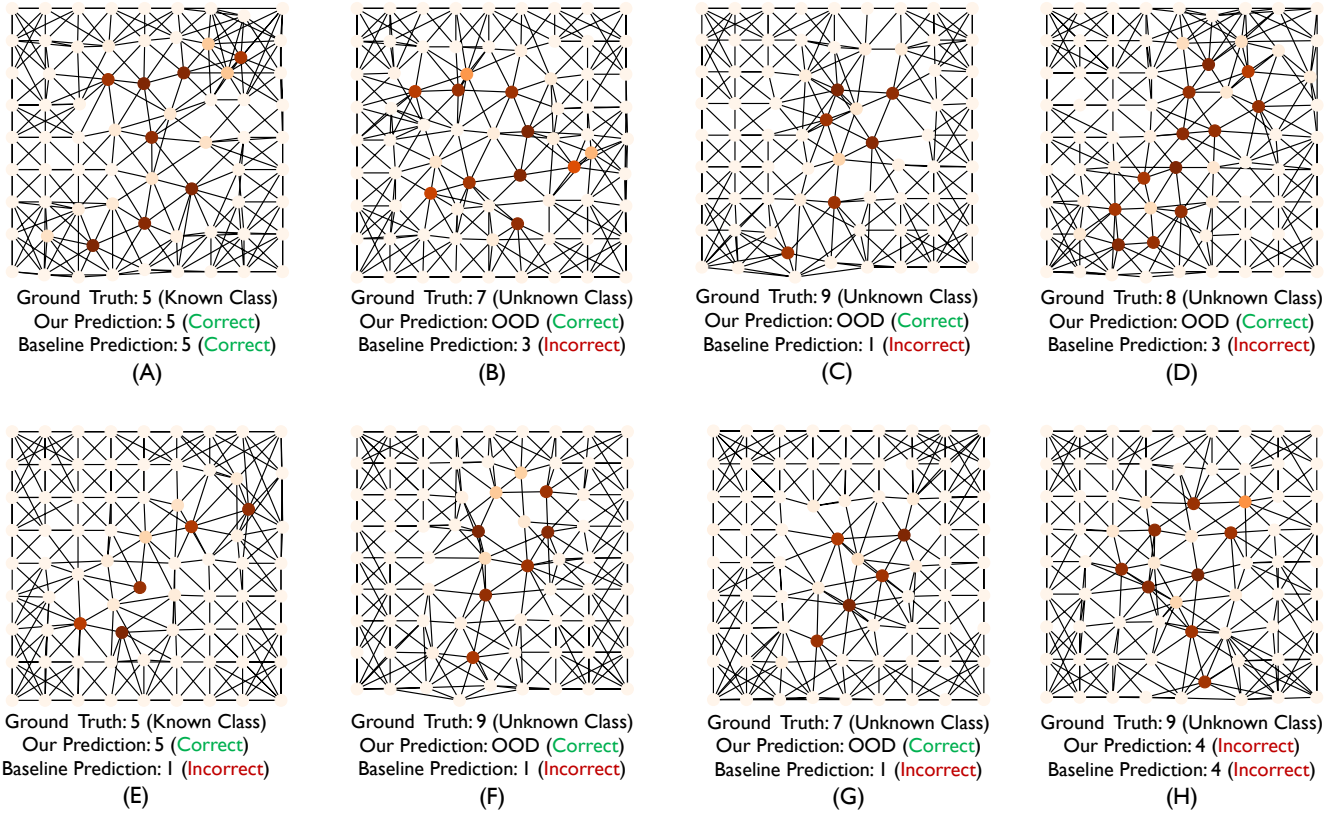


Fig. 4: Visualization of classification results. We visualize several graphs in the MNIST dataset with their corresponding ground truth classes, our prediction results and the prediction results of the baseline model using GIN and TopK pooling. The results show that in many cases, the proposed UGNN correctly detects OOD samples.

samples (since they are all unlabeled) and can only resort to the internal structures and attributes of the graphs to calculate the feature of each sample. This leads to coarse representations of OOD samples, and clustering them into too many prototypes becomes very challenging.

