# Active and Semi-supervised Graph Neural Networks for Graph Classification

Yu Xie, *Member, IEEE*, Shengze Lv, Yuhua Qian, *Member, IEEE*, Chao Wen, *Member, IEEE*, and Jiye Liang, *Senior Member, IEEE*

**Abstract**—Graph classification aims to predict the class labels of graphs and has a wide range of applications in many real-world domains. However, most of existing graph neural networks for graph classification tasks use 90% of labeled graphs for training and the remaining 10% for testing, which obviously struggle in solving the problem of the scarcity of labeled graphs in real-world graph classification scenarios. And it is arduous to label a large number of graph examples for training because of the difficulty and resource consumption in the tagging process. Motivated by this, we propose a novel active and semi-supervised graph neural network (ASGNN) framework, which endeavors to complete graph classification tasks with a small number of labeled graph examples and available unlabeled graph examples. In our framework, active learning selects high-uncertain and representative graph examples from the test set and add them to the training set after annotation. Semi-supervised learning is utilized to select the high-confidence unlabeled graph examples containing structural information from the test set, and add them to the training set after pseudo labeling. To improve the generalization performance of the graph classification model, multiple GNNs are trained collaboratively for promoting the expressiveness of each other and increasing the reliability of graph classification results. Overall, the ASGNN framework takes fully use of unlabeled graph examples to reinforce graph classification effectively, and can be applied to any existing supervised graph neural networks for graph classification. Experimental results on benchmark graph datasets demonstrate that the proposed framework yields competitive performance on graph classification tasks with only a small number of labeled graph examples.

**Index Terms**—Graph neural networks, active learning, semi-supervised learning, graph classification

## 1 Introduction

As a powerful data organization way, graph structured data are ubiquitous across many domains such as knowledge graphs, social networks, biological networks and recommender systems [1]–[3]. Graph classification [4] endeavors to identify the class labels of graphs and has become an important research hot-spot in numerous scenarios like text categorization, protein function prediction, chemical compound classification and malicious code detection, etc.

To tackle the task of graph classification, various prevalent methods are proposed. Graph kernel methods [5]–[9] leverage on structural properties such as walks, subtrees, shortest path lengths or graphlets for measuring the similarity among graphs and classify graphs into different categories by the supervised classifiers. The advantage of graph kernel methods is that they can be compatible with any standard plug and play classifier (SVM, random forest, multilayer perceptron, etc.) easily. However, they mainly follow a two-stage learning framework in which graph feature learning and classification are processed respectively. More recently, graph neural network (GNN) as the latest research achievement in the field of deep learning has become a representative tool for machine learning on graphs, especially for graph classification. Graph neural network based methods can extract expressive and discriminative graph structural features in a supervised end-to-end manner [10]–[15]. In specific, graph neural networks accentuate neighborhood aggregation and feature passing among nodes, both of which are responsible for recursively learning intricate structural information in graphs. Although graph neural networks have achieved the state-of-the-art graph classification results, they have one limitation that training high-quality networks is data-hungry and often depends on the abundant labeled graphs, since most existing graph neural network approaches for graph classification tasks use 90% of labeled graphs for training, and the remaining 10% for testing. Unfortunately, the high dimensionality and complexity of graph structured data [16], [17] bring great challenges to label annotation, and it is unrealistic to obtain a large amount of labeled graph examples for training graph neural networks in the real-world graph classification applications.

To cope with the graph classification situation with limited availability of class labels or no available graph labels during training, some works [18]–[20] introduced self-supervised learning to graph neural networks for graph classification. Nguyen et al. [18] extended the universal self-attention network from NLP tasks to graph classification and proposed U2GNN for learning graph embeddings that can memorize the dependencies among nodes and characterize node attributes and global network properties. Specifically, U2GNN first generates node embeddings by optimizing the unsupervised contrastive loss and then obtains the embedding representation of the entire graph by

- *Yu Xie, Yuhua Qian, Chao Wen and Jiye Liang are with the Key Laboratory of Computational Intelligence and Chinese Information Processing of Ministry of Education, Shanxi University, Taiyuan 030006, China. Yuhua Qian and Chao Wen are also with Institute of Big Data Science and Industry, the Engineering Research Center of Machine Vision and Data Mining of Shanxi Province, Shanxi University.*
- *Shengze Lv is with School of Computer Science and Technology, Xidian University, Xi'an 710071, China.*

*(Corresponding author: Yuhua Qian. E-mail: jinchengqyh@126.com.)*

summing all learned node embeddings. GCC [19] presents a self-supervised pre-training GNN framework, which treats subgraph instance discrimination in and across networks as a pre-training task and leverages contrastive learning to empower graph neural networks for learning the transferable structural representations. Actually, both U2GNN [18] and GCC [19] detach the model training and subsequent tasks, and are inclined to learn universal representations for graph-relevant tasks. To produce more discriminative graph representations tailored for graph classification, L-CGNN [20] optimizes the traditional cross-entropy graph classification loss coupled with the label contrastive coding loss, which can utilize the available label information of graph examples to encourage the instance-level intra-class compactness and inter-class separability. To the best of our knowledge, there is no work that dedicates to exploring the strength of active learning and semi-supervised learning for taking full use of small number of labeled examples and available unlabeled examples to achieve desirable graph classification to date, which is also very important in both industry and academic applications in reality.

Motivated by this, we propose a novel framework named active and semi-supervised graph neural network (ASGNN) to learn graph classification models with better generalization through collaboratively training of multiple graph neural networks. Specifically, we first train multiple graph neural network models using the training set which is continuously expanded by active learning. Active learning exploits multiple GNNs to select valuable unlabeled graph examples with high uncertainty and representativeness from the test set, and add them to the training set after annotation so as to improve the graph classification performance. Then, semi-supervised learning uses multiple graph neural networks to select many high-confidence unlabeled graph examples from the test set, and add them to the training set after pseudo labeling to further promote the model performance. To avoid performance degradation of the model due to the accumulation of mislabeled graph examples in the training set, we take out the pseudo labeled graph examples and relabel unlabeled graph examples at specific intervals. In a nutshell, the key contributions of this paper can be summarized as follows:

- A universal active and semi-supervised graph neural network (ASGNN) framework is proposed to tackle the challenge of the scarcity of labeled graph examples in real-world graph classification tasks.
- Active learning and semi-supervised learning modules designed in our framework exploit multiple graph neural networks to collaboratively select the valuable graph examples from the test set and add them to the training set after true or pseudo annotation, which can reliably enhance the generalization performance of graph classification models.
- Empirical evaluation on several real-world datasets demonstrate that our framework yields superior performance on graph classification tasks with a small amount of labeled graph examples and available unlabeled graph examples.

The remainder of this paper is organized as follows. Related works are reviewed in the next section. We then elaborate the proposed framework in Section 3. Experimental setup and discussion of results are provided in Section 4. Finally, we conclude the paper and give future directions.

## 2 Related Work

In this section, we review related works strongly related to our work, including graph neural networks, active learning and semi-supervised learning.

### 2.1 Graph Neural Networks

Graph neural network based methods for graph classification represent each graph as a low-dimensional vector, which is usually constructed based on the learned node representations that preserve the global structure of a whole graph. By virtue of graph neural networks, two similar graphs can be mapped into the embedding space closely. Recently, a great many graph neural networks are developed to tackle the problem of graph classification. MPNN [10] contemplates that related graph neural network models can be boiled down to a universal neural message passing framework, which is composed of a message passing phase for neighboring node feature aggregation and a readout phase for generating graph representations. A back-trackless aligned-spatial graph convolutional network [12] is proposed to learn effective features for graph classification, which is inspired by the idea of the arbitrary-sized graphs can be transformed into fixed-sized back-trackless aligned grid structures. Chen et al. [14] devised a simple lightweight graph feature network, which uses a simplified GNN with linear graph filtering and non-linear set function to conduct graph classification tasks with a fraction of computation cost. However, these variants usually perform the graph classification task with $90\%$ labeled graph examples for training and the remaining $10\%$ graph examples for validation. In many real-world application scenarios, labeling graph data is very difficult and laborious owing to the high structural complexity, which often leads to the lack of sufficient labeled graph data for graph classification tasks. Therefore, how to use a small amount of labeled graph data and available unlabeled graph data to complete the task of graph classification is a challenging problem.

### 2.2 Active Learning

Active learning focuses on promoting the classification performance of the model with a small amount of labeled examples and less label cost as much as possible. It first selects the data with high representativity from the test set by certain query strategies [21]. After being manually labeled by experts, the selected data coupled with their category labels are incorporated into the training set to iteratively promote the model performance. The key point of active learning is the query strategy which usually contains multiple indicators for obtaining examples with high representativity from test set. Recently, active learning mainly contains two categories of query methods, query synthesizing and query acquiring/pool based. The query synthesizing methods directly generate example data for model training by using generative models such as generative adversarial networks. In recent years, most of active

learning methods are query acquiring based, i.e., designing query strategies to select the valuable example data. Joshi et al. [22] developed a multi-class active learning setup and a value-of-information algorithm for multi-class image classification. To solve the text classification task, Goudjil et al. [23] presented an active learning method based on support vector machine to reduce the labeling effort, without compromising the classification accuracy, by intelligently selecting the examples to be labeled. Cai et al. [24] proposed an active graph embedding framework in order to optimize the node classification performance by actively selecting the labeled training nodes. However, to the best of our knowledge, there is no method focusing on applying active learning to GNNs for graph classification tasks. Inspired by this, we attempt to incorporate active learning for tackling graph classification tasks.

