

Edge Contrastive Learning: An Augmentation-Free Graph Contrastive Learning Model

Yujun Li¹, Hongyuan Zhang^{2,3}, Yuan Yuan^{1*}

¹School of Artificial Intelligence, Optics and Electronics (iOPEN),
Northwestern Polytechnical University, Xi'an 710072, P.R. China

²Institute of Artificial Intelligence (TeleAI), China Telecom, P. R. China

³The University of Hong Kong

yujunli361@gmail.com, hyzhang98@gmail.com, y.yuan1.ieee@gmail.com

Abstract

Graph contrastive learning (GCL) aims to learn representations from unlabeled graph data in a self-supervised manner and has developed rapidly in recent years. However, edge-level contrasts are not well explored by most existing GCL methods. Most studies in GCL only regard edges as auxiliary information while updating node features. One of the primary obstacles of edge-based GCL is the heavy computation burden. To tackle this issue, we propose a model that can efficiently learn edge features for GCL, namely **Augmentation-Free Edge Contrastive Learning (AFECL)** to achieve edge-edge contrast. AFECL depends on no augmentation consisting of two parts. Firstly, we design a novel edge feature generation method, where edge features are computed by embedding concatenation of their connected nodes. Secondly, an edge contrastive learning scheme is developed, where edges connecting the same nodes are defined as positive pairs, and other edges are defined as negative pairs. Experimental results show that compared with recent state-of-the-art GCL methods or even some supervised GNNs, AFECL achieves SOTA performance on link prediction and semi-supervised node classification of extremely scarce labels.

Code — <https://github.com/YujunLi361/AFECL>

1 Introduction

The recent success of graph neural networks has driven research in knowledge graphs (Zhu et al. 2022), recommendation (Cao et al. 2022; Li et al. 2023), e-commerce (Zhang et al. 2022a), biological systems (Rao et al. 2022), etc. A common approach to training graph neural networks is to use the supervised mode. For supervised learning, a sufficient amount of input data and label pairs are given otherwise this leads to overfitting (Feng et al. 2020). However, the supervised training process requires a large amount of labeled data and is expensive in real applications, since labels are expensive, or even long-tailed data (Yang et al. 2023).

Unsupervised and self-supervised learning (SSL) trains neural network models on unlabeled data, reducing dependence on labeled data (Cai, Jiang, and Yuan 2021; Zhang et al. 2023; Li, Zhao, and Yuan 2023; Zhang, Zhu, and Li

2024). Among SSL methods, contrastive learning (CL) has been shown to achieve comparable performance levels to its supervised counterparts across a spectrum of tasks, including Computer Vision (CV) (Chen et al. 2020) and Natural Language Processing (NLP) (Gao, Yao, and Chen 2021). With the development of CL, CL was introduced into the graph domain, combining GNN and CL to learn embedding, also dubbed graph contrastive learning (GCL) (Hassani and Khasahmadi 2020; Zhu et al. 2021; Xia et al. 2022).

Recent studies on graph contrastive learning focus on a similar contrastive learning paradigm. Specifically, graph contrastive learning aims to learn one or more encoders such that similar instances in the graph agree with each other, and dissimilar instances disagree with each other. Most existing GCL methods can be divided into two aspects: graph view generation by applying different transformations and learning towards minimizing contrastive loss based on mutual information (MI) estimators. For graph view generation, there are various graph transformation (augmentation) methods, such as node attribute masking (You et al. 2020), edge perturbation (Qiu et al. 2020; You et al. 2020), graph diffusion (Hassani and Khasahmadi 2020), π -noise augmentation (Li 2022; Zhang et al. 2024), etc. For contrastive objectives, there are contrastive losses widely used in CL, such as Donsker-Varadhan estimator (Donsker and Varadhan 1983; Belghazi et al. 2018), noise contrastive estimation (InfoNCE) (Gutmann and Hyvärinen 2010; Oord, Li, and Vinyals 2018), normalized temperature-scaled cross-entropy (NT-Xent) (Sohn 2016), to estimate and maximize the mutual information in CL computationally. However, there are some drawbacks of this paradigm.

On the one hand, MAGCL (Gong, Yang, and Shi 2023) argues augmentations that retain sufficiently complete task-relevant information cannot generate diverse enough views. Existing handcrafted graph augmented strategies may destroy the graph structure resulting in failing to maintain the task-related information intact. At the same time, using two encoders with the same neural architecture and tied parameters for the augmented graphs may hurt the diversity of the augmented views. Specifically, removing important edges can severely damage graph topology that is highly relevant to downstream tasks, resulting in poor graph embedding quality (Zhu et al. 2021). Generating augmentations in a heuristic (Zhu, Sun, and Koniusz 2021) or adversarial (You

*Corresponding author.

et al. 2021; Feng et al. 2022) manner, using two view encoders with the same architecture, may also lead to graph embedding quality low.

On the other hand, most existing GCL methods directly fetch the contrastive goal originally proposed in CL (Qiu et al. 2020; Zhu et al. 2020; Wan et al. 2021; You et al. 2021; Zhu, Sun, and Koniusz 2021), while paying no attention to the topological information and the information of edges in the graph data. They regard different augmentations of the same sample (node or graph) as positives and different augmentations of other samples as negatives and pull the positives close and the negatives far apart. However, they are usually based on the homophily assumption, which means that connected nodes should be more similar. In addition, for graphs, changes in edges can directly affect topological information. Exploiting the contrastive objective ignores topological information and contradicts the homophily assumption.

To address the above problems, we propose a new paradigm that applies edge contrastive learning without data augmentation. Unlike the existing augmentation-free method (Xiao et al. 2024) in node level, we are the first to propose and verify the validity of edge-level contrast called **Augmentation-Free Edge Contrastive Learning (AFECL)**. Specifically, our model adopts a simple graph neural network (GNN) as the encoder to generate node embeddings. Then, instead of only treating edges as auxiliary information to implement parameter updates, we directly compute the edge embeddings through node embeddings. For the contrastive objectives, AFECL fully considers network topological information and regards edges connecting the same node as positive pairs and edges that do not connect nodes as negative pairs. Overall, our main contributions are listed as follows:

- To the best of our knowledge, for the first time we study the edge-level pairs for contrast. We introduce a novel edge representation learning method generating edge embeddings through node embeddings. Moreover, we design a new edge-level contrastive loss where edges connecting the same node as positive pairs and edges that do not connect nodes as negative pairs.
- We develop a novel and effective paradigm for graph contrastive learning which provides a high degree of flexibility, high efficiency, and ease of use. Specifically, AFECL does not need additional handcrafted graph augmentations and can train on graphs at any scale by generating a small number of edge features.
- Experimental results show that, compared with recent state-of-the-art GCL methods or even some supervised GNNs, our method achieves SOTA performance on link prediction and semi-supervised node classification of extremely scarce labels.

