M.S. THESIS

Accurate Graph Classification via Two-Staged Contrastive Curriculum Learning

두 단계로 이루어진 대조 커리큘럼 학습을 통한 정확한 그래프 분류

February 2023

Interdisciplinary Program in Artificial Intelligence
College of Engineering
Seoul National University

Sooyeon Shim

Accurate Graph Classification via Two-Staged Contrastive Curriculum Learning

두 단계로 이루어진 대조 커리큘럼 학습을 통한 정확한 그래프 분류

지도교수 강 유

이 논문을 공학석사 학위논문으로 제출함

2023년 2월

서울대학교 대학원

협동과정 인공지능전공

심수연

심수연의 석사 학위논문을 인준함

2023년 2월

위 원 장 곽 노 준 (인)

부 위 원 장 강 유 (인)

위 원 문 태 섭 (인)

Abstract

Accurate Graph Classification via Two-Staged Contrastive Curriculum Learning

Sooyeon Shim
Interdisciplinary Program in Artificial Intelligence
College of Engineering
Seoul National University

Given a graph dataset, how can we generate meaningful graph representations that maximize classification accuracy? Learning representative graph embeddings is important to solve various real-world graph-based tasks. Graph contrastive learning aims to learn representations of unlabeled graphs by capturing the relationship between graphs. Recently, data augmentation has been widely used for contrastive learning. However, previous contrastive learning methods with random-based augmentations fail to capture semantic information within graphs. Furthermore, graph contrastive learning approaches with carefully designed augmentations are computationally inefficient.

We propose TAG (Two-staged contrAstive curriculum learning for Graphs), a two-staged contrastive learning method for graph classification. TAG exploits six model-agnostic augmentation algorithms that preserve the graph semantics by considering the degree centrality. The augmentation algorithms are used for the two-

staged learning of graph representations: node-level and graph-level. Experiments show that TAG outperforms both unsupervised and supervised methods in classification accuracy, achieving up to 4.21% and 4.76% points higher than the second-best unsupervised and supervised methods on average, respectively.

Keywords : Graph Classification, Contrastive Learning, Curriculum Learning

Student Number : 2021-28209

Contents

List of Figures

List of Tables

Chapter 1

Introduction

How can we generate graph representations for accurate graph classification? Various GNNs have been proposed to solve graph-based tasks such as link prediction [1, 2], node classification [3–7], and graph classification [8–13]. However, most GNNs for graph classification show high performance in only one of the two settings: unsupervised and supervised settings. Furthermore, graph classification models designed for supervised settings are not even applicable to unsupervised settings. There arises the need for designing an accurate graph classification model for both settings.

Graph contrastive learning learns the representations of graphs based on the similarity between graphs. The learning algorithm can be used in both settings: unsupervised and supervised settings [14–16]. Recent graph contrastive learning methods utilize data augmentation to ensure the similarity of the original graph and the newly generated graph. Random-based augmentations are used to generate graphs in [17–19], but information loss is inevitable in those methods. Graph contrastive learning with carefully designed augmentations [8, 20, 21] preserve more graph semantics compared to those with random-based ones. However, these methods increase the complexity of the model.

In this paper, we propose TAG (Two-staged contrAstive curriculum learning for Graphs), an accurate graph contrastive learning approach that can be applied to both supervised and unsupervised graph classification. We design six model-agnostic augmentation algorithms that preserve the semantic information of graphs.

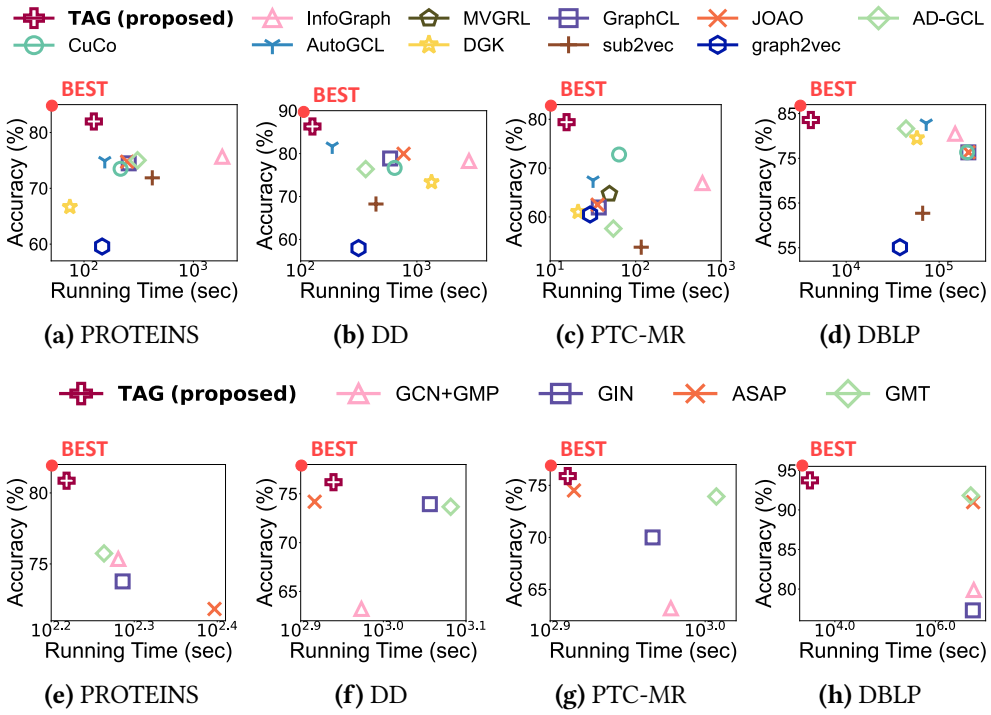

Figure 1: Overall performance of TAG with unsupervised and supervised graph classification methods. (a-d) are the performance in unsupervised setting, and (e-f) are that in supervised one. Note that TAG shows the highest classification accuracy with the shortest running time in both settings.

Three algorithms change the features of nodes, and the other three modify the structure of graphs based on degree centrality. We then conduct graph contrastive learning in two levels: node-level and graph-level. Node-level contrastive learning learns node embeddings based on the relationship between nodes. Graph-level contrastive learning learns the embeddings of graphs based on the computed node embeddings. Thus, the relationships of both nodes and graphs are reflected in the graph representations. Furthermore, TAG exploits a curriculum learning strategy to enhance performance. Figure 1 shows the overall performance of TAG; note that TAG outperforms the com-

Table 1: Description of symbols.

Symbol	Description
$\mathcal{D}$	a set of graphs for training
G_i	i-th graph in a set $\mathcal{D}$
$G_{f,i}$	feature-modified graph originated from G_i
$G_{s,i}$	structure-modified graph originated from G_i
v_j	j-th node in a graph G_i
u_j	j-th node in a graph $G_{f,i}$

petitors in both unsupervised and supervised settings. Table 1 describes symbols used in this paper.

Our main contributions are summarized as follows:

- **Data augmentation.** We propose six model-agnostic augmentation algorithms for graphs. Every augmentation method considers node centrality to preserve semantic information of original graphs.
- **Method.** We propose TAG, a two-staged contrastive curriculum learning method for accurate graph classification. The two-staged approach embeds the relational information of both nodes and graphs into the graph representations.
- **Experiments.** We perform experiments on seven benchmark datasets in supervised and unsupervised settings, achieving the best performance for most cases.