### 2.3 Semi-supervised Learning

Semi-supervised learning acts as a classical learning paradigm between supervised learning and unsupervised learning, and has attracted increasing attention in the field of pattern recognition and machine learning [25], [26]. Semi-supervised learning makes an effort to effectively enhance the performance of machine learning models by utilizing a small number of labeled examples and available unlabeled examples. In fact, both unlabeled examples and labeled examples are obtained from the total examples of independently and identically distributed, and unlabeled examples containing the data distribution information enable improving the generalization ability of machine learning models. Therefore, a few works have begun to investigate the use of semi-supervised learning on graph neural networks for node classification [27]. Wang et al. [28] proposed an adaptive multi-channel graph convolutional network (AM-GCN) for semi-supervised node classification, which adequately preserves the topological structures, node features and their correlated information to improve the capability of GCNs. Liao et al. [29] introduced graph partition neural networks (GPNNs) to handle extremely large graphs for semi-supervised node classification, which alternate between locally propagating information among nodes in small subgraphs and globally propagating information among the subgraphs. All these methods demonstrate that semi-supervised learning can help boost the performance of node classification tasks remarkably. As a matter of fact, promoting the performance of various classification tasks by using semi-supervised learning has made great strides. Jakob et al. [30] designed a semi-supervised approach using the tags associated with labeled and unlabeled images to learn a classifier for image classification. Zhu et al. [31] presented a multi-view semi-supervised learning framework, which leverages the information contained in pseudo labeled images to improve the prediction performance of image classification using multiple views of an image [32]. These research efforts indicate that semi-supervised learning can improve the performance of classification tasks in many cases. However, they paid scant attention on the truth that a small amount of training graph data may not achieve valuable results, and there is no work applying semi-supervised learning to GNNs for graph classification tasks. To tackle this, we incorporate semi-supervised learning and active learning to graph neural networks for graph classification tasks for the first time as far as we know.

## 3 Methodology

This section first introduces the problem definition and notations. Then we describe the proposed active and semi-supervised graph neural network framework for graph classification. Next, we expound on how our framework incorporates active learning and semi-supervised learning to improve the performance of graph classification with less labeled graphs in detail.

### 3.1 Problem Statement and Notations

#### 3.1.1 Graph Classification

A graph is represented as $\mathcal{G}_m = (V, E)$, where $V$ is a set of nodes, $E$ is a set of edges. Given the input space of graphs $\{\mathcal{G}_m\}_{m=1}^{M}$ and a set of class labels $\mathcal{Y}$, the goal of graph classification is to learn a mapping function $f : \{\mathcal{G}_m\}_{m=1}^{M} \rightarrow \mathcal{Y}$.

#### 3.1.2 Supervised Graph Classification

Given a training set $G_{\text{training}} = \{\mathcal{G}_1, \ldots, \mathcal{G}_l\}$ that contains a certain number of labeled graph examples, supervised graph classification aims at learning a mapping function $f$ to predict the class labels for unlabeled graph examples in the test set $G_{\text{test}} = \{\mathcal{G}_{l+1}, \ldots, \mathcal{G}_{l+u}\}$.

#### 3.1.3 Active Graph Classification

Given a training set $G_{\text{training}} = \{\mathcal{G}_1, \ldots, \mathcal{G}_l\}$ and a test set $G_{\text{test}} = \{\mathcal{G}_{l+1}, \ldots, \mathcal{G}_{l+u}\}$, active graph classification attempts to select a set of graphs $G_{\text{select}} = \{\mathcal{G}_{l+1}, \ldots, \mathcal{G}_{l+k}\}$ from the test set and adds them to the training set after annotation, so as to utilize the new training set $G_{\text{training}} = \{\mathcal{G}_1, \ldots, \mathcal{G}_l, \mathcal{G}_{l+1}, \ldots, \mathcal{G}_{l+k}\}$ to predict the class labels for unlabeled graph examples in the new test set $G_{\text{test}} = \{\mathcal{G}_{l+k+1}, \ldots, \mathcal{G}_{l+u}\}$.

#### 3.1.4 Semi-supervised Graph Classification

Given a training set $G_{\text{training}} = \{\mathcal{G}_1, \ldots, \mathcal{G}_l, \mathcal{G}_{l+1}, \ldots, \mathcal{G}_{l+u}\}$ that contains $l$ labeled graph examples and $u$ unlabeled graph examples, the purpose of semi-supervised graph classification is to predict the class labels for unlabeled graph examples in the test set $G_{\text{test}} = \{\mathcal{G}_{l+1}, \ldots, \mathcal{G}_{l+u}\}$.

A description of the notations used in this paper is given in TABLE 1.

### 3.2 Framework

The typical graph neural network based methods usually require a large amount of labeled graph examples to accomplish the graph classification task. However, graph classification tasks are frequently faced with a problem of the scarcity of labeled graph examples since large-scale labeled graph datasets often mean the high cost of time and labor. Fortunately, active learning and semi-supervised learning are two classical learning paradigms which can be utilized in the training process to solve the problem of the scarcity of labeled graph examples by enlarging the training set. In specific, active learning can select the graph examples with

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TBDATA.2021.3140205, IEEE Transactions on Big Data



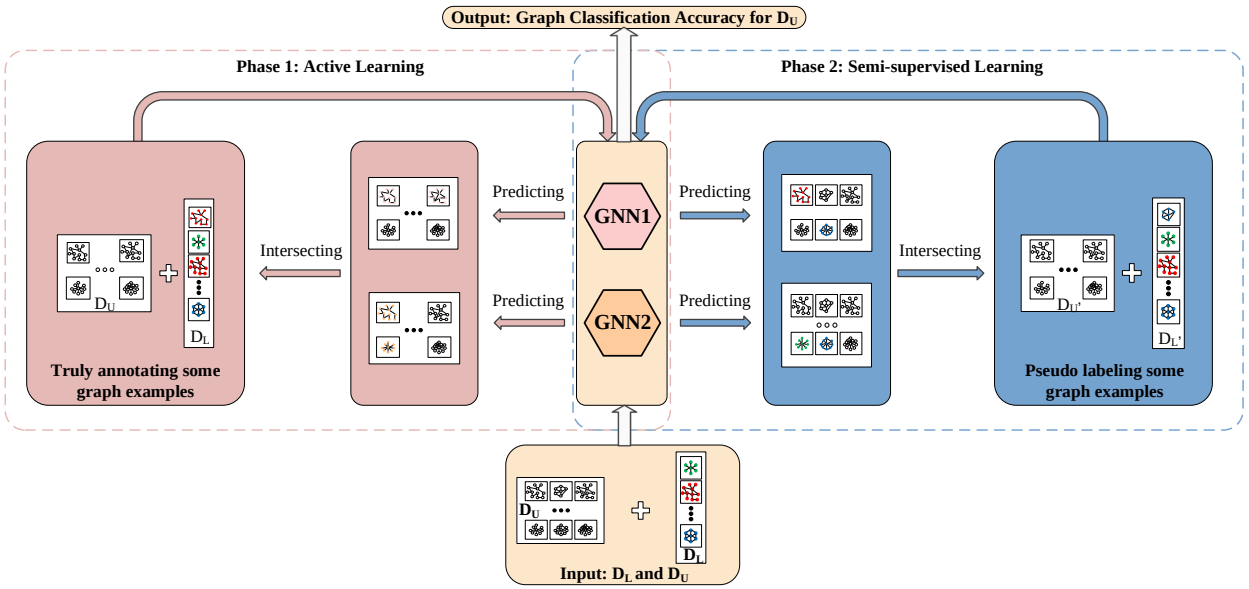



Fig. 1: Graphical illustration of our proposed active and semi-supervised graph neural network framework for graph classification. With labeled graph examples from $D_L$ and unlabeled graph examples from $D_U$, our framework explores active learning and then semi-supervised learning to train two graph neural networks (GNN1 and GNN2). In the phase of active learning, each GNN selects the graph examples with high uncertainty and representativity, followed by which we take out the graph examples that two GNNs simultaneously consider valuable from the test set, and add them to the training set after truly annotation. In the phase of semi-supervised learning, two GNNs select the graph examples with high confidence level by soft clustering respectively, and then add the intersection of them to the training set after pseudo labeling. Finally, we can confidently output the graph classification accuracy for $D_U$.

TABLE 1: The Notations Used in This Paper.

| Notations | Explanations |
|---|---|
| $\mathcal{G}$ | A graph example |
| $V$ | Set of nodes in a graph |
| $E$ | Set of edges in a graph |
| $y$ | The class label of a graph |
| $\mathcal{Y}$ | Set of class labels |
| $G_{\text{training}}$ | The training set for graph classification |
| $G_{\text{test}}$ | The test set for graph classification |
| $\Theta$ | The parameter matrix |
| $NN_\Theta$ | A neural network |
| $\mathcal{N}(v_i)$ | The neighbor set of node $v_i$ |
| $h_\mathcal{G}$ | The graph feature representation of $\mathcal{G}$ |
| $readout()$ | A readout function to obtain the graph-level representation |
| $D_L$ | The set of labeled graph examples |
| $D_U$ | The set of unlabeled graph examples |
| $p_E$ | The entropy percentage |
| $p_d$ | The euclidean percentage |
| $al\_k$ | Graph examples selected by active learning per epoch |
| $ss\_k$ | Graph examples selected by semi-supervised learning per epoch |
| $AL\_K$ | Proportion budget of graph examples selected by active learning |
| $SS\_K$ | Proportion budget of graph examples selected by semi-supervised learning |
| $invl$ | Interval of epochs for taking out pseudo labeled graph examples |

high value from the test set, and semi-supervised learning can select the graph examples with high confidence level from test set. The graph examples selected by active learning or semi-supervised learning can be added into training set after truly annotation or pseudo labeling respectively.