2 Related Work

In this section, we provide a brief review of existing GCL methods. Then, the proposed method and its related work are summarized.

Method	Data Augmentation	Contrastive Objective
DGI	No Data Augmentation	Node-graph
GMI	No Data Augmentation	Node-node
MVGRL	Graph Diffusion	Node-graph
GCC	Extract Subgraph	Graph-graph
GCA	Attribute Perturbation+Edge Perturbation	Node-node
ARIEL	Adversarial Graph Perturbation	Node-node
SPGCL	No Data Augmentation	Node-node
GraphACL	No Data Augmentation	Node-node
PIGCL	Attribute Perturbation+Edge Perturbation	Node-node
AFECL	No Data Augmentation	Edge-edge

Table 1: Comparison of related works.

2.1 Data Augmentations for Contrastive Learning

The general paradigm in CL requires data augmentation of the input data first. In computer vision (CV), there are many high-quality data augmentation schemes, such as rotations, blurring, resizing, cropping, flipping, etc (Chen et al. 2020). In natural language processing (NLP), due to its discrete nature, the data augmentation scheme is not as straightforward as in CV, such as word deletion, reordering, substitution, etc (Gao, Yao, and Chen 2021). In the graph domain, there are many data augmentation methods, but there is no universal graph data augmentation scheme. For example, DGI (Veličković et al. 2019) achieves data augmentation by disrupting the nodes of the original graph. GraphCL (You et al. 2020) proposes to achieve data augmentation through attribute masking, edge perturbation, node dropout and sub-graph extraction. GRACE (Zhu et al. 2020), GCA (Zhu et al. 2021) and CCA-SSG (Zhang et al. 2021) generate views by randomly masking node attributes and randomly removing edges. MoCL (Sun et al. 2021) demonstrates that infusing domain knowledge can serve as prior knowledge, which helps to find appropriate augmentations. MVGRL (Hassani and Khasahmadi 2020) augments the input graph via graph diffusion and ARIEL (Feng et al. 2022) augments the input graph via adversarial graph perturbation. SPAN (Lin and Chen 2023) generates augmentations by maximizing the spectral change. However, there are limitations in that appropriate augmentation methods must be selected for different graph datasets to the above handcraft augmentations, resulting in poor generalizability (You et al. 2020). To avoid manually tuning dataset-specific graph augmentations, LOCAL-GCL (Zhang et al. 2022b) and GraphACL (Xiao et al. 2024) define first-order neighbors of nodes as positive samples without augmentation.

The proposed model regards the original graph as the view, which not only avoids improper modification of the graph dataset topology but also has wide generalizability.

2.2 Graph Contrastive Learning Models

GCL often explores node-node, node-graph and graph-graph relationships as contrastive objectives (Xie et al. 2022). For node-graph contrast, most GCL methods employ graph-level encoders and node-level encoders to obtain the corresponding embeddings. For example, DGI (Veličković et al. 2019) contrasts node embeddings of the original graph and node embeddings of the corrupted graph with graph embeddings. MVGRL (Hassani and Khasahmadi 2020) contrasts the embedding of a view through a node-level en-

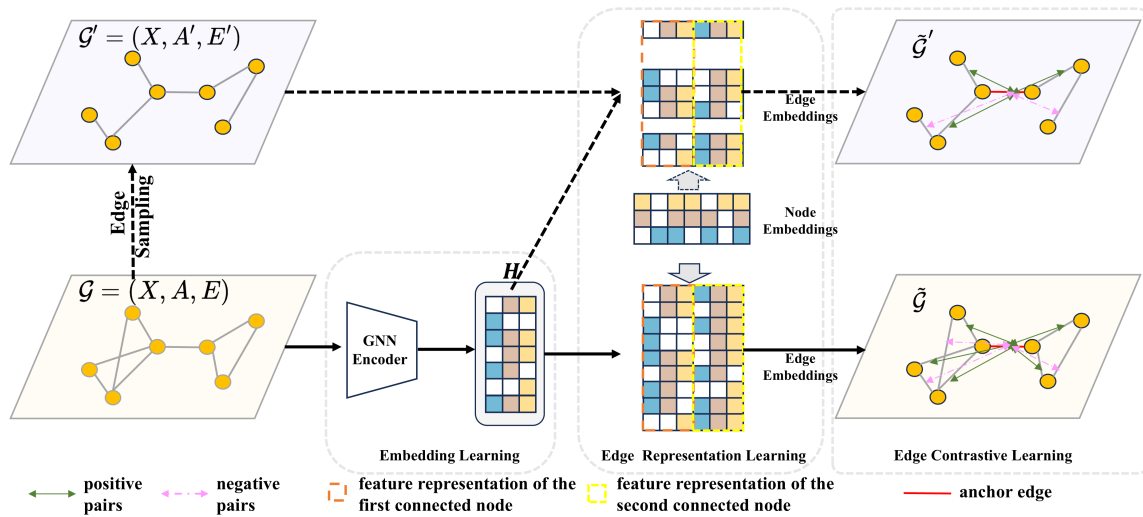


Figure 1: The architecture of the proposed AFECL framework. Firstly, the original graph is regarded as a view and fed into the encoder to learn the representation of the nodes. Then, the learned node representations are used to concatenate and generate edge representations. Note that in large graphs, edges need to be sampled first to generate representations behind the sampling. In the end, the model applies edge contrastive loss to maximize the mutual information between positive pairs of representations and minimize the mutual information between negative pairs of representations. The red edges represent both anchors and positives.

coder with the embedding of another view through a graph-level encoder. For graph-graph contrast, most GCL methods employ two graph-level encoders to obtain graph embeddings and contrast graph-level between positives and negatives. For example, GCC (Qiu et al. 2020) contrasts different subgraph embeddings obtained by different graph-level encoders. For node-node contrast, most GCL methods employ node-level encoders to obtain node embeddings and contrast node-level between positives and negatives. For example, GRACE (Zhu et al. 2020) and GCA (Zhu et al. 2021) contrast the embedding of each node, where the same node in different augmented views is regarded as positive pairs and other nodes are viewed as negative pairs. However, these methods do not consider the topological information of the graph. NCLA (Shen et al. 2023) proposes Neighbor Contrastive Loss, which treats nodes with different augmentations and their neighbor nodes as positive pairs, and other nodes as negative pairs. In addition, existing GCL methods ignore edge information and only use edges as auxiliary information to update node representations.

In our model, we propose a new edge contrastive loss for edge-edge GCL. Unlike existing contrastive objectives, our method is an edge-edge contrast, where edge representations are generated from node representations. Moreover, our method exploits network topology information to define positive and negative pairs, where positives are the edges connecting the same nodes and negatives are the others.