As for the number of OOD samples, the experiments demonstrate that an accurate estimation is not required to achieve competitive results. As can be seen from the middle column of Fig. 3, perturbing the presumed number of OOD samples in the range of 80% to 140% has little influence on the results. An interesting finding from the experiments is that the overestimation of the number has less influence compared to underestimation. One possible reason for this is that in the semi-supervised setting, there could be some samples belonging to the known classes but very different from labeled data. For example, there could be several sub-classes for a known class, but one sub-class does not appear in the labeled data. To some extent, this sub-class plays the role of an unknown class, and it is beneficial for the model to identify this fact.

For the temperature parameter, we observe that the model achieves the best accuracy when the temperature is set to 0.1, providing the correct "softness" for the softmax function in Eq. 13. An interesting phenomenon is that in the low labeling ratio case, the model is more sensitive to changes in the temperature parameter. The possible reason is that with less labels as supervision, the model relies relatively more on semi-supervised learning, and the influences of hyperparameters in Prototype-aware Semi-supervised Learning get amplified.

5.5 Visualization (RQ 4)

5.5.1 *Visualization of Classification Results*

In this subsection, we visualize eight graphs and our predictions in comparison with the prediction of the baseline model. Specifically, the experiments are conducted on the MNIST dataset, and we use GIN convolution [58] with TopK pooling [59] as the baseline. The result is shown in Fig. 4, and from the results, we have several observations:

- The task is generally more challenging than handwritten digits recognition in images, and GNN can perform relatively well. For example, in case (A), although the structure of the digit '5' is not very clear, both baseline model and the proposed UGNN classify this graph correctly.
- The baseline model is weak in detecting OOD samples, while the proposed UGNN is better at finding OOD samples. For example, in case (B), (C), (D), (F) and (G), the graphs belong to the unknown classes and the models are not provided with corresponding labels during training. In these more challenging cases, the proposed UGNN detects OOD samples correctly, whereas the baseline model yields seemingly reasonable but incorrect predictions.

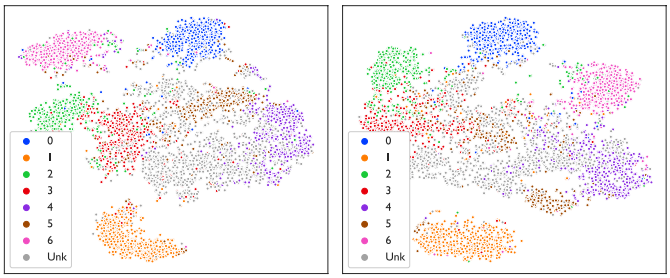


Fig. 5: Visualization of learned features using t-SNE. The experiments are conducted on the MNIST dataset with high labeling ratio, comparing the learned representations of our model (left) and a baseline model using GIN convolution and TopK pooling (right). Our model achieves the overall accuracy of 73.04%, while the baseline model reaches 62.55% overall accuracy. The results show that UGNN yields better representations.

- The proposed model is also better at classifying known classes. For example, in case (E), the graph belongs to the known class and while the baseline model fails to classify it correctly, our UGNN yields the right answer. This suggests that the proposed Subgraph-based OOD Detection and Prototype-aware Semi-supervised Learning not only help with finding out-of-distribution samples but also improve the representations of in-distribution samples.
- There are some very hard cases where both the proposed UGNN and the baseline model fail. For example, in case (H), the ground truth is '9', but the models do not see graphs with label '9' during training. To make things worse, it resembles the digit '4', which is often seen during training with labels. To some extent, it is reasonable for the model to make such mistakes.

5.5.2 Visualization of Learned Representations

We use t-SNE [68] to visualize the results of learned representations, which is shown in Fig. 5. More specifically, the experiments are conducted in the high labeling ratio case of the MNIST dataset, and we compare the results of UGNN in comparison with a GNN baseline that uses GIN convolution [58] and TopK pooling [59]. As can be seen from the results, our learned representations are more condensed. For example, for the class of digit 3 (red dots), our model yields more condensed features that are less confused with class Unk (unknown classes, gray dots). Another example is the class of digit 5 (brown dots), which is cut into two clusters (one cluster in the middle of the graph close to the red dots, the other lower in the graph between the purple dots and the orange dots) in the results of the baseline model. In comparison, our learned representations are better in that most brown dots are clustered into one group.