The rest of the paper is organized as follows. In Chapter 2, we introduce previous works related to ours. In Chapter 3, we propose TAG in detail. We show the experimental results in Chapter 4 and conclude in Chapter 5. The code of TAG is available at https://github.com/anonymous-tag-2022/TAG-2022.

Chapter 2

Related Works

2.1 Graph Contrastive Learning

Graph contrastive learning methods are categorized into node-level and graph-level contrastive learning methods for node and graph classification tasks, respectively.

2.1.1 Node-level Graph Contrastive Learning

Node-level graph contrastive learning methods are designed to handle node classification task in the unsupervised setting. DGI [10] is the first work that applies the concept of contrastive learning to the graph domain. JGCL [22] joints supervised setting, semi-supervised setting, and unsupervised setting to learn the optimal node representations. GMI [23] defines the concept of graph mutual information(GMI) and aims to maximize the mutual information in terms of node features and topology of the graph. GCC [24] learns transferable structural representation across various networks to guide the pre-training of graph neural networks. GRACE [25] jointly considers both topology and node attribute levels for corruption to generate graph views and maximizes the agreement in the views at the node level. Zhu et al. [26] propose GCA that removes unimportant edges by giving them large removable probabilities on the topology level and adds more noise to unimportant feature dimensions on the node attribute level for the adaptive augmentation. BGRL [27] is a scalable method with two encoders that learns by predicting alternative augmentations of the input.

Graph Barlow Twins(G-BT) [28] is a model that replaces negative samples with a cross-correlation-based loss function and does not introduce asymmetry in the network. TIFA-GCL [29] improves the performance of semi-supervised node classification task by focusing on the information distribution of a few labeled nodes from the global and local view. GCNSS [30] obtains node embeddings by designing a strategy for selecting negative samples which selects some nodes instead of using all nodes as negative samples and performing node-level contrastive learning. STENCIL [31] adopts multiview contrastive objective and learns node embeddings of a heterogenous graph by regarding a node in different views as positive samples and all other nodes in a graph as negative samples. PASCAL [32] merges all motifs that include the target node to construct subgraphs for the target node and captures topological information within local structures. SAIL [33] operates self-augmentation through graph knowledge distillation composed of two modules: intra-distilling module and inter-distilling module. Previous node-level graph contrastive learning approaches do not handle graph classification task. We jointly perform node-level contrastive learning with the graph-level for graph classification.

2.1.2 Graph-level Graph Contrastive Learning

Previous graph-level contrastive learning methods are divided into two types: model-specific and model-agnostic ones. Model-agnostic approaches use augmentation algorithms which do not engage in the training process. GraphCL [19] brings the contrastive learning method for images to the graph domain. CuCo [18] extends GraphCL by applying curriculum learning to properly learn the negative samples. MVGRL [17] learns graph-level representations by contrasting encodings from first-order neighbors and graph diffusion. These methods use random-based graph augmentations

that cannot preserve the core information of graphs well. We propose a graph contrastive learning method along with degree-based augmentations to address the issue. Model-specific augmentation approaches directly participate in the training process. InfoGraph [9] learns graph representations by contrasting them with patch-level representations obtained from the training process. You et al. [20] propose JOAO which changes the simple augmentations to be learnable. AD-GCL [21] adopts the structure of an adversarial attack to obtain graph representations. AutoGCL [8] generates new graphs by changing the softmax function into the Gumbel-Softmax function. These approaches are more complex than model-agnostic models. Therefore, we propose a contrastive learning method with simple augmentations for computational efficiency.

2.2 Graph Augmentation

Graph augmentation algorithms are divided into two types: model-specific and model-agnostic augmentation. Model-specific augmentation algorithms are restricted to a certain model. Thus, it is not easy for them to be generalized to contrastive learning directly. Model-agnostic graph augmentations are applied to any graph neural network. You et al. [19] suggest DropNode and ChangeAttr for graph contrastive learning. DropNode discards randomly selected nodes with their connections and ChangeAttr converts features of randomly selected nodes into random values. DropEdge [34] changes graph topology by removing a certain ratio of edges. GraphCrop [35] selects a subgraph from a graph through a random walk. Wang et al. [36] introduce NodeAug which contains three different augmentations: ReplaceAttr, RemoveEdge, and AddEdge. ReplaceAttr substitutes the feature of a chosen node with the average of its neighboring nodes' features. RemoveEdge discards edges based on the importance score of the edges. AddEdge attaches new edges to a central node which is

designated based on the importance score for nodes. Motif-similarity [37] adds and deletes edges from motifs that are frequent in a particular graph. Yoo et al. [38] proposes NodeSam and SubMix. NodeSam performs split and merge operations on nodes. SubMix replaces a subgraph of a graph with another subgraph cut off from another graph. However, previous model-agnostic augmentation algorithms changes nodes or edges that are randomly selected. On the other hand, our proposed TAG changes nodes based on the degree centrality, to keep crucial information of graphs.

Chapter 3

Proposed Method

We propose TAG, a two-staged contrastive curriculum learning framework for graphs. The main challenges and our approaches are as follows:

1. **How can we generate graph representations in both unsupervised and supervised settings?** We propose a two-staged graph contrastive curriculum learning method that is applied to both settings through two types of loss functions.

2. **How can we design augmentations for contrastive learning to preserve the semantics well?** We propose six data augmentation algorithms for graph contrastive learning. The augmentation algorithms consider degree centrality to minimize information loss.

3. **How can we determine the order of feeding the negative examples in contrastive learning?** We exploit curriculum learning to determine the order of negative samples and maximize the performance of the model.

The overall process of TAG is illustrated in Figure 2. Given a graph dataset, we first augment graphs, and then perform contrastive curriculum learning in two levels: nodes and graphs.

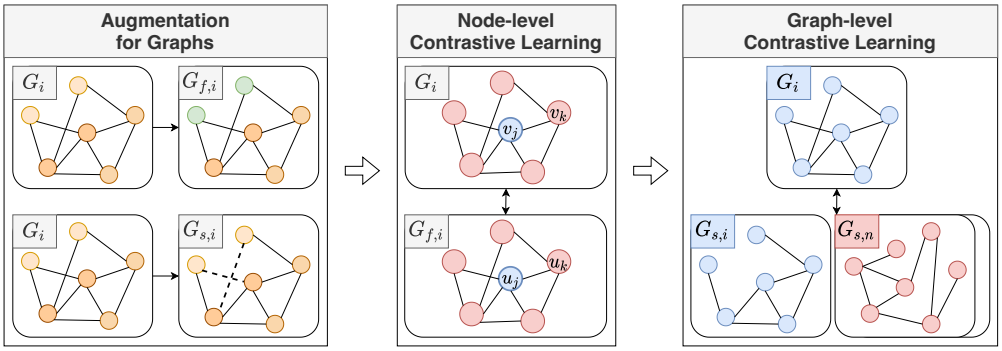

Figure 2: Overview of the proposed method. TAG performs node-level and graph-level contrastive learning on the feature-augmented graph $G_{f,i}$ and the structure-augmented graph $G_{s,i}$ obtained from the original graph G_i. In the contrastive learning steps, nodes and graphs colored with blue are positive samples, and those colored with red are negative ones.