Comparisons with related graph CL methods. In summary, we briefly compare the proposed AFECL with other proposed state-of-the-art graph contrastive learning methods, as shown in Table 1. It can be seen that the proposed AFECL method requires only one view and does not require

the use of additional node corruption like DGI (Veličković et al. 2019). Besides, these data augmentation methods have been proven to be effective, but they suffer from problems such as complexity and possible inappropriate modification of the original graph. AFECL does not require data augmentation and only uses the original graph faded into the encoder. Most importantly, AFECL is the only model that proposes to define edge-edge as the contrastive objective.

3 Methodology

In this section, we present AFECL in detail. Then, we elaborate on how to generate edge features in Section. 3.2, how to design the edge contrastive loss and define positives and negatives between edges in Section. 3.3. We propose a novel edge contrastive architecture for unsupervised graph representation learning, as shown in Figure 1.

3.1 Preliminaries

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ denote a graph, where $\mathcal{V} = \{v_1, v_2, \dots, v_N\}$ and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ represent the node set and edge set respectively. $\mathbf{X} \in \mathbb{R}^{N \times F}$ and $\mathbf{A} \in \{0, 1\}^{N \times N}$ denote the node feature matrix and the symmetric adjacency matrix, where $\mathbf{x}_i \in \mathbb{R}^F$ is the feature vector of v_i and $A_{ij} = 1$ iff $(v_i, v_j) \in \mathcal{E}$, otherwise $A_{ij} = 0$. \mathcal{N}_i represents the first-order neighbors of node i in the graph. $\mathbf{E} \in \mathbb{R}^{M \times D}$ denotes the edge feature matrix, where $\mathbf{e}_{ij} \in \mathbb{R}^D$ is the feature vector of edge (v_i, v_j) and e_k represents the k -th edge. Given \mathbf{X} and \mathbf{A} as the input, the proposed model employs the GNN encoder $f(\mathbf{X}, \mathbf{A})$ to learn the representations of nodes $\mathbf{H} = f(\mathbf{X}, \mathbf{A}) \in \mathbb{R}^{N \times F'}$, $F' \ll F$. Then, the proposed method exploits the embeddings of nodes to gener-

ate the representations of edges. These representations are learned by optimizing the the edge contrastive loss, and can be used in downstream tasks without access to the labels.

3.2 Edge Representation Learning

In our proposed model, we employ multi-head GAT as the encoder for all benchmark datasets. Each head is viewed as a single-layer feedforward neural network. In the k -th head attention, the edge coefficient between neighbor node i and j can be expressed as

$$\alpha_{ij}^{(k)} = \frac{\exp(\text{LeakyReLU}(\mathbf{a}^{(k)}[\mathbf{W}^{(k)}\mathbf{x}_i\|\mathbf{W}^{(k)}\mathbf{x}_j]))}{\sum_{v_p \in \mathcal{N}_i \cup \{v_i\}} \exp(\text{LeakyReLU}(\mathbf{a}^{(k)}[\mathbf{W}^{(k)}\mathbf{x}_i\|\mathbf{W}^{(k)}\mathbf{x}_p]))}, \quad (1)$$

where $\mathbf{W}^{(k)} \in \mathbb{R}^{F' \times F}$ represents the k -th head learnable weight matrix which is applied to every input node feature to learn embeddings, $\mathbf{a}^{(k)}$ represents the learnable weight vector of k -th head, $\|$ is the concatenation operation, and $\text{LeakyReLU}(\cdot)$ is a nonlinear activation. Note that $\alpha_{ij}^{(k)} = 0$ if $A_{ij} = 0$.

The edge coefficients are used to compute a linear combination of the features of neighbor nodes to learn the embedding of each node. Then, the GAT encoder uses a nonlinear activation ELU to serve as the final embeddings, as

$$\mathbf{h}_i^{(k)} = \text{ELU}\left(\sum_{v_j \in \mathcal{N}_i \cup \{v_i\}} \alpha_{ij}^{(k)} \mathbf{W}^{(k)} \mathbf{x}_j\right), \quad (2)$$

where $\mathbf{h}_i^{(k)}$ is the embedding of node i in the k -th head. Then, node i embedding of each head is concatenated, as

$$\mathbf{h}_i = \|\|_{k=1}^K \mathbf{h}_i^{(k)}. \quad (3)$$

To avoid an expensive computational burden, we exploit the node representation to generate edge representation, as

$$\mathbf{h}_{ij} = g(e^{ij}) = \varphi(f(v_i), f(v_j)) = \varphi(\mathbf{h}_i, \mathbf{h}_j), \quad (4)$$

where $f(\cdot)$ is a GNN encoder, $g(\cdot)$ represents the final learned edge representation module, and $\varphi(\cdot, \cdot)$ is a mapping function of $\mathbb{R}^{KF'} \times \mathbb{R}^{KF'} \rightarrow \mathbb{R}^{D'}$, a simple and feasible implementation as

$$\varphi(f(v_i), f(v_j)) = \mathbf{W}(f(v_i)\|f(v_j)) = \varphi(\mathbf{h}_i\|\mathbf{h}_j), \quad (5)$$

where $\mathbf{W} \in \mathbb{R}^{D' \times 2KF'}$ can be an identity matrix \mathbf{I} if $D' = 2KF'$ or a learnable weight matrix.

Compared to existing edge representation methods, the proposed model has two advantages.

1) The existing edge representation learning methods usually propose to additionally create the adjacency matrix for edges, either by defining the neighborhood structure or using the line graph transformation. **However, for a nearly complete graph obtaining the adjacency of edges requires $\mathcal{O}(n^2)$ time complexity.** We propose a simple method to generate the representation of edges, by using node representation, which can directly avoid learning each edge representation.

2) Learning edge representations by creating a structure for edges may not yield good performance for downstream tasks, such as node classification. In this case, there is little correlation between the representation of nodes and the representation of edges. In contrast, our proposed method uses the representation of **two nodes connected by edges to generate the representation of edges.**

3.3 Edge Contrastive Learning

After obtaining the embeddings of edges, we find it necessary to define positives and negatives in edge-edge GCL.

Let \mathbf{h}_i and \mathbf{h}_j denote the embeddings of v_i and v_j learned by the GNN encoder respectively. Then select \mathbf{h}_{ij} as the anchor, which is generated by \mathbf{h}_i and \mathbf{h}_j . In our proposed model, positive samples consist of three sources:

- the same edge, i.e., the embedding of the same edge in the view \mathbf{h}_{ij} ;
- $\{\mathbf{h}_{ik} \mid v_k \in \mathcal{N}_i\}$, the embedding of the edges adjacent to node i ;
- $\{\mathbf{h}_{kj} \mid v_k \in \mathcal{N}_j\}$, the embedding of the edges adjacent to node j .