For the unknown classes, we find it challenging to distinguish their features with other learned representations of known classes clearly. However, our results are better than the baseline's. As can be seen in Fig. 5, the proposed UGNN not only provides a more condensed representation distribution of OOD samples, but also sets clearer boundaries. For example, our model better separates the OOD samples with the class of digit 2 (green dots).

We attribute the more condensed representations and clearer boundaries among classes to the Subgraph-based OOD detection and the following Prototype-aware Graph Semi-supervised Learning. Compared to the baseline model that uses only cross entropy loss, our method can better capture the internal structure of the graphs that tend to distinguish themselves from other samples.

6 Conclusion

This research studies the topic of semi-supervised universal graph classification, which attempts not only to detect graph samples that do not correspond to known classes but also to classify the remaining samples into their respective classes. From a subgraph prospective, we offer a novel approach dubbed UGNN that overcomes both class shifts and label scarcity in this problem. On the one hand, to achieve reliable OOD sample detection, UGNN samples several subgraphs for each sample and then measures both prediction confidence and individual output uncertainty comprehensively. On the other hand, UGNN builds graph prototype representations and then use the posterior prototype assignments inferred from one subgraph view to monitor the semantics of unlabeled input from another view. Extensive experiments on four benchmark graph classification datasets demonstrates the efficacy of our UGNN. In future work, we will apply our UGNN to more realistic graph classification scenarios, including domain adaptation and domain generalization.

Acknowledgments

The authors are grateful to the anonymous reviewers for critically reading this article and for giving important suggestions to improve this article.

References

[1] K. Hansen, F. Biegler, R. Ramakrishnan, W. Pronobis, O. A. Von Lilienfeld, K.-R. Müller, and A. Tkatchenko, "Machine learning predictions of molecular properties: Accurate many-body potentials and nonlocality in chemical space," *The journal of physical chemistry letters*, vol. 6, no. 12, pp. 2326–2331, 2015.
[2] R. Ying, R. He, K. Chen, P. Eksombatchai, W. L. Hamilton, and J. Leskovec, "Graph convolutional neural networks for web-scale recommender systems," in *KDD*, 2018.
[3] J. Lee, I. Lee, and J. Kang, "Self-attention graph pooling," in *ICML*, 2019.
[4] Z. Ying, J. You, C. Morris, X. Ren, W. Hamilton, and J. Leskovec, "Hierarchical graph representation learning with differentiable pooling," in *NeurIPS*, 2018.
[5] C. Lu, Q. Liu, C. Wang, Z. Huang, P. Lin, and L. He, "Molecular property prediction: A multilevel quantum interactions modeling perspective," in *AAAI*, 2019.
[6] K. Schütt, P.-J. Kindermans, H. E. S. Felix, S. Chmiela, A. Tkatchenko, and K.-R. Müller, "Schnet: A continuous-filter convolutional neural network for modeling quantum interactions," in *NeurIPS*, 2017.
[7] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, "Neural message passing for quantum chemistry," in *ICML*, 2017.
[8] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *ICLR*, 2017.
[9] G. Zhong and C.-M. Pun, "Latent low-rank graph learning for multimodal clustering," in *ICDE*, 2021.

[10] D. Shimin, Y. Quanming, Y. Zhang, and C. Lei, "Efficient relation-aware scoring function search for knowledge graph embedding," in *ICDE*, 2021.

[11] Z. Wang, T. Xia, R. Jiang, X. Liu, K.-S. Kim, X. Song, and R. Shibasaki, "Forecasting ambulance demand with profiled human mobility via heterogeneous multi-graph neural networks," in *ICDE*, 2021.

[12] J. Li, Y. Rong, H. Cheng, H. Meng, W. Huang, and J. Huang, "Semi-supervised graph classification: A hierarchical graph perspective," in *WWW*, 2019.

[13] F.-Y. Sun, J. Hoffmann, V. Verma, and J. Tang, "Infograph: Unsupervised and semi-supervised graph-level representation learning via mutual information maximization," in *ICLR*, 2020.