3.1 Data Augmentation

Data augmentation is used to ensure the similarity between samples in contrastive learning. The most important challenge of augmentation is preserving the semantics, or keeping crucial information in determining graph labels. If augmentation cannot preserve the semantics well, the original graph and the augmented graph have different labels which increases the dissimilarity. Therefore, we propose six model-agnostic graph augmentation algorithms based on degree centrality to minimize information loss. Our idea is to change low-degree nodes to minimize the loss of semantics.

We categorize the six augmentation methods into two types: feature and structure modification. Feature modification algorithms generate new graphs by changing only the node feature. On the other hand, structure modification algorithms change the graph structure. We propose three algorithms for each type. The three algorithms designed for feature augmentation are listed as follows:

1. **Edit feature.** Randomly change the features of nodes with low degrees.

2. **Mix feature.** Mix the features of two selected nodes and then substitute the mixed features for the features of nodes with lower degrees.

3. **Add noise.** Add noise to the features of selected nodes. The nodes with low degrees are selected to be modified.

The algorithms for structure augmentation are as follows:

1. **Delete node.** Discard nodes with low degrees along with their connections.

2. **Delete edge.** Remove edges that are chosen from nodes with low degrees.

3. **Cut subgraph.** Select a subgraph with high degrees.

All algorithms consider degree centrality to keep semantic information. The number of nodes or edges to be augmented is decided through the augmentation ratio which is given as a hyperparameter. The augmentation is separately conducted with the contrastive learning process to improve the computational efficiency.

3.2 Two-staged Contrastive Learning

Graph contrastive learning is a self-supervised approach that allows a model to learn the representations of graphs without labels by teaching the model which graph instances are similar or different. We use the data augmentation algorithms proposed in Section 3.1 to generate similar graphs. TAG is composed of two stages, node-level and graph-level contrastive learning.

3.2.1 Node-level Contrastive Learning

The node-level contrastive learning in TAG embeds the nodes into a latent space where positive pairs of nodes are more closely located than negative ones.

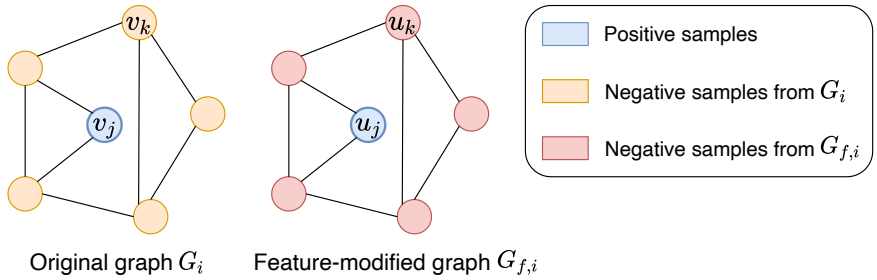

Figure 3: Positive and negative samples of the node-level contrastive learning. Nodes v_j and v_k are selected from the original graph G_i while nodes u_j and u_k are sampled from a feature-augmented graph $G_{f,i}$ at the same position, respectively.

Positive pairs (v_j, u_j) of nodes are obtained by selecting a node v_j from an original graph G_i, and a node u_j from a feature-augmented graph $G_{f,i}$ with the same position.

There are two types of negative node pairs: 1) pairs (v_j, v_k) of nodes both sampled from the original graph G_i, and 2) pairs (v_j, u_k) of nodes sampled from G_i and $G_{f,i}$, respectively. All nodes in G_i which are not selected for the positive pairs are used to generate the negative samples v_k. Similarly, every node u_k from $G_{f,i}$ except for the selected positive node u_j is treated as a negative sample. The process of sampling positive and negative pairs of nodes for the node-level contrastive learning is illustrated in Figure 3.

The node-level contrastive loss l_{n} is defined as follows:

$$l_{\mathrm{n}} = \sum_{j=1}^{K} \log \frac{\exp(\mathrm{sim}(\mathbf{v}_j, \mathbf{u}_j)/\tau)}{\sum_{k \neq j}^{K} \exp(\mathrm{sim}(\mathbf{v}_j, \mathbf{v}_k)/\tau) + \sum_{k \neq j}^{K} \exp(\mathrm{sim}(\mathbf{v}_j, \mathbf{u}_k)/\tau)} \tag{3.1}$$

where $\mathrm{sim}(\cdot)$ denotes the cosine similarity function, τ is the temperature parameter, and K is the number of nodes in a graph. Vectors $\mathbf{v}_j$ and $\mathbf{u}_j$ are the hidden representations of nodes v_j and u_j, respectively.

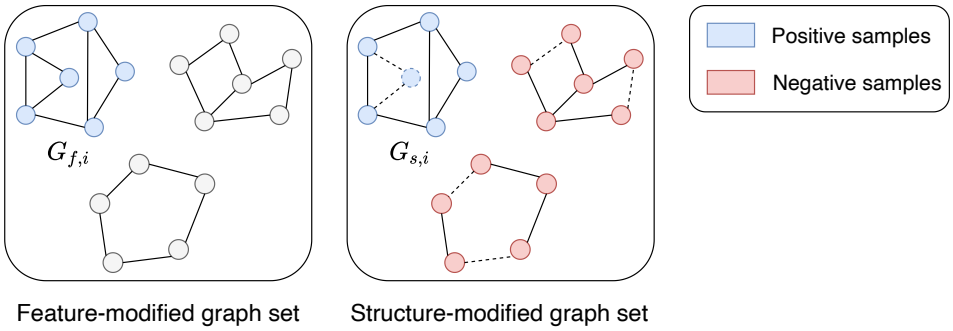

Figure 4: Illustration of positive and negative samples for graph-level contrastive learning. $(G_{f,i}, G_{s,i})$ is a positive pair originated from a graph G_i, and $(G_{f,i}, G_{s,i'})$ for $i \neq i'$ are negative pairs.

3.2.2 Graph-level Contrastive Learning

Graph representations are learned based on node embeddings. As with node-level contrastive learning, positive and negative samples are defined using augmentation in graph-level contrastive learning.

A positive pair $(G_{f,i}, G_{s,i})$ of graphs contains a feature-modified graph $G_{f,i}$ and a structure-modified graph $G_{s,i}$ of a graph G_i. Negative pairs are $(G_{f,i}, G_{s,i'})$ where $G_{i'}$ is a different graph from G_i. Figure 4 explains positive and negative samples designed for graph-level contrastive learning.

The graph-level contrastive loss l_{g} is written as below:

$$l_{\mathrm{g}} = \log \frac{\exp(\mathrm{sim}(\mathbf{z}_{f,i}, \mathbf{z}_{s,i})/\tau)}{\sum_{n \neq i}^{N} \exp(\mathrm{sim}(\mathbf{z}_{f,i}, \mathbf{z}_{s,n})/\tau)} \tag{3.2}$$

where $\mathbf{z}_{\cdot,i}$ is a representation of graph $G_{\cdot,i}$ and N is the number of graphs for training.