In this case, the number of positive pairs associated with the anchor \mathbf{h}_{ij} should be $|\mathcal{N}_i| + |\mathcal{N}_j| + 1$, where $|\mathcal{N}_i|$ is the number of neighbors of node i and $|\mathcal{N}_j|$ is the number of neighbors of node j . In the view, the edge contrastive loss associated with the anchor \mathbf{h}_{ij} is formulated as

$$\ell(\mathbf{h}_{ij}) = -\log \frac{\exp(\theta(\mathbf{h}_{ij}, \mathbf{h}_{ij})/\tau) + \sum_{v_k \in \mathcal{N}_i} (\exp(\theta(\mathbf{h}_{ij}, \mathbf{h}_{ik})/\tau)) + \sum_{v_k \in \mathcal{N}_j} (\exp(\theta(\mathbf{h}_{ij}, \mathbf{h}_{kj})/\tau))}{\exp(\theta(\mathbf{h}_{ij}, \mathbf{h}_{ij})/\tau) + \sum_{k \neq i, j} (\exp(\theta(\mathbf{h}_{ij}, \mathbf{h}_{ik})/\tau)) + \sum_{k \neq i, j} (\exp(\theta(\mathbf{h}_{ij}, \mathbf{h}_{kj})/\tau))} / (|\mathcal{N}_i| + |\mathcal{N}_j| + 1), \quad (6)$$

where τ is a temperature parameter and $\theta(\cdot)$ is the cosine similarity measure. We decompose the last two terms of the denominator of Eq. (6) as

$$\begin{aligned} \sum_{k \neq i, j} \exp(\theta(\mathbf{h}_{ij}, \mathbf{h}_{ik})/\tau) &= \sum_{v_k \in \mathcal{N}_i} \exp(\theta(\mathbf{h}_{ij}, \mathbf{h}_{ik})/\tau) + \sum_{v_k \notin \mathcal{N}_i} \exp(\theta(\mathbf{h}_{ij}, \mathbf{h}_{ik})/\tau), \\ \sum_{k \neq i, j} \exp(\theta(\mathbf{h}_{ij}, \mathbf{h}_{kj})/\tau) &= \sum_{v_k \in \mathcal{N}_j} \exp(\theta(\mathbf{h}_{ij}, \mathbf{h}_{kj})/\tau) + \sum_{v_k \notin \mathcal{N}_j} \exp(\theta(\mathbf{h}_{ij}, \mathbf{h}_{kj})/\tau), \end{aligned}$$

where the edges connected by the non-neighbor nodes of node i and the edges connected by the non-neighbor nodes of node j are regarded as negative pairs respectively. Minimizing Eq. (6) is equivalent to maximizing the agreement between positive pairs and minimizing that of negative pairs. That is, the embedding of each edge is driven to agree with itself and the embeddings of the edges connected to its neighboring nodes within a view. In the meanwhile, the embedding of each edge would disagree with the embedding of edges connected to non-neighbor nodes. Note that our proposed model is different from the existing multi-view graph contrastive learning model. In our model, positive samples include the anchor itself to prevent the learned representation from deviating from our objective. Besides, we do not need to set up a special asymmetric network structure or data augmentations to achieve contrastive learning. The final edge contrastive loss within a view, averaged over all edges, is defined as

$$\mathcal{L} = \frac{1}{M} \sum_{k=1}^M \ell(\mathbf{h}_{e_k}) = \frac{1}{M} \sum_{i=1}^N \sum_{j=1}^N \ell(\mathbf{h}_{ij}), \quad (7)$$

Algorithm 1: The pseudo-code for the proposed AFECL

Input: The feature matrix \mathbf{X} , adjacency matrix \mathbf{A} , edge feature matrix \mathbf{E} , Edge sampling probability p_s
Output: Embeddings of nodes \mathbf{H}

- 1: **for** epoch in 1 to \mathcal{T} **do**
- 2: **for** k in 1 to K **do**
- 3: Calculate the edge coefficient $\alpha_{ij}^{(k)}$ by Eq.(1).
- 4: Generate node embeddings $\mathbf{h}_i^{(k)}$ by Eq.(2).
- 5: **end for**
- 6: Compute node embedding by concatenation $\mathbf{h}_i = \parallel_{k=1}^K \mathbf{h}_i^{(k)}$.
- 7: **if** $p_s \neq 1$ **then**
- 8: Sampling the edges $\mathbf{A}' = \mathbf{A} \circ \mathbf{R}$.
- 9: Compute edge embeddings by Eqs.(4), (5).
- 10: **else**
- 11: Generate embeddings of edges by Eqs.(4), (5).
- 12: **end if**
- 13: Compute edge contrastive loss \mathcal{L} by Eq.(7).
- 14: Update parameters to minimize \mathcal{L} ;
- 15: **end for**

where $\ell(\mathbf{h}_{ij}) = 0$ if $A_{ij} = 0$.

Edge Sampling. We randomly sample a portion of the edges in the original graph. Since the number of edges is much more than the number of nodes, in large graphs we need to sample edges to achieve edge contrastive learning. Specifically, for simplicity, we first sample a random masking matrix $\mathbf{R} \in \{0, 1\}^{N \times N}$, each entry of which follows a Bernoulli distribution $R_{ij} \sim \text{Bernoulli}(p_s)$ if $A_{ij} = 1$ and $R_{ij} = 0$ otherwise. Here p_s represents the probability of each edge being sampled. In this way, the final sampled adjacency matrix can be computed as

$$\mathbf{A}' = \mathbf{A} \circ \mathbf{R}, \quad (8)$$

where \circ is the Hadamard product.

To sum up, at each training epoch, our model first passes through the GNN encoder to obtain the embeddings of the nodes. Then, we perform edge sampling on large graphs that do not require this operation on small graphs, and generate embeddings of these edges that are concatenated by connecting node embeddings. Finally, the parameters are updated by minimizing the objective in Eq. (7). The optimization process of our model is summarized in Algorithm 1.

Computational Complexity. The original graph is fed into a multi-head GAT encoder to learn the representation of nodes, where the time complexity is $\mathcal{O}((NFF' + MF')K)$. Note that N and M are the number of nodes and edges in the graph \mathcal{G} respectively, K is the number of heads, F is the number of input node features, and F' is the node embedding dimension. Edge representation learning is simply generated by concatenating the learned corresponding node embeddings, so the time complexity can be ignored. For small graph datasets, the time complexity of edge contrastive learning is $\mathcal{O}(M^2D')$. For large graph datasets, the time complexity of edge contrastive learning is $\mathcal{O}(M'^2D')$. In both cases, M or M' is slightly larger than N

and $D' = KF'$, so $\mathcal{O}(M^2D')$ or $\mathcal{O}(M'^2D')$ can be considered to be equivalent to $\mathcal{O}(N^2KF')$. Thus, the time complexity of AFECL is $\mathcal{O}((NFF' + MF')K + N^2KF')$. Since $M \ll N^2$, the overall time complexity of AFECL is $\mathcal{O}((NFF' + N^2F')K)$. So the time complexity of edge-edge AFECL is comparable to representative node-node GCL methods, e.g., GRACE (Zhu et al. 2020).