[14] Z. Hao, C. Lu, Z. Huang, H. Wang, Z. Hu, Q. Liu, E. Chen, and C. Lee, "Asgn: An active semi-supervised graph neural network for molecular property prediction," in *KDD*, 2020.

[15] M. Hein, M. Andriushchenko, and J. Bitterwolf, "Why relu networks yield high-confidence predictions far away from the training data and how to mitigate the problem," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 41–50.

[16] D. Hendrycks, M. Mazeika, and T. Dietterich, "Deep anomaly detection with outlier exposure," *arXiv preprint arXiv:1812.04606*, 2018.

[17] K. Lee, H. Lee, K. Lee, and J. Shin, "Training confidence-calibrated classifiers for detecting out-of-distribution samples," in *Proceedings of the International Conference on Learning Representations*, 2018.

[18] M. Caron, I. Misra, J. Mairal, P. Goyal, P. Bojanowski, and A. Joulin, "Unsupervised learning of visual features by contrasting cluster assignments," in *NeurIPS*, 2020.

[19] Y. M. Asano, C. Rupprecht, and A. Vedaldi, "Self-labelling via simultaneous clustering and representation learning," in *Proceedings of the International Conference on Learning Representations*, 2020.

[20] K. Zheng, W. Liu, L. He, T. Mei, J. Luo, and Z.-J. Zha, "Group-aware label transfer for domain adaptive person re-identification," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 5310–5319.

[21] Z. Zhang, P. Cui, and W. Zhu, "Deep learning on graphs: A survey," *IEEE Transactions on Knowledge and Data Engineering*, 2020.

[22] G. Guo, C. Wang, B. Yan, Y. Lou, H. Feng, J. Zhu, J. Chen, F. He, and P. Yu, "Learning adaptive node embeddings across graphs," *IEEE Transactions on Knowledge and Data Engineering*, 2022.

[23] T. Zhao, X. Zhang, and S. Wang, "Graphsmote: Imbalanced node classification on graphs with graph neural networks," in *Proceedings of the International ACM Conference on Web Search & Data Mining*, 2021, pp. 833–841.

[24] Z. Liu, Y. Fang, C. Liu, and S. C. Hoi, "Relative and absolute location embedding for few-shot node classification on graph," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2021, pp. 4267–4275.

[25] M. Zhang and Y. Chen, "Link prediction based on graph neural networks," in *Proceedings of the Conference on Neural Information Processing Systems*, 2018.

[26] L. Cai, J. Li, J. Wang, and S. Ji, "Line graph neural networks for link prediction," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[27] Z. Zhu, Z. Zhang, L.-P. Xhonneux, and J. Tang, "Neural bellman-ford networks: A general graph neural network framework for link prediction," in *Proceedings of the Conference on Neural Information Processing Systems*, 2021.

[28] X. Ma, J. Wu, S. Xue, J. Yang, C. Zhou, Q. Z. Sheng, H. Xiong, and L. Akoglu, "A comprehensive survey on graph anomaly detection with deep learning," *IEEE Transactions on Knowledge and Data Engineering*, 2021.

[29] C.-L. Li, K. Sohn, J. Yoon, and T. Pfister, "Cutpaste: Self-supervised learning for anomaly detection and localization," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 9664–9674.

[30] A. Deng and B. Hooi, "Graph neural network-based anomaly detection in multivariate time series," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2021, pp. 4027–4035.

[31] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *NeurIPS*, 2016, pp. 3844–3852.

[32] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, "Spectral networks and locally connected networks on graphs," *arXiv preprint arXiv:1312.6203*, 2013.

[33] M. Henaff, J. Bruna, and Y. LeCun, "Deep convolutional networks on graph-structured data," *arXiv preprint arXiv:1506.05163*, 2015.

[34] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," in *ICLR*, 2017.

[35] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How powerful are graph neural networks?" in *ICLR*, 2019.

[36] J. Lee, I. Lee, and J. Kang, "Self-attention graph pooling," in *International conference on machine learning*. PMLR, 2019, pp. 3734–3743.