Motivated by this, we propose a novel active and semi-supervised graph neural network framework for graph classification, whose graphical illustration is shown in Fig. 1. The proposed framework includes two paradigms, i.e., active learning and semi-supervised learning, which are efficiently applied to multiple GNNs for promoting the graph classification performance. In this way, although the graph examples with high value selected from test set in active learning by only one GNN probably are not reliable, this framework selects the graph examples that are commonly recognized as valuable to the performance gain of graph classification models through multiple GNNs, and then adds them into the training set after human annotation. As for semi-supervised learning, our framework selects the unlabeled graph examples with high confidence level in soft clustering jointly by multiple GNNs, and adds them to the training set after pseudo labeling for improving the graph classification results.

Graph structured data is originally organized in non-Euclidean geometric space and should first be represented as feature vectors so as to be convenient for subsequent graph classification tasks. In order to generate the graph-level representation $h_{\mathcal{G}_m}$ for a graph $\mathcal{G}_m$, we first learn the low-dimensional embedding features for each node by a powerful aggregation function:

$$h_{v_i}^{(t)} = NN_\Theta{}^{(t)}\left(h_{v_i}^{(t-1)} + \sum_{v_j \in \mathcal{N}(v_i)} h_{v_j}^{(t-1)}\right) \tag{1}$$

where $h_{v_i}^t$ denotes the feature vector of node $v_i$ at $t$-th layer, $\mathcal{N}(v_i)$ represents the set of neighboring nodes of node $v_i$, $NN_\Theta$ is a neural network such as multi-layer perceptron and $\Theta$ is the parameter matrix to be learned. The feature vector of node $v_i$ at $t$-th layer can be obtained from its own feature vector at $(t-1)$-th layer incorporated with the aggregation of feature vectors of its neighboring nodes at $(t-1)$-th layer. Then, a readout function is used to yield the graph-level representation $h_{\mathcal{G}_m}$ generally as the following form:

$$h_{\mathcal{G}_m} = readout\left(\left\{h_{v_i}^{(T)} \mid v_i \in \mathcal{G}_m\right\}\right) \tag{2}$$

where $T$ denotes the number of layers in the graph neural network. The readout function can be instantiated by a global sum/mean pooling, followed by fully connected and softmax layers to generate the categorical output. The training loss of two graph neural networks can be defined by:

$$\mathcal{L}_{\text{tra}} = \sum_{\mathcal{G}_m}(-\sum_{l=1}^{L} y_l[\mathcal{G}_m]\log\left(p_l[\mathcal{G}_m]\right)) \tag{3}$$

where $L$ denotes the number of categories of graph examples, $y_l[\mathcal{G}_m]$ represents the indicator variable (if the category $l$ is the same as the category of a graph example $\mathcal{G}_m$, $y_l[\mathcal{G}_m]$ is 1, otherwise it is 0) and $p_l[\mathcal{G}_m]$ is the predicted probability that the graph example $\mathcal{G}_m$ belongs to the category $l$.

To enhance the graph classification performance, we make graph neural networks see more data. Active learning is utilized to select the graph examples that contain rich information from test set, and then these examples are added into the training set after annotation. Generally, the more the representativeness and uncertainty of the graph examples in a dataset, the richer the information contained in these graph examples. We employ two indicators including euclidean distance and information entropy to measure the representativeness and uncertainty of graph examples, respectively. Selecting the graph examples that contain rich information by one GNN may be unreliable. Therefore, our framework explores two GNNs to collaboratively select the graph examples, and add the intersection of the selected graph examples by the two GNNs to the training set after human annotation to promote the performance of graph neural networks.

To further improve the performance of graph classification, semi-supervised learning is adopted to select the unlabeled graph examples that are most likely belonging to a certain category. Then graph neural networks can pseudo label them with predicted labels, and add the pseudo labeled graph examples to the training set. There is no doubt that selecting the unlabeled graph examples with high confidence level from the test set is a crucial issue. Soft clustering, a fuzzy clustering method, is exploited to tackle this issue by calculating the soft clustering scores of the graph examples and regard the graph examples with high score of soft clustering results as the pseudo labeled graph examples. Likewise, our framework takes into account the intersection of unlabeled graph examples selected by two GNNs as the graph examples waiting for pseudo labeling. After pseudo labeling the selected graph examples, the pseudo labeled graph examples with high confidence are added to the graph training set expanded by the previous active learning, and these pseudo labeled graph examples are taken out after a specific epoch. As a result, the performance of graph neural networks can be effectively promoted by the pseudo labeled graph examples. After active learning and semi-supervised learning, the label of each unlabeled graph example is determined by the averaged predicted probability of two GNNs. For clarity, Algorithm 1 summarizes the overall procedure of our proposed framework.

**Algorithm 1** Procedure of the proposed ASGNN framework

**Input:**

$D_L$, $D_U$

**Output:**

The graph classification accuracy of $D_U$

**Step 1 Initialization:**

Initialize $epoch = 0$, $al\_k$, $AL\_K$, $ss\_k$, $SS\_K$, $invl$.

**Step 2 Optimization:**

$D_{whole} \leftarrow D_L \cup D_U$

SHUFFLE($D_{whole}$)

**Step 2.1 Active learning:**

**while** $epoch <= \frac{AL\_K * |D_{whole}|}{al\_k}$ **do**

$(D_{L'}, D_{U'}) \leftarrow Algorithm2(D_L, D_U, al\_k)$

$D_L = D_{L'}, D_U = D_{U'}$

Update gradient of GNN1 and GNN2 by the SGD algorithm.

$epoch++$

**end while**

**Step 2.2 Semi-supervised learning:**

**while** $epoch <= \frac{SS\_K * |D_{whole}|}{ss\_k}$ **do**

$(D_{L'}, D_U) \leftarrow Algorithm3(D_L, D_U, ss\_k)$

$D_L = D_{L'}$

Update gradient of GNN1 and GNN2 by the SGD algorithm.

$epoch++$

**if** $epoch \ \% \ invl = 0$ **then**

Take out the pseudo labeled graph examples from $D_L$.

**else**

continue

**end if**

**end while**

**Step 3 Output the classification results:**

Output the classification accuracy of graph examples in $D_U$.

### 3.3 Active Learning

In this subsection, we illustrate the active learning module designed in our framework in detail. As shown in Fig. 2, the unlabeled graph example set $D_U$ and the labeled graph example set $D_L$ are taken as the input of two GNNs. GNN1 and GNN2 collaboratively select the graph examples that contain rich information from test set, and add them to the training set after annotation. For the sake of measuring the richness of the information contained in graph examples, two indicators are employed including information entropy and euclidean distance in our framework.

By virtue of the end-to-end graph neural networks, the classification probability of each graph example can be output by the Softmax formula straightforwardly as

$$p_l[\mathcal{G}_m] = \text{Soft}\max\left(S_l[h_{\mathcal{G}_m}]\right) = \frac{\exp\left(S_l[h_{\mathcal{G}_m}]\right)}{\sum_{l'=1}^{L}\exp\left(S_{l'}[h_{\mathcal{G}_m}]\right)} \tag{4}$$

where $S_l\left[h_{\mathcal{G}_m}\right]$ denotes the soft clustering score that a graph example $\mathcal{G}_m$ is predicted to belong to class $l$ as defined in Eq. 9. $\text{Softmax}\left(S_l[h_{\mathcal{G}_m}]\right)$ is the predicted probability of a graph $\mathcal{G}_m$ from $D_U$ is classified into the category $l$, and $L$ indicates the number of classes of graph examples. Furthermore, the

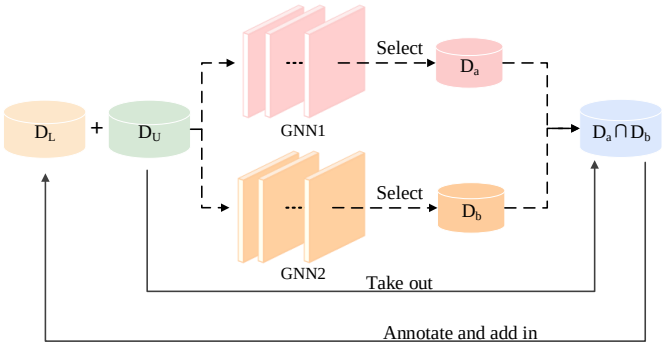



Fig. 2: Graphical illustration of the process of active learning. $D_U$ and $D_L$ denote set of unlabeled graph examples and set of labeled graph examples, the representative graph examples $D_a$ and $D_b$ are selected by GNN1 and GNN2 respectively. The intersection of $D_a$ and $D_b$ is utilized to promote the classification performance of graph neural networks by adding them to the training set after annotation.

predicted probability can be utilized to calculate the entropy of a graph example by the following formula so as to choose the highly uncertain graph examples from test set

$$E = -\sum_{l=1}^{L} p_l[\mathcal{G}_m]^* \log\left(p_l[\mathcal{G}_m]\right) \tag{5}$$

where $p_l[\mathcal{G}_m]$ represents the probability that a graph example $\mathcal{G}_m$ is predicted to belong to the class $l$ and $L$ denotes the number of classes of graph examples. To quantitatively describe the uncertainty of graph example $\mathcal{G}_m$ in the whole unlabeled graph dataset $D_U$, the proportion of all graph examples whose entropy value is less than $\mathcal{G}_m$ is defined as the entropy percentage $p_E$ of the graph example $\mathcal{G}_m$. The higher the entropy percentage of the graph example $\mathcal{G}_m$, the greater the uncertainty of the graph example $\mathcal{G}_m$ in the unlabeled graph dataset $D_U$.