4 Experiments

4.1 Datasets

In our experiments, there are totally eight benchmark datasets of node classification, which have been widely used in previous GCL methods. In the homophilic graph datasets, three citation networks include Cora, Citeseer, and Pubmed (Sen et al. 2008), a co-author network includes Coauthor-CS (Shchur et al. 2018) and a co-purchase network includes Amazon-photo (Shchur et al. 2018). For heterophilic graphs, we adopt Actor, Chameleon, and a larger graph, Penn94. The details of these datasets can be found in Appendix A.

4.2 Baselines

We consider 9 state-of-the-art methods for comparison on semi-supervised node classification task. Baselines trained without labels: DGI (Veličković et al. 2019), MVGRL (Hasani and Khasahmadi 2020), GCA (Zhu et al. 2021), ARIEL (Feng et al. 2022), GraphACL (Xiao et al. 2024), and PiGCL (He et al. 2024). Baselines trained with labels: GCN (Kipf and Welling 2016), GAT (Veličković et al. 2017).

4.3 Experimental Settings

The proposed model was implemented using PyTorch 1.13.1 (Paszke et al. 2019) and Deep Graph Library 1.1.2 (Wang et al. 2019), and trained by the Adam optimizer on all datasets. The detailed hyperparameters are in Appendix A.

We test AFECL on both semi-classification and link prediction. In scenarios with extremely limited labels, where the number of training nodes per class c is selected from 1, 2, 3, 4, we conducted experiments following (Shen et al. 2023). Besides, we also followed (Li, Han, and Wu 2018; Li et al. 2019) to refrain from utilizing a validation set with additional labels for model selection. To better verify whether the proposed method works, we conducted experiments with relatively sufficient labels. Specifically, we randomly select 20 training nodes per class. For citation networks, we followed (Yang, Cohen, and Salakhudinov 2016), which selects 500 nodes per class for validation and the rest of the nodes for testing. For a large graph dataset Penn94, we follow (Yang and Mirzasoleiman 2024) to **verify the scalability of AFECL**. Further detailed introduction can be found in Appendix A. For other graph networks, we followed (Liu, Gao, and Ji 2020), which selects 30 nodes per class for validation and the rest of the nodes for testing.

Each self-supervised GCL model is trained in an unsupervised manner and tests a simple L2-regularized logistic regression classifier for semi-supervised node classification. Note that for the self-supervised GCL models to learn embedding without labels, the labeled validation set is just used

Datasets	c	Methods								
		GCN	GAT	DGI	MVGRL	GCA	ARIEL	GraphACL	PiGCL	Ours
Cora	1	42.6±11.6	42.1±9.5	55.4±11.4	59.1±10.9	58.4±10.9	56.3±9.3	54.7±12.0	53.2±11.5	63.6±10.9
	2	55.0±7.5	53.2±9.0	64.9±9.0	67.8±8.6	66.0±7.8	65.8±8.0	64.1±9.0	61.8±9.2	72.4±7.7
	3	63.1±6.8	63.2±5.3	71.1±5.6	74.5±4.1	71.5±4.6	72.0±5.6	68.8±5.5	68.4±6.6	76.4±4.7
	4	66.4±6.4	66.3±5.9	72.9±4.5	76.1±3.2	72.9±4.3	74.6±4.8	70.9±5.0	71.3±4.8	78.0±3.3
	20	79.6±1.8	81.2±1.6	82.1±1.3	82.4±1.5	79.0±1.4	81.3±1.3	82.0±1.1	80.0±1.5	82.1±1.3
Cite seer	1	33.8±5.9	31.0±7.2	47.2±9.2	32.8±8.4	38.7±9.0	48.7±10.5	44.0±6.2	41.9±9.7	50.9±12.8
	2	44.8±5.5	41.1±7.2	58.6±4.3	47.8±7.5	49.6±5.3	57.3±4.8	54.2±4.1	56.7±5.5	62.6±5.9
	3	49.2±5.1	48.6±6.7	63.3±4.3	55.2±6.7	54.2±4.7	61.8±3.1	59.3±4.4	61.9±3.8	66.7±2.9
	4	51.7±4.5	52.8±6.6	65.8±2.1	59.3±5.5	57.3±3.3	64.4±2.2	62.6±2.5	64.7±3.6	67.6±2.9
	20	66.0±1.2	68.9±1.8	71.6±1.2	71.1±1.4	65.6±2.4	70.9±1.4	71.5±1.4	71.2±1.1	71.3±1.3
Pub-Med	1	48.6±7.1	47.9±8.5	50.0±9.5	55.3±9.3	57.7±10.5	49.4±7.7	50.3±8.0	50.2±6.4	59.8±12.6
	2	55.8±7.1	54.5±7.7	58.5±8.7	62.7±7.0	66.3±7.6	55.6±5.5	55.6±6.9	54.9±8.2	66.6±9.4
	3	62.1±7.3	61.5±6.8	62.4±7.2	68.5±5.8	71.9±5.4	59.3±5.7	60.6±6.9	57.3±9.1	71.3±7.1
	4	65.1±5.9	64.2±6.1	64.1±6.2	70.6±6.0	73.6±5.4	60.9±6.1	64.2±6.3	60.4±5.9	72.6±5.7
	20	79.0±2.5	78.5±1.8	78.3±2.4	79.5±2.2	81.5±2.5	74.2±2.5	78.6±1.9	76.5±3.5	81.2±1.7
Co-author CS	1	64.8±8.8	64.2±9.0	71.4±6.3	75.4±7.2	59.9±7.6	75.1±7.2	50.5±3.8	43.0±11.5	78.9±6.7
	2	79.2±4.2	80.2±4.1	79.6±5.3	84.7±2.7	72.5±4.6	83.7±3.6	61.5±3.2	62.7±8.9	85.5±4.3
	3	83.3±4.0	85.0±2.7	82.3±3.6	87.5±2.2	77.9±4.1	86.2±2.6	67.3±3.1	72.0±5.6	87.0±3.4
	4	84.2±3.1	86.6±2.1	84.8±2.8	88.5±1.8	80.3±3.1	87.0±1.8	71.9±2.8	77.2±4.5	87.9±2.7
	20	90.0±0.6	90.9±0.7	92.0±0.5	91.5±0.6	90.9±1.1	90.2±0.9	86.9±1.2	91.0±0.7	90.9±1.3
Amazon Photo	1	60.7±9.3	59.0±11.5	53.8±10.7	59.7±9.0	55.3±6.7	69.0±7.8	65.3±9.1	21.9±6.1	73.9±6.7
	2	75.2±7.2	71.7±6.4	62.7±8.5	73.4±6.8	68.0±5.6	75.5±7.1	74.4±5.4	31.3±8.5	81.3±2.6
	3	76.9±5.1	75.6±6.3	66.6±7.7	76.8±6.1	74.4±5.9	78.0±5.2	79.3±4.8	37.4±7.7	82.7±3.0
	4	81.0±4.6	79.3±5.9	70.8±6.0	82.0±2.3	78.8±3.9	80.9±4.6	81.2±5.1	43.5±6.0	84.0±2.9
	20	86.3±1.6	86.5±2.1	83.5±1.2	89.7±1.2	87.0±1.9	90.6±1.8	90.0±1.0	71.8±3.4	89.2±1.2

Table 2: Node classification performance with different label rates on homophilic graph datasets.