The final loss function $\mathcal{L}$ for TAG jointly uses the node-level and graph-level

contrastive losses. Given a set $\mathcal{D}$ of graphs for training,

$$\mathcal{L} = -\frac{1}{|\mathcal{D}|}\sum_{i\in\mathcal{D}}(l_{\mathrm{n}}(i) + l_{\mathrm{g}}(i)) \tag{3.3}$$

where $l_{\mathrm{n}}(i)$ and $l_{\mathrm{g}}(i)$ are node- and graph-level losses for a graph G_i, respectively.

3.2.3 Supervised Contrastive Learning

TAG supports supervised classification by using the cross-entropy loss $l_{\mathrm{ce}}(\cdot)$. Specifically, $l_{\mathrm{ce}}(\mathbf{y}_i, \hat{\mathbf{y}}_i)$ between the one-hot encoded label $\mathbf{y}_i$ and the prediction probability $\hat{\mathbf{y}}_i$ of a graph G_i is computed as follows:

$$l_{\mathrm{ce}}(\mathbf{y}_i, \hat{\mathbf{y}}_i) = \sum_{c=1}^{C} y_i(c) \log \hat{y}_i(c) \tag{3.4}$$

where C is the number of classes, $y_i(c)$ is c-th element of $\mathbf{y}_i$, and $\hat{y}_i(c)$ is the prediction probability of a graph G_i to class c. The probability vector $\hat{\mathbf{y}}_i$ is obtained through the softmax function after a fully-connected layer which is attached to the final layer of the graph neural network.

The final loss $\mathcal{L}_{\mathrm{sup}}$ for supervised learning is computed by adding the cross-entropy loss to the original loss in Equation (3.3):

$$\mathcal{L}_{\mathrm{sup}} = -\frac{1}{N}\sum_{i=1}^{N}(l_{\mathrm{n}}(i) + l_{\mathrm{g}}(i) + l_{\mathrm{ce}}(\mathbf{y}_i, \hat{\mathbf{y}}_i)) \tag{3.5}$$

where $l_{\mathrm{n}}(i)$ and $l_{\mathrm{g}}(i)$ are node- and graph-level losses for a graph G_i, respectively. N denotes the size of a set $\mathcal{D}$.

3.3 Curriculum Learning

We exploit a curriculum learning strategy to enhance the performance of the proposed method. Curriculum learning imitates the learning process of humans who starts learning from easier samples, and then learns more from harder samples. The difficulty of data is measured by a scoring function. We set the similarity function of contrastive loss as a scoring function. As the similarity of a positive and a negative samples increases, the score for hardness increases. Thus, we feed easier negative samples first, and then move on to harder negative samples as training continues to facilitate effective training.

Chapter 4

Experiments

We perform experiments to answer the following questions:

Q1. **Performance on Unsupervised Classification (Section 4.2).** How fast and accurate is TAG compared to previous methods for unsupervised graph classification?

Q2. **Performance on Supervised Classification (Section 4.3).** Does TAG show superior performance than other baselines in supervised graph classification task?

Q3. **Effectiveness of Proposed Augmentations (Section 4.4).** Do the proposed augmentation algorithms improve the performance of TAG?

Q4. **Ablation Study (Section 4.5).** Do our ideas, such as two-staged structure, curriculum learning, or proposed augmentations improve the accuracy of graph classification?

4.1 Experimental Settings

We introduce our experimental settings including datasets, competitors, and hyperparameters. All of our experiments are done on a single GPU machine with GeForce GTX 1080 Ti.

Table 2: Summarization of datasets.

Dataset	Graphs	Nodes	Edges	Features	Classes
MUTAG[1]	188	3,371	3,721	7	2
PROTEINS[1]	1,113	43,471	81,044	3	2
NCI1[1]	4,110	122,747	132,753	37	2
NCI109[1]	4,127	122,494	132,604	38	2
DD[1]	1,178	334,925	843,046	89	2
PTC-MR[1]	344	4,915	5,054	18	2
DBLP[1]	19,456	203,954	764,512	41,325	2

[1] https://chrsmrrs.github.io/datasets/

4.1.1 Datasets

We use seven benchmark datasets for graph classification task in our experiments, which are summarized in Table 2. MUTAG, PROTEINS, NCI1, NCI109, DD, and PTC-MR [39] are molecular datasets where the nodes stand for atoms and are labeled by the atom type, while edges are bonds between the atoms. DBLP [40] is a citation network dataset in the computer science field whose nodes represent scientific publications.

4.1.2 Competitors

We compare TAG in supervised and unsupervised settings. For the unsupervised setting, we compare TAG with ten previous approaches for unsupervised graph classification, including those for contrastive learning.

- **DGK [41]** learns latent representations of graphs by adopting the concept of the skip-gram model.
- **sub2vec [42]** is an unsupervised learning algorithm that captures two properties of subgraphs: neighborhood and structure.
- **graph2vec [43]** extends neural networks for document embedding to the graph domain, by viewing the graphs as documents.

- **InfoGraph [9]** generates graph representations by maximizing mutual information between graph-level and patch-level representations.
- **MVGRL [17]** learns graph representations by contrasting two diffusion matrices transformed from the adjacency matrix.
- **GraphCL [19]** brings image contrastive learning to graphs.
- **JOAO [20]** jointly optimizes augmentation selection together with the contrastive objectives.
- **AD-GCL [21]** uses an adversarial training strategy for edge-dropping augmentation of graphs.
- **CuCo [18]** adopts curriculum learning to graph contrastive learning for performance improvement.
- **AutoGCL [8]** uses the node representations to predict the probability of selecting a certain augment operation.

In the supervised setting, we compare the accuracy of TAG with four baselines:

- **GCN+GMP [44]** uses the graph convolutional network (GCN) to learn the node representations, and the global mean pooling (GMP) is applied to obtain the graph representation.
- **GIN [11]** uses multi-layer perceptrons (MLP) to update node representations, and sums them up to generate the graph representation.
- **ASAP [12]** alternatively clusters nodes in a graph and gathers the representations of clusters to obtain graph representations.
- **GMT [13]** designs graph pooling layer based on multi-head attention.

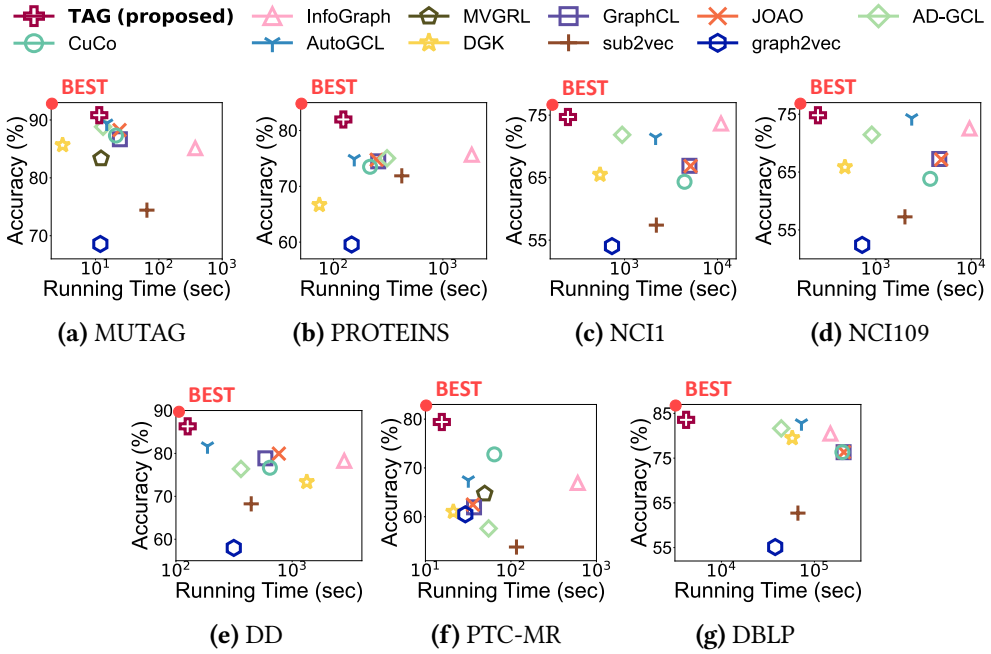

Figure 5: Overall performance of TAG with unsupervised graph classification methods. Note that TAG shows the highest classification accuracy with the shortest running time in most cases.