Considering that only using one indicator entropy to evaluate the information contained in a graph example is unreliable, we further exploit euclidean distance to measure the richness of the information contained in the graph example. Essentially, the clustering center i.e., $C = \{C_1, C_2, \ldots, C_L\}$ of available labeled graph examples can be obtained by the following formula:

$$C_l = \frac{\sum\{h_l\left[\mathcal{G}_m\right] \mid \mathcal{G}_m \in D_L \quad and \quad \mathcal{Y}\left[\mathcal{G}_m\right] = l\}}{|\{\mathcal{G}_m \mid \mathcal{G}_m \in D_L \quad and \quad \mathcal{Y}\left[\mathcal{G}_m\right] = l\}|} \tag{6}$$

where $h_l\left[\mathcal{G}_m\right]$ represents the embedding representation of each labeled graph example in the category $l$. The representativeness of unlabeled graph examples can be determined by the Euclidean distance from each unlabeled graph example to the nearest clustering center, which is shown as follows:

$$d_{\mathcal{G}_m} = \min_{l=1,\ldots,L}\left\{\|h_{\mathcal{G}_m} - C_l\|_2^2\right\} \tag{7}$$

Later, we calculate the proportion of all graph examples whose Euclidean distance is less than $\mathcal{G}_m$, and define it as the euclidean percentage $p_d$ of the unlabeled graph example $\mathcal{G}_m$. The higher the euclidean percentage of graph example $\mathcal{G}_m$, the greater the representativeness of the graph example $\mathcal{G}_m$ in the unlabeled graph dataset $D_U$.

**Algorithm 2** Procedure of active learning in the ASGNN framework

**Input:**

$D_L$, $D_U$, $al\_k$

**Output:**

New training set $D_{L'}$ and test set $D_{U'}$ after active learning

**Step 1 Process:**

Calculate the entropy percentage and euclidean percentage for each graph by GNN1 and GNN2 simultaneously.
GNN1 selects $al\_k$ representative graph examples $D_a$.
GNN2 selects $al\_k$ representative graph examples $D_b$.
Take out $D_a \cap D_b$ from $D_U$ and add them into $D_L$ after annotation.

**Step 2 Output:**

Output the new training set $D_{L'}$ and test set $D_{U'}$ after active learning.

Finally, the representativeness of each graph example is determined by multiple indicators, which is advantageous to select graph examples with high uncertainty and strong representativeness as the graph examples with rich information to be labeled. After selection, we add them to the training set after annotation, so as to boost the classification performance of GNN models and further improve the accuracy of graph classification tasks. The multiple indicator weighting formula is defined as follows:

$$I_{\mathcal{G}_m} = \alpha^* p_E + (1-\alpha)^* p_d \tag{8}$$

where $p_E$ and $p_d$ represent the entropy percentage and euclidean percentage, $\alpha$ and $1-\alpha$ are the weights of two indicators, respectively. $I_{\mathcal{G}_m}$ indicates the richness of information contained in the graph example $\mathcal{G}_m$. According to the $I$ value of each unlabeled graph example that simultaneously considers the uncertainty and representativeness of the graph, the graph example with the most significant performance gain to the training model can be selected.

For the sake of enhancing the generalization ability of graph classification models, GNN1 selects the unlabeled graph examples $D_a$ with rich information based on two indicators including entropy percentage and euclidean percentage, and GNN2 also selects the unlabeled graph examples $D_b$ based on these two indicators. The intersection of $D_a$ and $D_b$ is taken out from $D_U$ as the final result of selection and is then added into $D_L$ after annotation to promote the performance of graph neural networks for improving the accuracy of graph classification tasks. Algorithm 2 summarizes the detailed procedure of the active learning module designed in our framework.

### 3.4 Semi-supervised Learning

The semi-supervised learning module designed in our proposed framework attempts to select the unlabeled graph examples that are most likely belonging to a certain category, to pseudo label them with predicted labels, and add these pseudo labeled graph examples to the training set for promoting the performance of GNNs. Therefore, selecting

the unlabeled graph examples with high confidence level from the test set is a primary problem. We tackle this problem by regarding the unlabeled graph examples with high soft clustering scores as graph examples waiting to be pseudo labeled.

$$S_l\left[\mathcal{G}_m\right] = \frac{\left(\left\|h_{\mathcal{G}_m} - C_l\right\|_2^2\right)^{-1}}{\sum_{l'=1}^{L}\left(\left\|h_{\mathcal{G}_m} - C_{l'}\right\|_2^2\right)^{-1}} \tag{9}$$

$S_l\left[\mathcal{G}_m\right]$ denotes the soft clustering score that a graph example $\mathcal{G}_m$ is predicted to belong to the class $l$ and $L$ represents the number of categories of graph examples. Then the graph examples that achieve the high soft clustering score can be selected as the graph examples waiting to be pseudo labeled for each category.

**Algorithm 3** Procedure of semi-supervised learning in the ASGNN framework

**Input:**

$D_L$, $D_U$, $ss\_k$

**Output:**

New training set $D_{L'}$ and the original test set $D_U$ after semi-supervised learning

**Step 1 Process:**

Calculate the soft clustering results of each graph by GNN1 and GNN2 respectively according to Eq. 9.
GNN1 selects $ss\_k$ high confidence level examples $D_a$.
GNN2 selects $ss\_k$ high confidence level examples $D_b$.
Pseudo label $D_a \cap D_b$ and add them into $D_L$.

**Step 2 Output:**

Output the new training set $D_{L'}$ and the original test set $D_U$ after semi-supervised learning.

Likewise, our ASGNN framework considers the intersection of unlabeled graph examples selected by two GNNs as the graph examples waiting for pseudo labeling. After pseudo labeling the graph examples, the pseudo labeled graph examples with high confidence level are added to the training set. Every $invl$ (such as 5) epochs, we take out the pseudo labeled graph examples from training set, and reselect high-confidence pseudo labeled graph examples as shown in Algorithm 1. Because if the mis-predicted graph examples are not taken out later, the training error will accumulate in each iteration of training and the model performance will be bound to decline in the long run. The semi-supervised learning module designed in our framework simultaneously selects valuable unlabeled graph examples by multiple GNNs, and adds them into the training set after pseudo labeling to effectively improve the classification performance of graph neural networks. Algorithm 3 summarizes the detailed procedure of the semi-supervised learning module designed in our framework.

### 3.5 Discussion

Trained on only small number of labeled graph examples, the performance of graph neural networks is usually relevantly unsatisfactory or even poor although they can work to predict the labels for unlabeled graphs after training. Heuristically, our whole framework attempts to spirally improve the graph classification performance based on active learning and semi-supervised learning, two of classical weakly-supervised learning paradigms. Two aspects are taken into account while designing the active and semi-supervised graph neural network framework.

- In the early training of graph neural networks for graph classification, directly applying semi-supervised learning to GNNs may provide a lot of false pseudo labels for unlabeled graph examples, which easily lead to performance degradation due to the accumulation of the training error. To tackle this, an active learning strategy is presented to select the graph examples that contain rich information and deserve high attention, and then truly annotate them for enlarging the training set, which empower graph neural networks to enhance the generalization ability promptly.
- Nevertheless, it is not only expensive and but not necessary to continuously and truly annotate graph examples in the whole training process, especially in practical applications of graph classification. Therefore, we further design a semi-supervised learning strategy to take effective utilization of the trained GNN itself to make prediction, and select the high-confidence predicted graph examples for training GNNs with better classification performance. Note that we take out the pseudo-labeled graph examples at every certain epoch, which endeavors to avoid the performance decline due to the error accumulation.

Last but not least, the ASGNN framework utilizes two graph neural networks to collaboratively select the critical graph examples for updating model parameters, which is conducive to achieve reliable and desirable graph classification.

## 4 Experiments

In this section, we first introduce the experimental datasets and baselines. Then, graph classification results and the parameter sensitivity analysis are presented in detail. Finally, we conduct the ablation study and effectiveness verification of our proposed framework.