Methods	c	GraphACL	PiGCL	AFECL
Actor	1	19.6±2.4	19.6±3.2	21.9±3.0
	2	21.2±2.5	19.9±3.2	22.1±3.7
	3	21.3±2.4	20.7±2.9	22.5±3.1
	4	21.5±1.8	21.1±1.9	22.6±3.2
	20	22.1±1.5	21.1±1.7	25.0±2.3
Chameleon	1	25.8±2.7	21.1±4.0	26.9±4.0
	2	28.5±4.2	25.2±3.4	30.1±3.4
	3	29.3±3.0	26.4±4.4	30.8±3.8
	4	32.0±2.6	26.9±2.8	33.0±3.7
	20	49.2±2.8	34.1±2.6	49.5±3.0

Table 3: Node classification performance with different label rates on heterophilic graph datasets.

to tune the hyperparameters of the logistic regression classifier. We trained the model 20 runs for different random splits of data and reported the averaged performance of all models on each dataset. For all baselines, we use the official code published by the authors. We also report previously published results of other methods as done in (Shen et al. 2023). As for link prediction, we followed (Shiao et al. 2023) in transductive settings. Specifically, we first train the encoder by the different models and freeze its weight before training the decoder to evaluate the link prediction task. We run this 5 times and report the result of the mean average.

4.4 Overall Performance

Node Classification. The results of homophilic and heterophilic graphs node classification accuracy are summarized in Table 2 and From the tables, it is not hard to find that the proposed model shows strong performance across five benchmark graph datasets. Table 3.

Specifically, the proposed model achieves the best or second-best results when the labels are deficient (i.e., only 1, 2, 3 and 4 labeled training nodes per class).

Dataset	E2E-GCN	T-BGRL	AFECL
Cora	91.1±0.4	91.0±0.5	96.5±0.0
Citeseer	92.2±0.6	95.3±0.3	96.6±0.2
Coauthor-CS	96.4±0.5	95.6±0.2	98.2±0.2

Table 4: Area under the ROC curve for the methods in link prediction.

For relatively sufficient labels, i.e., 20 labeled training nodes per class, our proposed model is competitive with the previous state-of-the-art methods. The strong performance verifies the superiority of the proposed model. The performance of the proposed model is mainly attributed to two folds.

Firstly, general augmentation (node masking, dropping edges and etc.) adopted in GCL baselines may alter the semantics of datasets and lead to ineffective embeddings. Our proposed model does not need to use augmentation, which avoids destroying the original topology. The existing baselines tend to regard edges as auxiliary information to learn the representation of nodes. However, the number of edges is significantly more than the number of nodes, which indicates that edge information is relatively wealthy. Our proposed model utilizes edges as the contrastive samples to define positives and negatives, which is different from node-node GCL baselines. Compared to existing baselines, our proposed method performs strongly, and the model performs relatively better than other baselines when there is less labeled data.

Secondly, the existing GCL baselines generally adopt contrastive losses, which treat all other nodes except themselves as negatives without considering the graph topology information. We proposed a novel contrastive loss regarding edges connected to the same node as positives and the

Method	c	Cora	CiteSeer	PubMed	CS	Photo
ECL	1	63.6	50.9	59.8	78.9	73.9
	2	72.4	62.6	66.6	85.5	81.3
	3	76.4	66.7	71.3	87.0	82.7
	4	78.0	67.6	72.6	87.9	84.0
	20	82.1	71.3	81.2	90.9	89.2
w/o ECL	1	48.1	41.7	45.1	50.7	54.2
	2	57.2	49.9	49.7	62.8	67.6
	3	63.2	55.2	52.8	68.1	72.0
	4	66.1	58.1	55.6	69.6	75.4
	20	76.7	67.0	64.7	78.6	83.6

Table 5: Ablation study on node classification with different label rates.

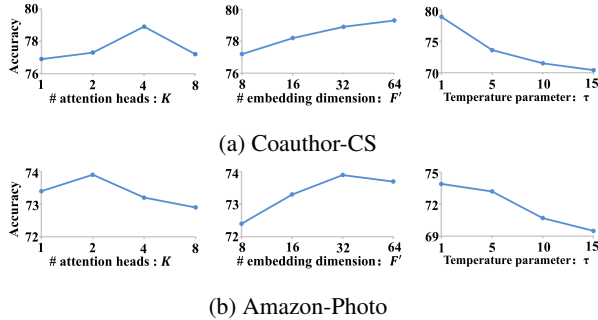


Figure 2: Sensitivity analysis of the hyperparameters K , F' and τ on our model.

remaining as negatives to avoid graph augmentation. Besides, the edge contrastive loss can take full advantage of network topology owing to the definition of positives and negatives following the homogeneity assumption. For node classification, using edge contrastive loss can extract higher-level representation to improve performance when the labeled rate of nodes is low. In summary, the performance of our proposed model compared to existing state-of-the-art baselines verifies the effectiveness of our model. AFECL not only avoids altering the structure of data but also exploits the graph topology to learn the representations.

Link Prediction. In the link prediction task, we evaluate the AFECL on the Cora, Citeseer, and Coauthor-CS datasets, as shown in Table. 4. Intuitively, our proposed model is an edge-level contrastive method, which motivates AFECL to be effective in link prediction.