[37] B. Zhang, Y. Wang, W. Hou, H. Wu, J. Wang, M. Okumura, and T. Shinozaki, "Flexmatch: Boosting semi-supervised learning with curriculum pseudo labeling," in *Proceedings of the Conference on Neural Information Processing Systems*, 2021, pp. 18 408–18 419.

[38] H. Li, N. Wang, X. Yang, X. Wang, and X. Gao, "Towards semi-supervised deep facial expression recognition with an adaptive confidence margin," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 4166–4175.

[39] K. Sohn, D. Berthelot, N. Carlini, Z. Zhang, H. Zhang, C. A. Raffel, E. D. Cubuk, A. Kurakin, and C.-L. Li, "Fixmatch: Simplifying semi-supervised learning with consistency and confidence," in *NeurIPS*, 2020.

[40] Y. You, T. Chen, Y. Sui, T. Chen, Z. Wang, and Y. Shen, "Graph contrastive learning with augmentations," *NeurIPS*, 2020.

[41] W. Ju, J. Yang, M. Qu, W. Song, J. Shen, and M. Zhang, "Kgnn: Harnessing kernel-based networks for semi-supervised graph classification," in *WSDM*, 2022.

[42] Y. Song and D. Wang, "Learning on graphs with out-of-distribution nodes," in *Proceedings of the International ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 2022, pp. 1635–1645.

[43] S. Liang, Y. Li, and R. Srikant, "Enhancing the reliability of out-of-distribution image detection in neural networks," in *Proceedings of the International Conference on Learning Representations*, 2018.

[44] V. Sehwag, M. Chiang, and P. Mittal, "Ssd: A unified framework for self-supervised outlier detection," *arXiv preprint arXiv:2103.12051*, 2021.

[45] A. Vyas, N. Jammalamadaka, X. Zhu, D. Das, B. Kaul, and T. L. Willke, "Out-of-distribution detection using an ensemble of self supervised leave-out classifiers," in *Proceedings of the European Conference on Computer Vision.*, 2018, pp. 550–564.

[46] K. Lee, K. Lee, H. Lee, and J. Shin, "A simple unified framework for detecting out-of-distribution samples and adversarial attacks," in *Proceedings of the Conference on Neural Information Processing Systems*, 2018.

[47] E. Ranjan, S. Sanyal, and P. Talukdar, "Asap: Adaptive structure aware pooling for learning hierarchical graph representations," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020, pp. 5470–5477.

[48] Y. You, T. Chen, Y. Shen, and Z. Wang, "Graph contrastive learning automated," in *ICML*, 2021.

[49] M. N. Rizve, K. Duarte, Y. S. Rawat, and M. Shah, "In defense of pseudo-labeling: An uncertainty-aware pseudo-label selection framework for semi-supervised learning," in *Proceedings of the International Conference on Learning Representations*, 2021.

[50] K. Riesen and H. Bunke, "Iam graph database repository for graph based pattern recognition and machine learning," in *Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR)*. Springer, 2008, pp. 287–297.

[51] V. P. Dwivedi, C. K. Joshi, T. Laurent, Y. Bengio, and X. Bresson, "Benchmarking graph neural networks," *arXiv preprint arXiv:2003.00982*, 2020.

[52] B. Knyazev, G. W. Taylor, and M. Amer, "Understanding attention and generalization in graph neural networks," in *Proceedings of the Conference on Neural Information Processing Systems*, 2019.

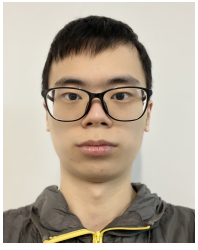
[53] N. Shervashidze, P. Schweitzer, E. J. Van Leeuwen, K. Mehlhorn, and K. M. Borgwardt, "Weisfeiler-lehman graph kernels." *Journal of Machine Learning Research*, vol. 12, no. 9, 2011.

[54] K. M. Borgwardt and H.-P. Kriegel, "Shortest-path kernels on graphs," in *Fifth IEEE international conference on data mining (ICDM'05)*. IEEE, 2005, pp. 8–pp.