### 4.1 Datasets

To demonstrate the superiority of our proposed framework, twelve graph classification benchmarks are employed in the graph classification experiments, including MUTAG, PTC_MR, COLLAB, BZR_MD, BZR, NCI1, PROTEINS, ER_MD, COX2_MD, DHFR, DHFR_MD and PTC_FR. The MUTAG [33] dataset consists of 188 chemical compounds divided into two classes according to their mutagenic effect on a bacterium. The PTC_MR [33] dataset contains compounds labeled according to carcinogenicity on male rats. COLLAB [34] is a scientific collaboration dataset, derived from 3 public collaboration datasets [35], namely, High Energy Physics, Condensed Matter Physics and Astro Physics. BZR_MD is derived from the chemical compound dataset BZR, which is a set of 405 ligands for the benzodiazepine receptor [36]. NCI1 [37] is a chemical compound

TABLE 2: Statistics Information of Twelve Graph Classification Datasets.

| Dataset | Num_G | Num_Classes | Avg. nodes | Avg. edges | Node labels | Edge labels |
|---|---|---|---|---|---|---|
| MUTAG | 188 | 2 | 17.93 | 19.79 | + | + |
| PTC_MR | 344 | 2 | 14.29 | 14.69 | + | + |
| COLLAB | 5000 | 3 | 74.49 | 2457.78 | − | − |
| BZR_MD | 306 | 2 | 21.30 | 225.06 | + | + |
| BZR | 405 | 2 | 35.75 | 38.36 | + | − |
| NCI1 | 4110 | 2 | 29.87 | 32.30 | + | − |
| PROTEINS | 1113 | 2 | 39.06 | 72.82 | + | − |
| ER_MD | 446 | 2 | 21.33 | 234.85 | + | + |
| COX2_MD | 303 | 2 | 26.28 | 335.12 | + | + |
| DHFR | 467 | 2 | 42.43 | 44.54 | + | − |
| DHFR_MD | 393 | 2 | 23.87 | 283.01 | + | + |
| PTC_FR | 351 | 2 | 14.56 | 15.00 | + | + |

dataset and consists of 4110 compounds, which represent the activity against non-small cell lung cancer cell lines. PROTEINS is a dataset obtained from [38] where nodes are secondary structure elements and there is an edge between two nodes if they are neighbors in the amino-acid sequence or in 3D space. The bioinformatics dataset ER_MD [36] assembles 446 estrogen receptor ligands from multiple sources and the chemical dataset. COX2_MD [36] contains 303 cyclooxygenase-2 inhibitors. DHFR [36] is a set of 467 inhibitors of dihydrofolate reductase and DHFR_MD [36] is a subset of DHFR. PTC_FR [33] consists of compounds labeled according to carcinogenicity on rodents of male mice. TABLE 2 provides the summary statistics of all datasets used in graph classification experiments, including the total number of graphs, the number of classes of graphs, the average number of nodes in graphs, the average number of edges in graphs, the number of categories of nodes and the number of categories of edges. "–" denotes the lack of corresponding attributes.

### 4.2 Baselines

Two typical GNN representatives, GIN [13] and GFN [14], are respectively adopted as baselines. GIN is one of the provably most powerful GNNs under the neighborhood aggregation framework endeavoring to preserve the high-order neighborhood relations, while GFN can be seen as a simple lightweight GNN with linear graph filtering and non-linear set function, which is powerful enough to perform well than many sophisticated GNNs. To verify the superiority of our framework, a competitive label contrastive coding based graph neural network (LCGNN) [20] is compared, which shows more advantages than unsupervised GCC [19] and U2GNN [18]. LCGNN utilizes the label information effectively for graph classification tasks by extending the contrastive learning to the supervised setting and introducing the label contrastive coding. GIN+LCGNN and HGP-SL+LCGNN are two implementations of LCGNN in [20], thus we compare our ASGNN with GIN+LCGNN that uses GIN as the graph encoders, and HGP-SL+LCGNN that sets the graph encoders as HGP-SL [39]. Besides, a newly developed method named graph multi-set transformer (GMT) [40] is included for comparison, which explores the multi-head attention based global pooling for characterizing the interaction among nodes and outperforms the state-of-the-art graph pooling methods. Our ASGNN framework attempts to utilize a small amount of labeled data to complete graph classification tasks, which is different from the conventional methods that use 90% of labeled data. In order to compare under the same conditions, we settle the ratio of labeled training set and unlabeled test set to 10% and 90%, respectively. More importantly, we present the results of applying the two modules including active learning and semi-supervised learning designed in our framework to a single GNN (i.e., GIN+ASGNN and GFN+ASGNN), and further demonstrate the results of applying these two modules in multiple GNNs (GIN+GFN+ASGNN).

### 4.3 Comparison with Different Methods

TABLE 3 shows the experimental results of our framework and baselines for graph classification on twelve benchmark datasets. GIN+ASGNN implies the two modules including active learning and semi-supervised designed in our framework are applied to a single graph neural network GIN, and so does GFN+ASGNN. As for GIN+GFN+ASGNN, it represents the ASGNN framework that incorporates the graph neural networks GIN and GFN. For clarity, the boldface refers to the best graph classification accuracy among different methods on each dataset.

The graph classification results of GIN and GIN+ASGNN in TABLE 3 demonstrate that the classification accuracy is improved when the two modules including active learning and semi-supervised learning designed in our framework are applied to single GIN. The performance improvement is especially obvious on MUTAG, PROTEINS and PTC_MR datasets, which are $3.36\%$, $4.13\%$ and $3.74\%$, respectively. When the two modules including active learning and semi-supervised learning carefully designed in our framework are applied to the original GFN, the average accuracy of graph classification increases about $0.12\%$ to $7.89\%$. The diversity in the performance improvement of our framework may be caused by the difference of the size and structure of graph datasets. We also observe that, compared to the baseline methods GIN+LCGNN, GIN+ASGNN can acquire better results on most datasets, especially on the MUTAG, PROTEINS and PTC_FR datasets, which increase by $4.47\%$, $6.79\%$ and $4.23\%$, respectively. Furthermore, our framework implemented on single GIN/GFN yields competitive or even higher graph classification results than the state-of-the-art baseline methods GMT and HGP-SL+LCGNN. Overall, our ASGNN framework incorporated with multiple GNNs expresses the most evident and stable

TABLE 3: Graph Classification Accuracy of Baseline Methods and Our Proposed Framework on Different Datasets with Different Labeling Rate of 10%.

| Methods | MUTAG | PTC_MR | COLLAB | BZR_MD | BZR | NCI1 | PROTEINS | ER_MD | COX2_MD | DHFR | DHFR_MD | PTC_FR |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| GIN | 76.52±0.01 | 55.02±0.01 | 62.77±0.00 | 54.88±0.01 | 69.89±0.01 | 69.05±0.00 | 66.07±0.01 | 60.36±0.01 | 50.70±0.01 | 51.39±0.01 | 58.35±0.01 | 54.11±0.01 |
| GFN | 75.50±0.06 | 54.00±0.05 | 74.30±0.01 | 61.70±0.06 | 70.60±0.04 | 73.61±0.01 | 69.91±0.04 | 65.20±0.03 | 55.41±0.04 | 65.62±0.03 | 61.51±0.14 | 57.43±0.05 |
| GMT | 70.29±0.02 | 53.60±0.03 | 69.22±0.01 | 55.43±0.04 | 75.53±0.01 | 62.12±0.03 | 65.20±0.05 | 63.67±0.01 | 52.42±0.02 | 64.30±0.04 | 63.84±0.06 | 59.65±0.02 |
| GIN+LCGNN | 75.41±0.01 | 54.57±0.03 | 65.24±0.02 | 55.20±0.02 | 70.52±0.04 | 70.24±0.03 | 63.41±0.02 | 59.03±0.01 | 49.34±0.02 | 52.50±0.06 | 60.36±0.03 | 53.62±0.01 |
| HGP-SL+LCGNN | 68.67±0.02 | 55.91±0.02 | 67.21±0.03 | 54.31±0.02 | 72.15±0.03 | 60.74±0.01 | 64.23±0.01 | 60.58±0.05 | 50.36±0.02 | 55.68±0.01 | 61.52±0.06 | 56.18±0.04 |
| GIN+ASGNN | 79.88±0.01 | 57.75±0.02 | 65.32±0.06 | 56.44±0.01 | 72.07±0.00 | 70.34±0.01 | 70.20±0.01 | 62.17±0.01 | 50.88±0.01 | 53.34±0.01 | 59.81±0.02 | 57.85±0.01 |
| GFN+ASGNN | 79.04±0.04 | 57.60±0.04 | 74.42±0.01 | 65.09±0.04 | 78.49±0.06 | 74.27±0.01 | 72.81±0.02 | 66.72±0.03 | 56.75±0.04 | 69.28±0.03 | 62.66±0.04 | 58.93±0.12 |
| GIN+GFN+ASGNN | **81.83±0.03** | **59.10±0.04** | **75.56±0.01** | **66.86±0.05** | **79.73±0.02** | **75.53±0.01** | **73.92±0.03** | **67.84±0.03** | **57.92±0.04** | **69.89±0.03** | **66.01±0.02** | **61.15±0.04** |

improvement in the task of graph classification. The reason may be that multiple GNNs are more reliable in selecting the valuable graph examples by mutual learning in active learning and semi-supervised learning. In specific, multiple GNNs can confidently select the unlabeled graph examples from different views, to simultaneously improve the model performance of GNNs. Moreover, the gain of these graph examples to the model performance is often higher and more stable than the gain of the graph examples selected by a single GNN to the performance of graph classification.

Experimental results in TABLE 3 fully reveals the effectiveness of exploiting active and semi-supervised learning for graph classification. And the two paradigms including active learning and semi-supervised learning can be applied to a single graph neural network to promote the generalization performance and improve the classification accuracy of graph classification tasks. Overall, two obvious conclusions can be concluded from the experimental results. The proposed framework incorporating two GNNs can promote the graph classification accuracy than that only based on single GNN. Active learning and semi-supervised learning on multiple graph neural networks can facilitate the model training and achieve an advanced performance increment for graph classification results.