4.1.3 Hyperparameters

We use GCN [44] to learn node embeddings and apply the global mean pooling algorithm to generate a graph embedding. The augmentation ratio is set to 0.4 which decides the amount of data to be changed. We train each model using the Adam optimizer with a learning rate of 0.0001. We set the number of epochs to 5.

4.2 Performance on Unsupervised Classification

We evaluate TAG by measuring unsupervised graph classification accuracy and running time. We compare the graph classification accuracy of TAG with previous un-

Table 3: Accuracy of graph classification in unsupervised setting. Bold and underlined text denote the best and the second-best accuracy, respectively. OOM and Avg. denote the out of memory error and average accuracy, respectively. Note that TAG shows the best classification accuracy.

Method	Unsupervised Setting (SVM)								Unsupervised Setting (MLP)							
	MUT.	PROT.	NCI1	N109	DD	PTC	DBLP	**Avg.**	MUT.	PROT.	NCI1	N109	DD	PTC	DBLP	**Avg.**
DGK [41]	85.67	66.67	65.47	65.86	73.35	61.03	79.49	71.08	66.46	67.40	54.18	53.02	62.31	49.10	75.25	61.10
sub2vec [42]	74.42	71.88	57.40	57.26	68.25	53.80	62.71	63.67	55.26	53.91	51.75	51.56	47.95	51.20	60.37	53.15
graph2vec [43]	68.57	59.57	54.04	52.41	57.98	60.50	55.12	58.31	64.94	56.79	54.36	52.17	55.34	55.84	54.69	56.30
InfoGraph [9]	85.12	75.65	73.65	72.52	78.27	66.92	80.48	76.09	71.78	69.28	60.27	**60.20**	73.07	57.61	76.49	66.96
MVGRL [17]	83.33	OOM	OOM	OOM	OOM	64.71	OOM	21.15	89.47	OOM	OOM	OOM	OOM	58.82	OOM	21.19
GraphCL [19]	86.67	74.48	66.89	67.26	78.86	61.93	76.33	73.20	75.06	69.37	60.10	59.49	71.73	60.50	75.28	67.36
JOAO [20]	88.25	74.76	66.84	67.17	79.96	62.50	76.34	73.69	75.53	69.28	60.17	59.71	73.18	63.19	75.21	68.04
AD-GCL [21]	88.82	75.03	71.85	71.46	76.39	57.61	81.65	74.69	81.76	62.14	58.09	59.56	61.08	55.97	77.51	65.16
CuCo [18]	87.31	73.50	64.33	63.83	76.66	72.78	76.34	73.54	71.23	66.59	60.46	58.89	73.52	57.25	73.46	65.91
AutoGCL [8]	89.42	74.93	71.43	74.34	81.69	67.50	82.82	77.45	84.21	70.54	60.56	59.98	72.03	68.03	**79.06**	70.63
TAG	**90.83**	**81.98**	**74.70**	**74.82**	**86.32**	**79.41**	**83.56**	**81.66**	**93.10**	**71.17**	**60.58**	60.16	**73.91**	**77.14**	78.00	**73.44**

Table 4: Accuracy of graph classification in supervised setting. Bold and underlined text denote the best and the second-best accuracy, respectively. Avg. denotes the average accuracy. Note that TAG shows the best accuracy.

Supervised Setting								
Method	**MUT.**	**PROT.**	**NCI1**	**N109**	**DD**	**PTC**	**DBLP**	**Avg.**
GCN+GMP [44]	82.35	73.56	63.21	63.19	62.00	75.35	79.88	71.36
GIN [11]	90.49	75.77	73.94	70.00	77.78	73.77	77.31	77.01
ASAP [12]	86.52	77.56	74.19	74.51	82.66	71.83	90.98	79.75
GMT [13]	89.07	79.73	73.67	73.93	84.95	75.74	91.82	81.27
TAG	**95.45**	**87.68**	**76.19**	**75.89**	**92.40**	**80.86**	**93.73**	**86.03**

supervised methods in Table 3. We adopt support vector machine (SVM) and multilayer perceptron (MLP) as base classifiers for TAG and the baselines. Note that TAG achieves the best accuracy, giving 4.21% points and 2.81% points higher accuracy than the second-best competitors on average with SVM and MLP classifiers, respectively.

The overall performance of TAG including the running time is summarized in Figure 5. Note that TAG shows the highest classification accuracy in most cases with the shortest running time.

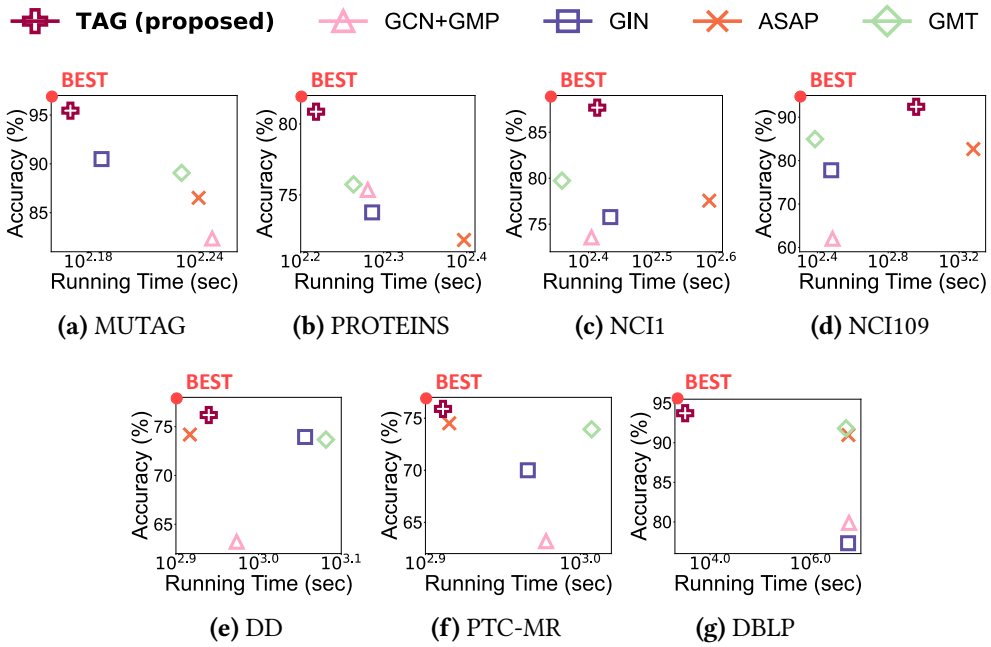

Figure 6: Overall performance of TAG with supervised graph classification methods. Note that TAG shows the highest classification accuracy with the shortest running time for most datasets.