[55] N. Shervashidze, S. Vishwanathan, T. Petri, K. Mehlhorn, and K. Borgwardt, "Efficient graphlet kernels for large graph comparison," in *Artificial intelligence and statistics*. PMLR, 2009, pp. 488–495.

[56] M. Welling and T. N. Kipf, "Semi-supervised classification with graph convolutional networks," in *J. International Conference on Learning Representations (ICLR 2017)*, 2016.
[57] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," *Advances in neural information processing systems*, vol. 30, 2017.
[58] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How powerful are graph neural networks?" in *International Conference on Learning Representations*, 2018.
[59] H. Gao and S. Ji, "Graph u-nets," in *international conference on machine learning*. PMLR, 2019, pp. 2083–2092.
[60] E. Ranjan, S. Sanyal, and P. Talukdar, "Asap: Adaptive structure aware pooling for learning hierarchical graph representations," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 04, 2020, pp. 5470–5477.
[61] F.-Y. Sun, J. Hoffmann, V. Verma, and J. Tang, "Infograph: Unsupervised and semi-supervised graph-level representation learning via mutual information maximization," *arXiv preprint arXiv:1908.01000*, 2019.
[62] Y. You, T. Chen, Y. Sui, T. Chen, Z. Wang, and Y. Shen, "Graph contrastive learning with augmentations," *Advances in Neural Information Processing Systems*, vol. 33, pp. 5812–5823, 2020.
[63] H. Yue, C. Zhang, C. Zhang, and H. Liu, "Label-invariant augmentation for semi-supervised graph classification," *arXiv preprint arXiv:2205.09802*, 2022.
[64] S. Li, X. Wang, A. Zhang, Y. Wu, X. He, and T.-S. Chua, "Let invariant rationale discovery inspire graph contrastive learning," in *International Conference on Machine Learning*. PMLR, 2022, pp. 13 052–13 065.
[65] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," in *CVPR*, 2020.
[66] P. Khosla, P. Teterwak, C. Wang, A. Sarna, Y. Tian, P. Isola, A. Maschinot, C. Liu, and D. Krishnan, "Supervised contrastive learning," *Advances in Neural Information Processing Systems*, vol. 33, pp. 18 661–18 673, 2020.
[67] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.
[68] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne." *Journal of machine learning research*, vol. 9, no. 11, 2008.

Xiao Luo is a postdoctoral researcher in Department of Computer Science, University of California, Los Angeles, USA. Prior to that, he received the Ph.D. degree in School of Mathematical Sciences from Peking University, Beijing, China and the B.S. degree in Mathematics from Nanjing University, Nanjing, China, in 2017. His research interests includes machine learning on graphs, image retrieval, statistical models and bioinformatics.



Yusheng Zhao is a graduate student in School of Computer Science, Peking University, Beijing, China. His research interest includes machine learning with graphs, vision-and-language and adversarial learning.

Yifang Qin is an undergraduate student in School of EECS, Peking University, Beijing, China. His research interests include graph representation learning and recommender sysmtes.

Wei Ju is currently a postdoc research fellow in Computer Science at Peking University. Prior to that, he received his Ph.D. degree in Computer Science from Peking University, Beijing, China, in 2022. He received the B.S. degree in Mathematics from Sichuan University, Sichuan, China, in 2017. His current research interests lie primarily in the area of machine learning on graphs including graph representation learning and graph neural networks, and interdisciplinary applications such as drug discovery and recommender systems. He has published more than 20 papers in top-tier venues and has won the best paper finalist in IEEE ICDM 2022.

Ming Zhang received her B.S., M.S. and Ph.D. degrees in Computer Science from Peking University respectively. She is a full professor at the School of Computer Science, Peking University. Prof. Zhang is a member of Advisory Committee of Ministry of Education in China and the Chair of ACM SIGCSE China. She is one of the fifteen members of ACM/IEEE CC2020 Steering Committee. She has published more than 200 research papers on Text Mining and Machine Learning in the top journals and conferences. She won the best paper of ICML 2014 and best paper nominee of WWW 2016. Prof. Zhang is the 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, National First-Class Undergraduate Course by MOE China.