### 4.4 Parameter Sensitivity Analysis

In this section, we investigate how hyper-parameters affect the performance of our proposed framework for graph classification. Specifically, we evaluate how three key parameters $AL\_K$, $SS\_K$ and $\alpha$ influence the graph classification accuracy.

For the hyper-parameter $AL\_K$ in the proposed framework, $AL\_K$ represents the proportion budget of the graph examples selected from the test set by active learning and we study the impact of its different values $AL\_K \in \{1\%, 2\%, 5\%, 8\%, 10\%\}$ on graph classification. Considering that the results of different datasets show similar tends, we take MUTAG, PTC_MR, PROTEINS and DHFR datasets as examples for analysis. Fig. 3 manifests how the parameter $AL\_K$ influences the graph classification accuracy on four datasets when our framework incorporated with the single GIN, single GFN and multiple GNNs. It demonstrates that the performance of our proposed framework achieves an improvement with the rise of the number of representative examples marked by active learning.

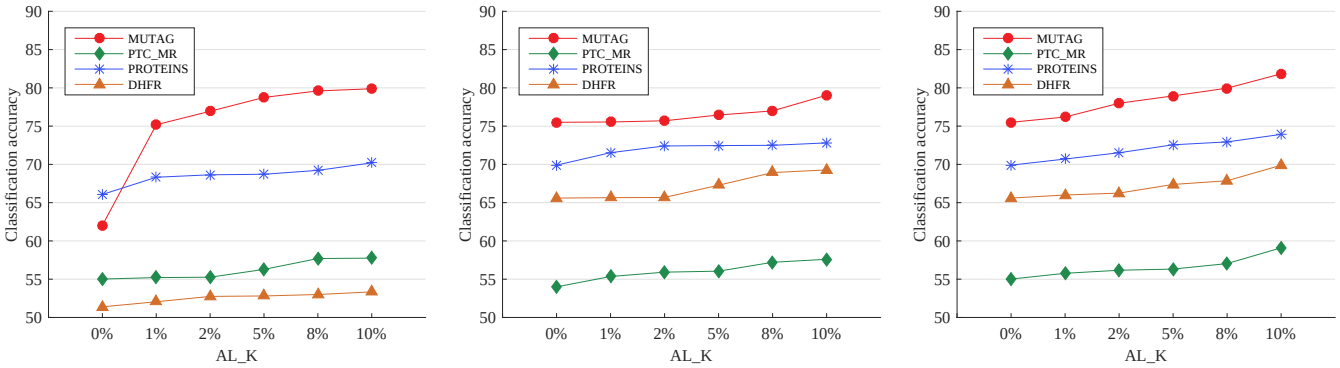
We further investigate the influences of different $SS\_K$ of our proposed framework on MUTAG, PTC_MR, PROTEINS and DHFR datasets for graph classification. $SS\_K$ represents the proportion budget of the graph examples selected from the test set by semi-supervised learning and we set $SS\_K \in \{2\%, 4\%, 6\%, 8\%, 10\%\}$. Fig. 4 presents how the parameter $SS\_K$ influences the graph classification accuracy on MUTAG, PTC_MR, PROTEINS and DHFR datasets when our framework incorporated with the single GIN, single GFN and multiple GNNs. It exhibits that the performance of our proposed framework brings an improvement with the increase of the number of pseudo labeled examples marked by semi-supervised learning.

Finally, the influences of the weight of entropy percentage $\alpha$ and euclidean percentage $1-\alpha$ on the graph classification accuracy are explored. We set $\alpha \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$ and the weight of euclidean percentage is correspondingly as $1-\alpha \in \{0.9, 0.7, 0.5, 0.3, 0.1\}$. Fig. 5 indicates the influence for graph classification accuracy on MUTAG, PTC_MR, PROTEINS, and DHFR datasets with different proportions of $\alpha$ when our framework incorporated with the single GIN, single GFN and multiple GNNs. Our framework yields the best graph classification accuracy when the weight $\alpha$ is $0.5$. Additionally, when one of the two indicators (entropy percentage or euclidean percentage) is assigned as high weights, the classification accuracy of our framework is relatively decreased.

To summarize, with two hyper-parameters $AL\_K$ and $SS\_K$ increasing from $0\%$ to $10\%$, the accuracy of graph classification achieves an incremental improvement. The increment of $AL\_K$ and $SS\_K$ indicates that more examples with rich information are selected from test set and added into the training set for promoting the performance of graph neural networks. It proves that active learning and semi-supervised learning can utilize a small number of labeled graph examples or pseudo labeled graph examples to expand the training set for promoting the performance of graph neural networks. In Fig. 3(a) and Fig. 4(a), on the small dataset MUTAG, the performance of our framework incorporated with single GIN has a significant improvement with the increase of $AL\_K$ and $SS\_K$ at the beginning. Fig. 3(c) and Fig. 4(c) show us that when the multiple GNNs are incorporated with our framework, with the labeling rate increasing to $10\%$, the classification accuracy still has the potential to improve with the increment of $AL\_K$ and $SS\_K$. Overall, the ASGNN framework proposed in this paper incorporated with single GNN or multiple GNNs all achieve an improvement of graph classification accuracy. In Fig. 5, with the weight of entropy percentage increases from $0.1$ to $0.5$, the performance of our framework is gradually improved, due to more attention paid to the effect of entropy percentage on the process of selecting examples with rich information by active learning. When $\alpha$ is near $0.5$, we obtain the best classification performance. However, when the value of $\alpha$ continuously increases, the classification accuracy of our framework will drop slowly. The premier reason may be that the model does not reasonably consider

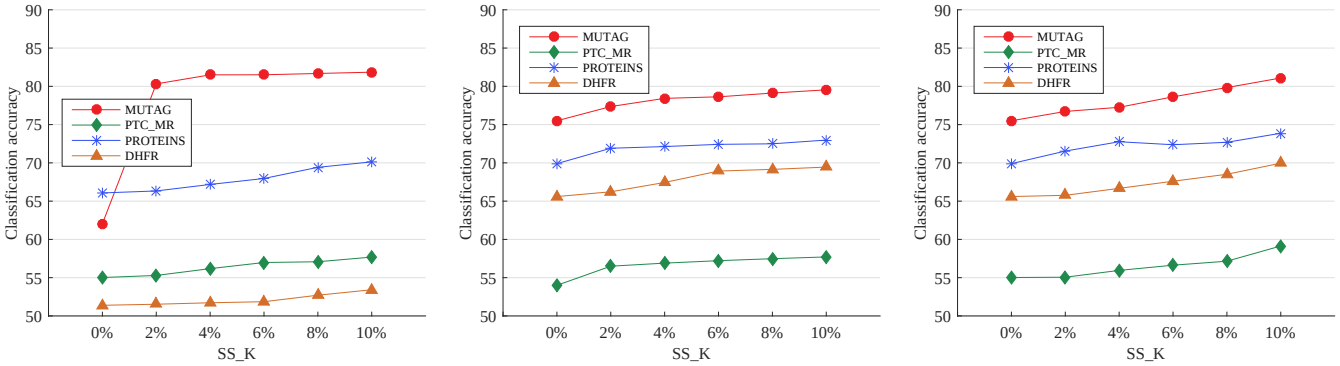
This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TBDATA.2021.3140205, IEEE Transactions on Big Data





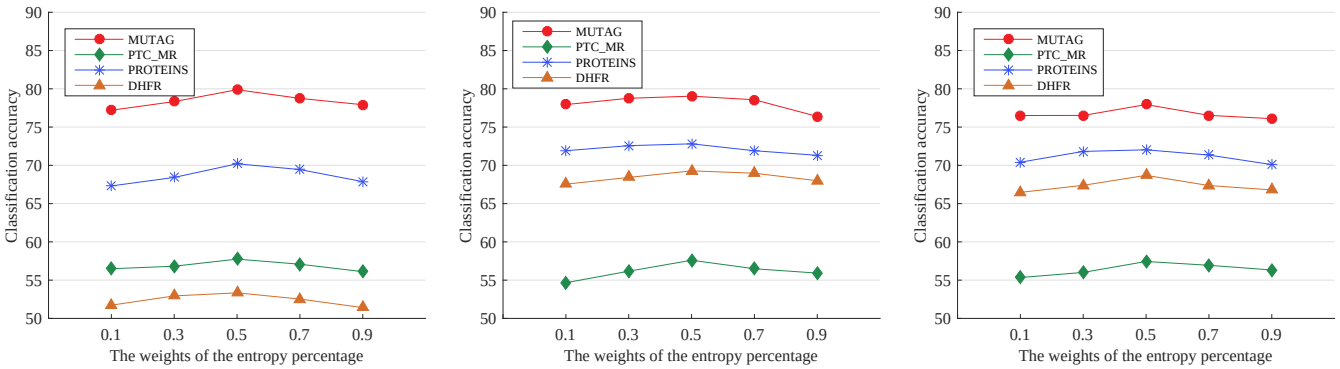

(a) Active learning selects examples by single GIN (b) Active learning selects examples by single GFN (c) Active learning selects examples by multiple GNNs

Fig. 3: Sensitivity analysis of $AL\_K$ from $0\%$ to $10\%$ on MUTAG, PTC_MR, PROTEINS and DHFR datasets.