4.3 Performance on Supervised Classification

We compare the proposed method with four baselines designed for graph classification in supervised settings. We use classification accuracy and running time as evaluation metrics. Table 4 shows the classification accuracy of TAG along with the competitors. TAG gives the highest accuracy, with 4.76% points higher average accuracy than the second-best method. Note that TAG in the supervised setting achieves 4.37% points and 12.59% points higher average accuracy than that in the unsupervised setting with SVM and MLP classifiers, respectively.

Figure 6 shows the running time of TAG and baselines in supervised setting. Note that TAG presents the shortest running time with the highest accuracy in most

Table 5: Comparison of the proposed augmentation by TAG with previous approaches. We report the best and the second-best accuracy as bold and underline, respectively. Avg. denotes the average accuracy. Note that TAG presents the best accuracy among the models.

Method	MUT.	PROT.	NCI1	N109	DD	PTC	DBLP	Avg.
CuCo + DropNode [19]	87.31	73.50	64.36	63.80	76.75	60.52	76.33	71.80
CuCo + DropEdge [34]	88.86	72.61	63.72	63.41	77.50	64.81	76.33	72.46
CuCo + GraphCrop [35]	88.28	72.96	63.24	63.32	77.17	63.37	71.59	71.42
CuCo + ChangeAttr [19]	86.23	73.68	63.60	63.75	76.66	61.05	69.87	70.69
CuCo + NodeAug [36]	82.46	74.24	64.48	63.93	79.70	78.33	78.77	74.56
CuCo + Motif-Similarity [37]	<u>90.00</u>	70.68	<u>66.64</u>	63.92	78.79	77.50	77.19	74.96
CuCo + NodeSam [38]	89.11	76.74	64.23	64.69	<u>82.05</u>	<u>79.17</u>	<u>78.98</u>	76.42
CuCo + SubMix [38]	89.04	<u>76.97</u>	64.48	<u>67.99</u>	78.94	<u>79.17</u>	78.77	<u>76.48</u>
TAG	**90.83**	**81.98**	**74.70**	**74.82**	**86.32**	**79.41**	**83.56**	**81.66**

of the cases. This shows that TAG learns meaningful graph representations not only for unsupervised graph classification, but also supervised one.

4.4 Effectiveness of Proposed Augmentations

We compare the proposed augmentations of TAG with eight previous model-agnostic augmentation algorithms for graphs. ChangeAttr modifies features and the other methods change the structure of graphs. Recall that TAG performs graph contrastive learning in two levels: node-level and graph-level. For node-level, TAG needs feature-augmented graphs. For graph-level, TAG needs feature and structure augmentations. Thus, both augmentation algorithms are necessary for TAG. MVGRL [17], GraphCL [19], and CuCo [18] are previous methods that adopt model-agnostic graph augmentations. However, MVGRL causes out-of-memory errors for large-scale graph datasets. CuCo is more elaborate than GraphCL since it additionally performs curriculum learning. Therefore, we compare TAG with previous augmentation algorithms by applying them to CuCo.

Table 6: Ablation study for TAG. We report accuracies of graph classification using SVM and MLP classifiers. Bold, underlined, and Avg. text denote the best, the second-best, and the average accuracy, respectively. The methods w/o curriculum and w/o node-level refer to the proposed method without curriculum learning and node-level contrastive learning, respectively. The other methods operate TAG by fixing the augmentations. TAG shows the best performance for all cases.

Method	Unsupervised Setting (SVM)								Unsupervised Setting (MLP)							
	MUT.	PROT.	NCI1	N109	DD	PTC	DBLP	**Avg.**	MUT.	PROT.	NCI1	N109	DD	PTC	DBLP	**Avg.**
w/o curriculum	90.07	78.14	65.99	64.66	80.36	70.08	78.06	75.33	91.18	67.58	54.88	55.15	69.73	70.77	76.68	69.42
w/o node-level	90.40	77.07	65.20	65.25	80.14	69.32	77.48	74.98	86.09	66.80	54.36	55.06	70.34	70.94	76.71	68.61
Edit feature + Delete node	90.70	80.75	67.25	65.06	81.09	79.29	80.54	77.81	92.70	69.90	58.08	56.91	72.27	75.10	77.39	71.76
Edit feature + Delete edge	90.54	81.11	66.61	64.41	80.25	79.04	83.14	77.87	87.55	69.96	59.55	58.78	68.60	76.58	76.99	71.14
Edit feature + Cut subgraph	90.82	80.85	66.13	63.54	80.08	66.30	78.58	75.19	92.47	70.95	55.16	56.58	67.63	74.94	77.33	70.72
Mix feature + Delete node	90.76	79.93	74.61	72.91	86.00	71.93	82.71	79.83	92.94	70.35	60.33	59.20	65.98	74.47	72.96	70.89
Mix feature + Delete edge	90.65	81.56	73.05	73.91	85.52	77.77	83.48	80.85	88.30	70.01	59.32	57.31	70.71	70.26	73.17	69.87
Mix feature + Cut subgraph	90.77	78.95	74.55	74.76	82.54	78.56	83.03	80.45	**93.10**	69.45	60.10	56.19	69.40	77.12	77.88	71.89
Add noise + Delete node	**90.83**	80.63	67.06	64.35	82.11	71.59	78.45	76.43	91.37	68.36	56.94	58.10	68.70	72.89	76.91	70.47
Add noise + Delete edge	90.36	81.05	65.44	64.63	80.75	71.90	83.30	76.78	91.62	69.49	58.86	55.92	70.76	76.81	77.30	71.54
Add noise + Cut subgraph	90.75	81.30	66.62	65.84	81.11	71.20	79.04	76.55	90.01	70.73	58.01	58.13	69.24	75.58	77.67	71.34
TAG	**90.83**	**81.98**	**74.70**	**74.82**	**86.32**	**79.41**	**83.56**	**81.66**	**93.10**	**71.17**	**60.58**	**60.16**	**73.91**	**77.14**	**78.00**	**73.44**

Table 5 shows the classification results using different augmentations. TAG outperforms the baselines in all cases. Specifically, TAG achieves 5.18% points higher average accuracy than the strongest baseline SubMix. Note that random-based augmentations DropNode, DropEdge, GraphCrop, and ChangeAttr degrade the performance of CuCo for all datasets. This proves that random-based augmentation methods have difficulty preserving the semantics. In contrast, TAG with the proposed augmentations help enhance the performance.

4.5 Ablation Study

We perform an ablation study for TAG and report the result in Table 6. The methods w/o curriculum and w/o node-level are the proposed method without curriculum learning and without two-staged structure which performs only graph-level contrastive learning, respectively. We also run TAG by fixing the proposed augmentations. For example, the Edit feature + Delete node method operates TAG using edit

feature and delete node algorithms for feature and structure modification, respectively. TAG with the curriculum learning improves the classification performance of SVM and MLP by 6.33% and 4.02% points on average, respectively, compared to that without the curriculum learning. Operating TAG using both node-level and graph-level contrastive learning achieves 6.68% and 4.83% points higher average accuracy than performing TAG only using graph-level contrastive learning with SVM and MLP classifiers, respectively. Furthermore, TAG achieves the best performance when it utilizes all of the proposed augmentation algorithms. The results show that the proposed ideas, i.e., the two-staged framework, exploitation of curriculum learning, and the proposed augmentation algorithms for contrastive learning improve the accuracy of graph classification.