4.5 Ablation Study

In this section, we conduct ablation experiments to demonstrate the effectiveness of edge contrastive loss of the proposed model. Table 5 shows the classification accuracy of our method for different label rates when using different edge contrastive losses to define different positive and negative pairs. The first loss variant ECL is our proposed method. The second loss variant w/o ECL is to define edges corresponding to nodes connected by only one node as negative pairs. Note that in this case w/o ECL can be regarded as defining two identical augmented graphs corresponding to the same edge as a positive pair, and the same augmented graph with different edges as a negative pair. From the table, we can observe that the proposed edge contrastive loss consistently yields the highest accuracy in the node classi-

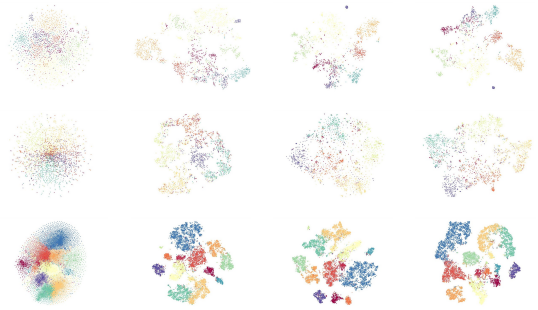


Figure 3: t-SNE visualization of representations on benchmark datasets. The rows represent the visualization results on Cora, CiteSeer, and Coauthor-CS, respectively. The columns suggest the results on the original dataset, GCA, GraphACL, and AFECL.

fication task across the five datasets. Moreover, if data augmentation is not used, defining edges corresponding to nodes connected by only one node as negative pairs will lead to poor results. This reflects that defining positive and negative pairs by considering topological information is very effective for AFECL.

4.6 Hyperparameter Analysis

Figure 2 shows the sensitivity analysis on the hyperparameters K , F' and τ of our model. Since the size of the dimension of $D' = 2KF'$, we do not need additional sensitivity analysis of the parameter D' . For large graphs, our method requires edges sampling, so we observe the impact of different parameters on two various datasets, i.e., Coauthor-CS (co-author network) and Amazon-photo (co-purchase network) as examples. We observe that more heads lead to higher complexity and experiments often achieve the best results when $K = 2$ or $K = 4$. Our model performs better on both two datasets with embedding dimensions F' of $\{32, 64\}$, and has the worst results on embedding dimensions F' of 8. Larger τ will result in lower accuracy for both datasets, note the temperature parameter τ in $\{1, 5, 10, 15\}$. A more detailed hyperparameter analysis can be found in Appendix A.

4.7 Visualization of Embeddings

To more intuitively understand the node embeddings learned by exploiting edge contrast, we use t-SNE (Van der Maaten and Hinton 2008) to visualize the graph representations of AFECL and the baselines, as shown in Figure 3. We can observe that the node embeddings generated by AFECL as well as the baseline method are grouped according to the corresponding node labels.

5 Conclusion

In this paper, we propose AFECL, a novel contrastive framework where the contrastive samples are edges. Firstly, we put forward the module of edge representation learning, where edge features are computed by their connected nodes. Secondly, we managed to implement the edge-level contrast by ECL. Extensive experimental results confirm the effectiveness of AFECL.

References

- Belghazi, M. I.; Baratin, A.; Rajeshwar, S.; Ozair, S.; Bengio, Y.; Courville, A.; and Hjelm, D. 2018. Mutual information neural estimation. In *International conference on machine learning*, 531–540. PMLR.
- Cai, Z.; Jiang, Z.; and Yuan, Y. 2021. Task-related self-supervised learning for remote sensing image change detection. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 1535–1539. IEEE.
- Cao, X.; Shi, Y.; Wang, J.; Yu, H.; Wang, X.; and Yan, Z. 2022. Cross-modal knowledge graph contrastive learning for machine learning method recommendation. In *Proceedings of the 30th ACM International Conference on Multimedia*, 3694–3702.
- Chen, T.; Kornblith, S.; Norouzi, M.; and Hinton, G. 2020. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, 1597–1607. PMLR.
- Donsker, M. D.; and Varadhan, S. S. 1983. Asymptotic evaluation of certain Markov process expectations for large time. IV. *Communications on pure and applied mathematics*, 36(2): 183–212.
- Feng, S.; Jing, B.; Zhu, Y.; and Tong, H. 2022. Adversarial graph contrastive learning with information regularization. In *Proceedings of the ACM Web Conference 2022*, 1362–1371.
- Feng, W.; Zhang, J.; Dong, Y.; Han, Y.; Luan, H.; Xu, Q.; Yang, Q.; Kharlamov, E.; and Tang, J. 2020. Graph random neural networks for semi-supervised learning on graphs. *Advances in neural information processing systems*, 33: 22092–22103.
- Gao, T.; Yao, X.; and Chen, D. 2021. Simcse: Simple contrastive learning of sentence embeddings. *arXiv preprint arXiv:2104.08821*.
- Gong, X.; Yang, C.; and Shi, C. 2023. Ma-gcl: Model augmentation tricks for graph contrastive learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, 4284–4292.
- Gutmann, M.; and Hyvärinen, A. 2010. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, 297–304. JMLR Workshop and Conference Proceedings.
- Hassani, K.; and Khasahmadi, A. H. 2020. Contrastive multi-view representation learning on graphs. In *International conference on machine learning*, 4116–4126. PMLR.
- He, D.; Zhao, J.; Huo, C.; Huang, Y.; Huang, Y.; and Feng, Z. 2024. A New Mechanism for Eliminating Implicit Conflict in Graph Contrastive Learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, 12340–12348.
- Kipf, T. N.; and Welling, M. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.
- Li, Q.; Han, Z.; and Wu, X.-M. 2018. Deeper insights into graph convolutional networks for semi-supervised learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32.
- Li, Q.; Wu, X.-M.; Liu, H.; Zhang, X.; and Guan, Z. 2019. Label efficient semi-supervised learning via graph filtering. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 9582–9591.
- Li, X. 2022. Positive-incentive noise. *IEEE Transactions on Neural Networks and Learning Systems*.
- Li, Y.; Chen, C.; Zheng, X.; Zhang, Y.; Han, Z.; Meng, D.; and Wang, J. 2023. Making users indistinguishable: Attribute-wise unlearning in recommender systems. In *Proceedings of the 31st ACM International Conference on Multimedia*, 984–994.
- Li, Z.; Zhao, B.; and Yuan, Y. 2023. Bio-Inspired Audio-visual Multi-Representation Integration via Self-Supervised Learning. In *Proceedings of the 31st ACM International Conference on Multimedia*, 3755–3764.
- Lin, L.; and Chen, J. 2023. Spectral Augmentation for Self-Supervised Learning on Graphs. In *The Eleventh International Conference on Learning Representations*.
- Liu, M.; Gao, H.; and Ji, S. 2020. Towards deeper graph neural networks. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, 338–348.
- Oord, A. v. d.; Li, Y.; and Vinyals, O. 2018. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*.
- Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32.
- Qiu, J.; Chen, Q.; Dong, Y.; Zhang, J.; Yang, H.; Ding, M.; Wang, K.; and Tang, J. 2020. Gcc: Graph contrastive coding for graph neural network pre-training. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, 1150–1160.
- Rao, J.; Zheng, S.; Mai, S.; and Yang, Y. 2022. Communicative Subgraph Representation Learning for Multi-Relational Inductive Drug-Gene Interaction Prediction. *arXiv preprint arXiv:2205.05957*.
- Sen, P.; Namata, G.; Bilgic, M.; Getoor, L.; Galligher, B.; and Eliassi-Rad, T. 2008. Collective classification in network data. *AI magazine*, 29(3): 93–93.
- Shchur, O.; Mumme, M.; Bojchevski, A.; and Günnemann, S. 2018. Pitfalls of graph neural network evaluation. *arXiv preprint arXiv:1811.05868*.
- Shen, X.; Sun, D.; Pan, S.; Zhou, X.; and Yang, L. T. 2023. Neighbor contrastive learning on learnable graph augmentation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, 9782–9791.
- Shiao, W.; Guo, Z.; Zhao, T.; Papalexakis, E.; Liu, Y.; and Shah, N. 2023. Link Prediction with Non-Contrastive Learning. In *Eleventh International Conference on Learning Representations (ICLR)*.