(a) Semi-supervised learning selects examples by single GIN (b) Semi-supervised learning selects examples by single GFN (c) Semi-supervised learning selects examples by multiple GNNs

Fig. 4: Sensitivity analysis of $SS\_K$ from $0\%$ to $10\%$ on MUTAG, PTC_MR, PROTEINS and DHFR datasets.




(a) Active learning selects examples by single GIN (b) Active learning selects examples by single GFN (c) Active learning selects examples by multiple GNNs

Fig. 5: Sensitivity analysis of $\alpha$ from $0.1$ to $0.9$ on MUTAG, PTC_MR, PROTEINS and DHFR datasets.

the influence of two indicators on the process of selecting examples with rich information by active learning, but pays more attention to the euclidean percentage on the selection.

### 4.5 Ablation Study

To demonstrate the effectiveness of two learning steps designed in our framework, we conduct ablation study on active learning and semi-supervised learning, respectively. Finding that results of different datasets show similar tends, we take MUTAG, PTC_MR, PROTEINS and DHFR as examples. The effect of removing different components in our framework on the performance of graph classification is shown in TABLE 4. It is obvious that both active and semi-supervised learning steps have positive contributions to the single GNN for graph classification, respectively. With the integration of active learning and semi-supervised

TABLE 4: Ablation Study on Active Learning and Semi-supervised Learning Steps of Our Framework on Different Datasets with Labeling Rate of 10%. Bold Numbers are the Best or the Second Best Graph Classification Accuracy for clarity.

| GIN | GFN | Active learning | Semi-supervised learning | MUTAG | PTC_MR | PROTEINS | DHFR |
|---|---|---|---|---|---|---|---|
| ✓ | | | | 76.52±0.01 | 55.02±0.01 | 66.07±0.01 | 51.39±0.01 |
| ✓ | | ✓ | | 78.26±0.02 | 56.51±0.01 | 69.02±0.03 | 52.29±0.05 |
| ✓ | | | ✓ | 78.20±0.02 | 56.96±0.04 | 68.31±0.05 | 52.41±0.01 |
| ✓ | | ✓ | ✓ | **79.88±0.01** | **57.75±0.02** | 70.20±0.01 | 53.34±0.01 |
| | ✓ | | | 75.50±0.06 | 54.00±0.05 | 69.91±0.04 | 65.62±0.03 |
| | ✓ | ✓ | | 77.44±0.02 | 56.00±0.03 | 70.51±0.01 | 67.39±0.02 |
| | ✓ | | ✓ | 77.03±0.03 | 56.98±0.02 | 70.19±0.04 | 67.31±0.05 |
| | ✓ | ✓ | ✓ | 79.04±0.04 | 57.60±0.04 | **72.81±0.02** | **69.28±0.03** |
| ✓ | ✓ | ✓ | ✓ | **81.83±0.03** | **59.10±0.04** | **73.92±0.03** | **69.89±0.03** |
| (Improvement over the second best baselines) | | | | **(1.95)** | **(1.35)** | **(1.11)** | **(0.61)** |

learning on single GNN, the accuracy of graph classification is further enhanced, indicating that it is feasible to exploit the two paradigms together to improve the generalization performance. Furthermore, our framework implemented on two GNNs yields more desirable graph classification accuracy than that only based on single GNN, achieving the gain of 0.61%-1.95% at least. All these results fully verify that our framework incorporating with the carefully designed active learning and semi-supervised learning strategies can effectively select and exploit the valuable graph examples in different phases to facilitate the model training of GNNs and enhance the graph classification accuracy.

### 4.6 Verification of Effectiveness on Used Labeled Data

TABLE 5: Graph Classification Accuracy of Different Methods on Datasets with Different Amount of Labeled Data.

| Methods | MUTAG | PTC_MR | PROTEINS | DHFR |
|---|---|---|---|---|
| GIN | 85.02±0.01 (90%) | 59.33±0.03 (90%) | 71.80±0.01 (90%) | 55.61±0.01 (90%) |
| GIN+ASGNN | 85.82±0.03 (50%) | 60.78±0.02 (50%) | 72.42±0.01 (50%) | 57.87±0.02 (30%) |
| GFN | 87.21±8.22 (90%) | 62.45±9.46 (90%) | 75.46±5.06 (90%) | 75.30±4.19 (90%) |
| GFN+ASGNN | 87.94±0.05 (50%) | 63.60±0.03 (50%) | 75.70±0.02 (50%) | 75.51±0.01 (30%) |
| GIN+GFN +ASGNN | **88.20±0.04 (50%)** | **64.40±0.01 (30%)** | **76.20±0.01 (25%)** | **76.89±0.02 (30%)** |

In this section, we explore the relation between the performance of the graph classification model and the amount of training data required. Considering that results of different datasets show similar tends, we take MUTAG, PTC_MR, PROTEINS and DHFR as examples for illustration. TABLE 5 reports the experimental results of our framework and different methods for graph classification with different amount of labeled data on MUTAG, PTC_MR, PROTEINS and DHFR datasets. The amount of labeled data used in GIN and GFN is $90\%$, $90\%$, $90\%$ and $90\%$ for MUTAG, PTC_MR, PROTEINS and DHFR datasets, respectively. To achieve the comparable classification accuracy, the labeled data required in GIN+ASGNN and GFN+ASGNN is $50\%$, $50\%$, $50\%$ and $30\%$ for MUTAG, PTC_MR, PROTEINS and DHFR datasets, respectively. In GIN+GFN+ASGNN, the amount of labeled data required is $50\%$, $30\%$, $25\%$ and $30\%$ on MUTAG, PTC_MR, PROTEINS and DHFR datasets. In this case, the graph classification results in GIN+ASGNN and GFN+ASGNN outperform GIN and GFN, respectively. And GIN+GFN+ASGNN can achieve the best classification accuracy only using the minimal number of labeled graph examples. The main reason is that the two paradigms including active learning and semi-supervised learning in our framework can select the informative graph examples and many pseudo labeled graph examples with structure information for promoting the performance of graph classification. These results further verify the effectiveness of our framework.

## 5 Conclusion

Graph neural networks play an increasingly important role in solving the problem of graph classification and existing works widely utilize a large volume of labeled graph examples for training. However, there are rare labeled graph examples in the real-world application of graph classification and it is very expensive to label a large number of graph examples manually. In this paper, we propose a novel active and semi-supervised graph neural network framework to complete the graph classification tasks with a small number of labeled graph examples and available unlabeled graph examples. The two learning paradigms including active learning and semi-supervised learning designed in our framework exploit multiple GNNs for collaboratively increasing the reliability of graph classification results. The framework can not only gracefully explore unlabeled graph examples for facilitating graph classification tasks, but also can be utilized to other existing graph neural networks for graph classification. Experimental results on benchmark graph classification datasets manifest that the proposed framework is effective on graph classification tasks only with a small number of labeled examples and available unlabeled graph examples.

For future works, we would like to further extend the proposed framework for more complex and challenging graph classification problems. Besides, the design of other active and semi-supervised graph neural network frameworks for efficient graph classification is also a promising research direction.

## Acknowledgments

The authors are very grateful for the editors and reviewers for their insightful comments and suggestions. This work was supported by the National Key Research and Development Program of China (Grant No. 2020AAA0106100), the Key Program of the National Natural Science Foundation of China (Grant No. 62136005), the National Natural Science Foundation of China (Grant Nos. 62106131 and 62106134),

and the Basic Research Program of Shanxi Province, China (Grant Nos. 20210302124032 and 20210302124394).

## REFERENCES


[1] D. Zhang, J. Yin, X. Zhu, and C. Zhang, "Network representation learning: A survey," *IEEE Trans. Big Data*, vol. 6, no. 1, pp. 3–28, 2020.

[2] J. Cheng, M. Chen, M. Zhou, S. Gao, C. Liu, and C. Liu, "Overlapping community change-point detection in an evolving network," *IEEE Trans. Big Data*, vol. 6, no. 1, pp. 189–200, 2020.

[3] V. La Gatta, V. Moscato, M. Postiglione, and G. Sperli, "An epidemiological neural network exploiting dynamic graph structured data applied to the COVID-19 outbreak," *IEEE Trans. Big Data*, vol. 7, no. 1, pp. 45–55, 2021.

[4] H. Cai, V. W. Zheng, and K. C. Chang, "A comprehensive survey of graph embedding: Problems, techniques, and applications," *IEEE Trans. Knowl. Data Eng.*, vol. 30, no. 9, pp. 1616–1637, 2018.

[5] K. M. Borgwardt and H. P. Kriegel, "Shortest-path kernels on graphs," in *Proc. 5th IEEE Int. Conf. Data Min.*, 2005, pp. 74–81.

[6] N. Shervashidze, S. Vishwanathan, T. Petri, K. Mehlhorn, and K. Borgwardt, "Efficient graphlet kernels for large graph comparison," in *Proc. 12th Int. Conf. Artif. Intell. Statist.*, 2009, pp. 488–495.

[7] G. Nikolentzos, P. Meladianos, J. P. Tixier, K. Skianis, and M. Vazirgiannis, "Kernel graph convolutional neural networks," in *Proc. Int. Conf. Artif. Neural Netwoks*, 2018, pp. 22–32.