Chapter 5

Conclusion

We propose TAG, a two-staged contrastive curriculum learning model for graphs. We introduce two types of data augmentations for graphs and propose six model-agnostic augmentation algorithms that minimize information loss. The proposed method conducts contrastive curriculum learning in two stages. In the first stage, TAG gathers the relational information between nodes from an original graph and a feature-modified graph. In the second stage, the proposed method utilizes both feature-modified and structure-modified graphs to learn the similarity between them. We exploit curriculum learning to effectively train the model via carefully selected ordering of feeding negative samples. We evaluate TAG by measuring the graph classification accuracy and running time. TAG shows the fastest running time and the best accuracy achieving up to 4.21% points and 4.76% points higher average accuracy than the second-best competitors in unsupervised and supervised settings, respectively. Future works include designing an accurate graph classification method for hypergraphs.

References

[1] J. Jung, W. Jin, and U. Kang, "Random walk-based ranking in signed social networks: model and algorithms," *Knowl. Inf. Syst.*, 2020.

[2] W. Jin, J. Jung, and U. Kang, "Supervised and extended restart in random walks for ranking and link prediction in networks," *CoRR*, vol. abs/1710.06609, 2017.

[3] J. Yoo, H. Jeon, J. Jung, and U. Kang, "Accurate node feature estimation with structured variational graph autoencoder," in *KDD*, 2022.

[4] J. Jung, J. Yoo, and U. Kang, "Signed random walk diffusion for effective representation learning in signed graphs," *Plos one*, 2022.

[5] J. Yoo, J. Kim, H. Yoon, G. Kim, C. Jang, and U. Kang, "Accurate graph-based PU learning without class prior," in *ICDM*, 2021.

[6] J. Kim, J. Jung, and U. Kang, "Compressing deep graph convolution network with multi-staged knowledge distillation," *Plos one*, 2021.

[7] J. Yoo, H. Jeon, and U. Kang, "Belief propagation network for hard inductive semi-supervised learning," in *IJCAI*, 2019.

[8] Y. Yin, Q. Wang, S. Huang, H. Xiong, and X. Zhang, "Autogcl: Automated graph contrastive learning via learnable view generators," in *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022*.

[9] F. Sun, J. Hoffmann, V. Verma, and J. Tang, "Infograph: Unsupervised and semi-supervised graph-level representation learning via mutual information maximization," in *8th International Conference on Learning Representations, ICLR 2020*.

[10] P. Velickovic, W. Fedus, W. L. Hamilton, P. Liò, Y. Bengio, and R. D. Hjelm, "Deep graph infomax," in *7th International Conference on Learning Representations, ICLR 2019*.

[11] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How powerful are graph neural networks?," in *7th International Conference on Learning Representations, ICLR 2019*.

[12] E. Ranjan, S. Sanyal, and P. P. Talukdar, "ASAP: adaptive structure aware pooling for learning hierarchical graph representations," in *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020*.

[13] J. Baek, M. Kang, and S. J. Hwang, "Accurate learning of graph representations with graph multiset pooling," in *9th International Conference on Learning Representations, ICLR 2021*.

[14] Z. Tan, K. Ding, R. Guo, and H. Liu, "Supervised graph contrastive learning for few-shot node classification," 2022.

[15] H. Jia, J. Ji, and M. Lei, "Supervised contrastive learning with structure inference for graph classification," *CoRR*, vol. abs/2203.07691, 2022.

[16] Y. Wang, J. Zhang, H. Li, Y. Dong, H. Yin, C. Li, and H. Chen, "Clusterscl: Cluster-aware supervised contrastive learning on graphs," in *WWW*, 2022.

[17] K. Hassani and A. H. K. Ahmadi, "Contrastive multi-view representation learning on graphs," in *Proceedings of the 37th International Conference on Machine Learning, ICML 2020*.

[18] G. Chu, X. Wang, C. Shi, and X. Jiang, "Cuco: Graph representation with curriculum contrastive learning," in *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI 2021*.

[19] Y. You, T. Chen, Y. Sui, T. Chen, Z. Wang, and Y. Shen, "Graph contrastive learning with augmentations," in *Annual Conference on Neural Information Processing Systems, NeurIPS 2020*.

[20] Y. You, T. Chen, Y. Shen, and Z. Wang, "Graph contrastive learning automated," in *Proceedings of the 38th International Conference on Machine Learning, ICML 2021*.

[21] S. Suresh, P. Li, C. Hao, and J. Neville, "Adversarial graph augmentation to improve graph contrastive learning," in *Annual Conference on Neural Information Processing Systems, NeurIPS 2021*.

[22] S. Akkas and A. Azad, "JGCL: joint self-supervised and supervised graph contrastive learning," in *Companion of The Web Conference 2022*.

[23] Z. Peng, W. Huang, M. Luo, Q. Zheng, Y. Rong, T. Xu, and J. Huang, "Graph representation learning via graphical mutual information maximization," in *WWW '20: The Web Conference 2020*.

[24] 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 *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*.

[25] Y. Zhu, Y. Xu, F. Yu, Q. Liu, S. Wu, and L. Wang, "Deep graph contrastive representation learning," *CoRR*, vol. abs/2006.04131, 2020.

[26] Y. Zhu, Y. Xu, F. Yu, Q. Liu, S. Wu, and L. Wang, "Graph contrastive learning with adaptive augmentation," in *WWW '21: The Web Conference 2021*.

[27] S. Thakoor, C. Tallec, M. G. Azar, M. Azabou, E. L. Dyer, R. Munos, P. Veličković, and M. Valko, "Large-scale representation learning on graphs via bootstrapping," in *ICLR*, 2022.

[28] P. Bielak, T. Kajdanowicz, and N. V. Chawla, "Graph barlow twins: A self-supervised representation learning framework for graphs," *Knowl. Based Syst.*, 2022.

[29] D. Chen, Y. Lin, L. Li, X. Ren, P. Li, J. Zhou, and X. Sun, "Rethinking the promotion brought by contrastive learning to semi-supervised node classification," in *IJCAI*.

[30] R. Miao, Y. Yang, Y. Ma, X. Juan, H. Xue, J. Tang, Y. Wang, and X. Wang, "Negative samples selecting strategy for graph contrastive learning," *Inf. Sci.*

[31] Y. Zhu, Y. Xu, H. Cui, C. Yang, Q. Liu, and S. Wu, "Structure-enhanced heterogeneous graph contrastive learning," in *SDM*, 2022.

[32] Z. Chen, Y. Peng, S. Yu, C. Cao, and F. Xia, "Subgraph adaptive structure-aware graph contrastive learning," *Mathematics*, 2022.

[33] L. Yu, S. Pei, L. Ding, J. Zhou, L. Li, C. Zhang, and X. Zhang, "SAIL: self-augmented graph contrastive learning," in *AAAI*, 2022.