- Sohn, K. 2016. Improved deep metric learning with multi-class n-pair loss objective. *Advances in neural information processing systems*, 29.
- Sun, M.; Xing, J.; Wang, H.; Chen, B.; and Zhou, J. 2021. MoCL: data-driven molecular fingerprint via knowledge-aware contrastive learning from molecular graph. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 3585–3594.
- Thakoor, S.; Tallem, C.; Azar, M. G.; Azabou, M.; Dyer, E. L.; Munos, R.; Veličković, P.; and Valko, M. 2021. Large-scale representation learning on graphs via bootstrapping. *arXiv preprint arXiv:2102.06514*.
- Van der Maaten, L.; and Hinton, G. 2008. Visualizing data using t-SNE. *Journal of machine learning research*, 9(11).
- Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Lio, P.; and Bengio, Y. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903*.
- Veličković, P.; Fedus, W.; Hamilton, W. L.; Liò, P.; Bengio, Y.; and Hjelm, R. D. 2019. Deep Graph Infomax. In *International Conference on Learning Representations*.
- Wan, S.; Zhan, Y.; Liu, L.; Yu, B.; Pan, S.; and Gong, C. 2021. Contrastive graph poisson networks: Semi-supervised learning with extremely limited labels. *Advances in Neural Information Processing Systems*, 34: 6316–6327.
- Wang, H.; Zhang, J.; Zhu, Q.; Huang, W.; Kawaguchi, K.; and Xiao, X. 2022. Single-pass contrastive learning can work for both homophilic and heterophilic graph. *arXiv preprint arXiv:2211.10890*.
- Wang, M.; Zheng, D.; Ye, Z.; Gan, Q.; Li, M.; Song, X.; Zhou, J.; Ma, C.; Yu, L.; Gai, Y.; et al. 2019. Deep graph library: A graph-centric, highly-performant package for graph neural networks. *arXiv preprint arXiv:1909.01315*.
- Xia, J.; Wu, L.; Chen, J.; Hu, B.; and Li, S. Z. 2022. Simgrace: A simple framework for graph contrastive learning without data augmentation. In *Proceedings of the ACM Web Conference 2022*, 1070–1079.
- Xiao, T.; Zhu, H.; Chen, Z.; and Wang, S. 2024. Simple and asymmetric graph contrastive learning without augmentations. *Advances in Neural Information Processing Systems*, 36.
- Xie, Y.; Xu, Z.; Zhang, J.; Wang, Z.; and Ji, S. 2022. Self-supervised learning of graph neural networks: A unified review. *IEEE transactions on pattern analysis and machine intelligence*, 45(2): 2412–2429.
- Yang, H.; Gao, J.; Yuan, Y.; and Li, X. 2023. Imbalanced Aircraft Data Anomaly Detection. *arXiv preprint arXiv:2305.10082*.
- Yang, W.; and Mirzasoleiman, B. 2024. Graph Contrastive Learning under Heterophily via Graph Filters. *arXiv:2303.06344*.
- Yang, Z.; Cohen, W.; and Salakhudinov, R. 2016. Revisiting semi-supervised learning with graph embeddings. In *International conference on machine learning*, 40–48. PMLR.
- You, Y.; Chen, T.; Shen, Y.; and Wang, Z. 2021. Graph contrastive learning automated. In *International Conference on Machine Learning*, 12121–12132. PMLR.
- You, Y.; Chen, T.; Sui, Y.; Chen, T.; Wang, Z.; and Shen, Y. 2020. Graph contrastive learning with augmentations. *Advances in neural information processing systems*, 33: 5812–5823.
- Zhang, G.; Li, Z.; Huang, J.; Wu, J.; Zhou, C.; Yang, J.; and Gao, J. 2022a. efraudcom: An e-commerce fraud detection system via competitive graph neural networks. *ACM Transactions on Information Systems (TOIS)*, 40(3): 1–29.
- Zhang, H.; Shi, J.; Zhang, R.; and Li, X. 2023. Non-Graph Data Clustering via-Bipartite Graph Convolution. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 45(07): 8729–8742.
- Zhang, H.; Wu, Q.; Wang, Y.; Zhang, S.; Yan, J.; and Yu, P. S. 2022b. Localized contrastive learning on graphs. *arXiv preprint arXiv:2212.04604*.
- Zhang, H.; Wu, Q.; Yan, J.; Wipf, D.; and Yu, P. S. 2021. From canonical correlation analysis to self-supervised graph neural networks. *Advances in Neural Information Processing Systems*, 34: 76–89.
- Zhang, H.; Xu, Y.; Huang, S.; and Li, X. 2024. Data Augmentation of Contrastive Learning is Estimating Positive-Incentive Noise. *arXiv preprint arXiv:2408.09929*.
- Zhang, H.; Zhu, Y.; and Li, X. 2024. Decouple Graph Neural Networks: Train Multiple Simple GNNs Simultaneously Instead of One. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.
- Zhu, H.; Sun, K.; and Koniusz, P. 2021. Contrastive laplacian eigenmaps. *Advances in Neural Information Processing Systems*, 34: 5682–5695.
- Zhu, Y.; Xu, Y.; Yu, F.; Liu, Q.; Wu, S.; and Wang, L. 2020. Deep graph contrastive representation learning. *arXiv preprint arXiv:2006.04131*.
- Zhu, Y.; Xu, Y.; Yu, F.; Liu, Q.; Wu, S.; and Wang, L. 2021. Graph contrastive learning with adaptive augmentation. In *Proceedings of the Web Conference 2021*, 2069–2080.
- Zhu, Z.; Galkin, M.; Zhang, Z.; and Tang, J. 2022. Neural-symbolic models for logical queries on knowledge graphs. In *International Conference on Machine Learning*, 27454–27478. PMLR.