[8] M. Neumann, R. Garnett, C. Bauckhage, and K. Kersting, "Propagation kernels: Efficient graph kernels from propagated information," *Mach. Learn.*, vol. 102, pp. 209–245, 2016.

[9] G. Siglidis, G. Nikolentzos, S. Limnios, C. Giatsidis, K. Skianis, and M. Vazirgiannis, "GraKel: A graph kernel library in python." *J. Mach. Learn. Res.*, vol. 21, no. 54, pp. 1–5, 2020.

[10] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, "Neural message passing for quantum chemistry," in *Proc. 34th Int. Conf. Mach. Learn.*, 2017, pp. 1263–1272.

[11] M. Zhang, Z. Cui, M. Neumann, and Y. Chen, "An end-to-end deep learning architecture for graph classification," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018, pp. 4438–4445.

[12] L. Bai, L. Cui, Y. Jiao, L. Rossi, and E. Hancock, "Learning backtrackless aligned-spatial graph convolutional networks for graph classification," *IEEE Trans. Pattern Anal. Mach. Intell.*, 2020.

[13] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How powerful are graph neural networks?" in *Proc. 7th Int. Conf. Learn. Represent.*, 2019.

[14] T. Chen, S. Bian, and Y. Sun, "Are powerful graph neural nets necessary? A dissection on graph classification," *arXiv preprint arXiv:1905.04579*, 2019.

[15] Y. Xie, C. Yao, M. Gong, C. Chen, and A. K. Qin, "Graph convolutional networks with multi-level coarsening for graph classification," *Knowl. Based Syst.*, vol. 194, p. 105578, 2020.

[16] N. Annamalai, C. Mahinthan, V. Rajasekar, C. Lihui, L. Yang, and J. Shantanu, "graph2vec: Learning distributed representations of graphs," in *Proc. 13th Int. Workshop Min. Learn. Graphs*, 2017.

[17] H. Dai, H. Li, T. Tian, X. Huang, L. Wang, J. Zhu, and L. Song, "Adversarial attack on graph structured data," in *Proc. 35th Int. Conf. Mach. Learn.*, 2018, pp. 1123–1132.

[18] T. D. N. Dai Quoc Nguyen and D. Phung, "Unsupervised universal self-attention network for graph classification," *arXiv preprint arXiv:1909.11855*, 2019.

[19] J. Qiu, Q. Chen, Y. Dong, J. Zhang, H. Yang, M. Ding, K. Wang, and J. Tang, "GCC: Graph contrastive coding for graph neural network pre-training," in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discovery Data Min.*, 2020, pp. 1150–1160.

[20] Y. Ren, J. Bai, and J. Zhang, "Label contrastive coding based graph neural network for graph classification," in *Proc. 26th Int. Conf. Database Syst. Adv. Appl.*, 2021.

[21] K. Konyushkova, R. Sznitman, and P. Fua, "Learning active learning from data," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 4225–4235.

[22] A. J. Joshi, F. Porikli, and N. P. Papanikolopoulos, "Scalable active learning for multiclass image classification," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, pp. 2259–2273, 2012.

[23] M. Goudjil, M. Koudil, M. Bedda, and N. Ghoggali, "A novel active learning method using SVM for text classification," *Int. J. Autom. Comput.*, vol. 15, pp. 1–9, 2016.

[24] H. Cai, V. W. Zheng, and K. C. Chang, "Active learning for graph embedding," *CoRR*, 2017. [Online]. Available: http://arxiv.org/abs/1705.05085

[25] M. Zhu, X. Wang, C. Shi, H. Ji, and P. Cui, "Interpreting and unifying graph neural networks with an optimization framework," in *Proc. 30th World Wide Web Conf.*, 2021, pp. 1215–1226.

[26] J. Xu, S. Zhu, H. Guo, and S. Wu, "Automated labeling for robotic autonomous navigation through multi-sensory semi-supervised learning on big data," *IEEE Trans. Big Data*, vol. 7, no. 1, pp. 93–101, 2021.

[27] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu, "A comprehensive survey on graph neural networks," *IEEE Trans. Neural Networks Learn. Syst.*, vol. 32, no. 1, pp. 4–24, 2021.

[28] X. Wang, M. Zhu, D. Bo, P. Cui, C. Shi, and J. Pei, "AM-GCN: Adaptive multi-channel graph convolutional networks," in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discovery Data Min.*, 2020, pp. 1243–1253.

[29] R. Liao, M. Brockschmidt, D. Tarlow, A. L. Gaunt, R. Urtasun, and R. S. Zemel, "Graph partition neural networks for semi-supervised classification," in *Proc. 6th Int. Conf. Learn. Represent.*, 2018.

[30] M. Guillaumin, J. Verbeek, and C. Schmid, "Multimodal semi-supervised learning for image classification," in *Proc. 23rd Comput. Vis. Pattern Recognit.*, 2010, pp. 902–909.

[31] Zhu, Songhao, Sun, Xian, Jin, and Dongliang., "Multi-view semi-supervised learning for image classification." *Neurocomputing*, vol. 208, pp. 136–142, 2016.

[32] N. H. Minsky and P. P. Pal, "Providing multiple views for objects," *Software Pract. Exper.*, vol. 30, pp. 803–823, 2015.

[33] N. Kriege and P. Mutzel, "Subgraph matching kernels for attributed graphs," in *Proc. 29th Int. Conf. Mach. Learn.*, 2012, p. 291298.

[34] P. Yanardag and S. Vishwanathan, "Deep graph kernels," in *Proc. 21th ACM SIGKDD Int. Conf. Knowl. Discovery Data Min.*, 2015, pp. 1365–1374.

[35] J. Leskovec, J. Kleinberg, and C. Faloutsos, "Graphs over time: Densification laws, shrinking diameters and possible explanations," in *Proc. 11th ACM SIGKDD Int. Conf. Knowl. Discovery Data Min.*, 2005, pp. 177–187.

[36] J. J. Sutherland, L. A. O'brien, and D. F. Weaver, "Spline-fitting with a genetic algorithm: A method for developing classification structure-activity relationships," *J. Chem. Inf. Comput. Sci.*, vol. 43, no. 6, pp. 1906–1915, 2003.

[37] N. Wale, I. A. Watson, and G. Karypis, "Comparison of descriptor spaces for chemical compound retrieval and classification," *Knowl. Inf. Syst.*, vol. 14, pp. 347–375, 2008.

[38] K. M. Borgwardt, C. S. Ong, S. Schönauer, S. Vishwanathan, A. J. Smola, and H.-P. Kriegel, "Protein function prediction via graph kernels," *Bioinf.*, vol. 21, pp. 147–156, 2005.

[39] Z. Zhang, J. Bu, M. Ester, J. Zhang, C. Yao, Z. Yu, and C. Wang, "Hierarchical graph pooling with structure learning," in *Proc. 34th AAAI Conf. Artif. Intell.*, 2020.

[40] J. Baek, M. Kang, and S. J. Hwang, "Accurate learning of graph representations with graph multiset pooling," in *Proc. 9th Int. Conf. Learn. Represent.*, 2021.


**Yu Xie** received the Ph.D. degree in pattern recognition and intelligent systems from Xidian University, Xi'an, China, in 2020. From 2019 to 2020, he was a research assistant with School of Computer Science and Engineering, Nangyang Technological University. He is currently a teacher with School of Computer and Information Technology, Shanxi University. His research interests include graph neural networks and artificial intelligence security.

**Shengze Lv** was born in 1996. He is currently working toward the M.S. degree in computer science and technology, Xidian University, Xi'an, China. His research interests include graph representation learning and graph classification.

**Yuhua Qian** received the M.S. and Ph.D. degrees from Shanxi University, Taiyuan, China, in 2005 and 2011, respectively. He is currently a Professor with the Key Laboratory of Computational Intelligence and Chinese Information Processing of Ministry of Education, Shanxi University. He is best known for multigranulation rough sets in learning from categorical data and granular computing. He is involved in research on pattern recognition, feature selection, rough set theory, granular computing, and artificial intelligence. He has authored over 100 articles on these topics in international journals and coferences. He served on the Editorial Board of the International Journal of Knowledge-Based Organizations and Artificial Intelligence Research. He has served as the Program Chair or Special Issue Chair of the Conference on Rough Sets and Knowledge Technology, the Joint Rough Set Symposium, and the Conference on Industrial Instrumentation and Control, and PC Members of many machine learning, data mining, and granular computing conferences.

**Chao Wen** received the Ph.D. degree from Xidian University, China, in 2017. He is currently a Lecturer with the Institute of Big Data Science and Industry, Shanxi University, China. His research interests include artificial intelligence, array signal processing and passive sensing.

**Jiye Liang** received the M.S. and Ph.D. degrees from Xi'an Jiaotong University, Xi'an, China, in 1990 and 2001, respectively. He is currently a Professor with the School of Computer and Information Technology, Shanxi University, Taiyuan, China, where he is also the Director of the Key Laboratory of Computational Intelligence and Chinese Information Processing of the Ministry of Education. His research interests include computational intelligence, granular computing, data mining, and knowledge discovery. He has authored more than 200 journal and conference papers in his research fields, including the IEEE Transactions on Pattern Analysis and Machine Intelligence, IEEE Transactions on Knowledge and Data Engineering, Artificial Intelligence, Journal of Machine Learning Research, Machine Learning, and International Conference on Machine Learning, etc.