[34] Y. Rong, W. Huang, T. Xu, and J. Huang, "Dropedge: Towards deep graph convolutional networks on node classification," in *8th International Conference on Learning Representations, ICLR 2020*.

[35] Y. Wang, W. Wang, Y. Liang, Y. Cai, and B. Hooi, "Graphcrop: Subgraph cropping for graph classification," *CoRR*, vol. abs/2009.10564, 2020.

[36] Y. Wang, W. Wang, Y. Liang, Y. Cai, J. Liu, and B. Hooi, "Nodeaug: Semi-supervised node classification with data augmentation," in *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*.

[37] J. Zhou, J. Shen, and Q. Xuan, "Data augmentation for graph classification," in *CIKM '20: The 29th ACM International Conference on Information and Knowledge Management*.

[38] J. Yoo, S. Shim, and U. Kang, "Model-agnostic augmentation for accurate graph classification," in *WWW '22: The ACM Web Conference 2022*.

[39] C. Morris, N. M. Kriege, F. Bause, K. Kersting, P. Mutzel, and M. Neumann, "Tu-dataset: A collection of benchmark datasets for learning with graphs," *CoRR*, vol. abs/2007.08663, 2020.

[40] S. Pan, X. Zhu, C. Zhang, and P. S. Yu, "Graph stream classification using labeled and unlabeled graphs," in *29th IEEE International Conference on Data Engineering, ICDE 2013*.

[41] P. Yanardag and S. V. N. Vishwanathan, "Deep graph kernels," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2015*.

[42] B. Adhikari, Y. Zhang, N. Ramakrishnan, and B. A. Prakash, "Sub2vec: Feature learning for subgraphs," in *Advances in Knowledge Discovery and Data Mining - 22nd Pacific-Asia Conference, PAKDD 2018*.

[43] A. Narayanan, M. Chandramohan, R. Venkatesan, L. Chen, Y. Liu, and S. Jaiswal, "graph2vec: Learning distributed representations of graphs," *CoRR*, vol. abs/1707.05005, 2017.

[44] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *5th International Conference on Learning Representations, ICLR 2017*.

요 약

그래프 데이터셋이 주어졌을 때, 어떻게 하면 그래프 분류 성능을 최대화하는 의미 있는 그래프 표현을 학습할 수 있을까? 표현력 있는 그래프 임베딩을 학습하는 것은 다양한 실세계 그래프 기반 문제들을 해결하는데 중요하다. 그래프 대조 학습은 그래프 사이의 관계를 분석하여 레이블이 없는 그래프들의 표현을 학습하는 방법이다. 최근에 등장한 대조 학습 방법들은 데이터 증강 기법을 사용한다. 그러나 무작위 기반의 증강 기법을 사용하는 이전 방법들은 그래프 내의 주요 정보를 얻기 어렵다. 주요 정보를 잘 추출하기 위해 신중히 설계된 증강 기법을 사용하는 방법들의 경우, 연산적으로 비효율적이라는 단점이 있다.

본 논문에서는 그래프 분류를 위한 두 단계의 대조 학습 방법인 TAG (Two-staged contrAstive curriculum learning for Graphs)를 제안한다. TAG 는 정점의 연결 중심성을 계산함으로써 그래프 내의 주요 정보를 잘 보존하는 여섯 개의 모델 독립적인 증강 기법들을 활용한다. 증강 기법들은 두 단계, 즉 정점 단계와 그래프 단계로 이루어진 그래프 표현 학습에 사용된다. 실험을 통해 TAG 가 기존의 비지도 학습 모델들 중 가장 뛰어난 성능을 가진 모델보다 평균적으로 4.21% 더 높은 그래프 분류 성능을 달성하였으며, 기존의 지도 학습 모델들 중 가장 뛰어난 성능을 보인 모델보다 평균적으로 4.76% 더 높은 그래프 분류 성능을 기록함을 보여준다.

주요어 : 그래프 분류, 대조 학습, 커리큘럼 학습

학번 : 2021-28209

감사의 글

본 논문이 완성되기까지 많은 분들의 격려와 도움을 받았습니다. 그분들 덕분에 이 논문을 잘 마무리할 수 있었고, 저 또한 많이 성장할 수 있었습니다. 이 자리를 빌려 저에게 아낌없는 지지와 도움을 주신 모든 분들께 감사의 인사를 전합니다.

먼저 지도 교수님이신 강유 교수님께 진심으로 감사하다는 말씀을 전합니다. 여러모로 많이 부족했던 저를 이끌어주시고 연구자로 성장할 수 있도록 도와주셔서 감사드립니다. 교수님께 배울 수 있어서 영광이었고, 교수님의 가르침을 토대로 앞으로도 더 발전하는 모습 보여드릴 수 있도록 노력하겠습니다. 또한, 바쁜 일정 속에서도 귀중한 시간을 할애하여 학위 논문 심사를 맡아주시고 많은 조언을 해주신 곽노준 교수님과 문태섭 교수님께도 감사의 인사를 드립니다.

2년 동안 대학원 생활을 함께했던 서울대학교 데이터마이닝 연구실 분들께도 감사를 표합니다. 입학 당시 무지했던 제게 정신적, 학업적으로 많은 도움을 주셨던 유재민 선배님과 장준기 선배님께 감사하다는 말을 전합니다. 그리고 함께 연구하며 연구실에서 생활하는 동안 큰 힘이 되어준 재리, 가현이, 정인이, 지연 언니, 호영 오빠, Vladimir 오빠, 태훈 오빠, 전경 오빠, Saurav, 재영이, 준경이, 그리고 진기에게도 고맙다는 말을 전합니다. 그 외에 제가 언급하지 못한 모든 선배님들과 동기들, 그리고 후배들에게 감사를 표하며 모두 학위 과정을 잘 마치시기를 바랍니다.

마지막으로, 항상 제 옆을 지켜주는 친구들과 가족들에게 진심으로 고맙다는 인사를 전합니다. 언제나 저를 응원해주는 혜민이, 하영이, 지혜, 유림이, 경민이, 지우, 소연 언니에게 정말 고맙고, 앞으로도 변치 않는 우정으로 각자 분야에서 최고가 되기를 늘 응원합니다. 같은 분야에서 서로 버팀목이 되어주는 지선이, 수민이, 현서, 정민 언니, 규림이, 하리, 도윤이, 그리고 땅울림 멤버들에게도 고마움을 표합니다. 소마에서 만나서 지금까지도 큰 도움이 되어주는 우성 언니, 수빈이, 성찬 오빠, 채원 오빠, 경호 오빠, 동훈 오빠, 성준 오빠, 정필 오빠, 수형 오빠, 상준이에게도 항상 고맙다는 말을 전합니다. 제 룸메이트이자 학위 과정을 함께 끝마친 지민이에게도 고생 많았고 고마웠다고 말하고 싶습니다. 그리고 저를 사랑으로 키워주고 늘 아낌없이 베풀어주는 우리 엄마와 아빠에게 항상 많이 고맙고 사랑한다는 말을 전합니다.

미처 언급하지 못한 모든 분들께도 다시 한 번 감사의 인사를 드리며, 더 성장하는 모습 보여드릴 수 있도록 노력하겠습니다. 감사합니다.