Adversarial Contrastive Graph Augmentation with Counterfactual Regularization

Tao Long[1], Lei Zhang[1], Liang Zhang[2] *, Laizhong Cui[1]

[1]College of Computer Science and Software Engineering, Shenzhen University , Shenzhen, Guangdong, China
[2]Shenzhen Research Institute of Big Data, Shenzhen, Guangdong, China
longtao2021@email.szu.edu.cn, {leizhang,cuilz}@szu.edu.cn, zhangliangshuxue@gmail.com,

Abstract

With the advancement of graph representation learning, self-supervised graph contrastive learning (GCL) has emerged as a key technique in the field. In GCL, positive and negative samples are generated through data augmentation. While recent works have introduced model-based methods to enhance positive graph augmentations, they often overlook the importance of negative samples, relying instead on rule-based methods that can fail to capture meaningful graph patterns. To address this issue, we propose a novel model-based adversarial contrastive graph augmentation (ACGA) method that automatically generates both positive graph samples with minimal sufficient information and hard negative graph samples. Additionally, we provide a theoretical framework to analyze the process of positive and negative graph augmentation in self-supervised GCL. We evaluate our ACGA method through extensive experiments on representative benchmark datasets, and the results demonstrate that ACGA outperforms state-of-the-art baselines.

Code — https://github.com/longtao-09/ACGA

Introduction

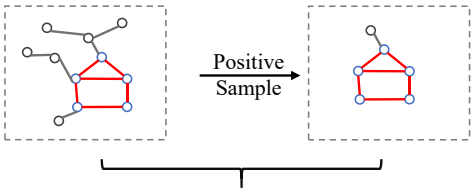
Graphs, one of the most common non-Euclidean data structures, are found in a wide range of applications, such as citation networks (Tang et al. 2008), social networks (Hamilton, Ying, and Leskovec 2017), and recommender systems (Ying et al. 2018). Graph representation learning (GRL) algorithms, which aim to encode graph-structured data into low-dimensional vector representations, have recently shown great potential in these applications. Their effectiveness has been demonstrated across various downstream tasks, including link prediction (Zhang and Chen 2018), node classification (Kipf and Welling 2016a), and graph classification (Lee, Lee, and Kang 2019). However, annotating graph data labels requires significant resources (Vogel 1997). As a result, self-supervised GRL, which operates with limited or even no labels, has garnered significant attention.

Graph contrastive learning (GCL) has recently emerged as one of the most important techniques for self-supervised graph representation learning (GRL) (Chen et al. 2020b; Khosla et al. 2020). In GCL, each original graph generates a set of positive and negative samples through sampling or data augmentation. Due to the complex semantic structure of graphs, the primary challenge in GCL is generating high-quality positive and negative pairs. Empirical, rule-based graph augmentation methods—such as node dropping, edge perturbation, and subgraph sampling—are widely used to create these pairs. For instance, GMI (Peng et al. 2020) and SUBG-CON (Jiao et al. 2020) propose more efficient GCL methods by generating positive and negative samples from local subgraphs of each node. However, in these rule-based graph augmentation methods, the patterns in the generated samples can be quite limited. For example, in DGI (Hjelm et al. 2018), a positive sample is defined as the mean representation of all nodes in the original graph. Such positive samples may not be suitable for nodes with unique characteristics, such as users from minority communities in social networks.

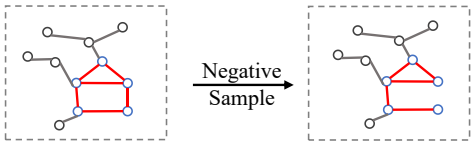
Recent works have started to focus on model-based graph augmentation methods using adversarial training models, which avoid ad-hoc decisions based on human priors and have the potential to augment diverse heterogeneous graph data. For example, ARIEL (Shengyu Feng and Tong 2021) introduced a new adversarial approach to data augmentation in graph contrastive learning, regularizing the mutual information between positive pairs to stabilize training. AD-GCL (Susheel Suresh and Neville 2021) enabled graph neural networks (GNNs) to capture key information during training by optimizing adversarial graph augmentation strategies with edge regularization. However, all these methods primarily focus on automatically augmenting positive graph samples and often overlook the importance of negative samples. To illustrate the learning process with augmented graphs, we present a house example in Fig.1. The goal is to train an optimal encoder that captures the key information related to the house. Previous works like AD-GCL and ARIEL generate only positive samples, which help the encoder learn house-related information from a positive perspective, as shown in Fig.1(a). However, learning from a negative perspective is equally important for the encoder, as illustrated in Fig. 1(b). Hard negative graph samples can assist the encoder in capturing a more complete understanding of the house.

To address the aforementioned issues, we propose a novel

*Corresponding authors



(a) Positive View



(b) Negative View

Figure 1: A toy example for the encoder

adversarial contrastive graph augmentation method that generates both positive samples with minimal sufficient information and hard negative samples. These samples assist the encoder in extracting key information from both positive and negative perspectives. A significant challenge in generating positive and negative samples is that it is difficult to classify a graph as positive or negative without knowing the label information. To overcome this, we introduce a counterfactual graph approximation method to generate typical positive and negative graphs, ensuring high semantic similarity or dissimilarity with the original graph. We further propose a novel Conditional Variational Graph Auto-encoder (C-VGAE) framework, where the typical positive and negative graphs serve as regularization and the original graph as the condition, enabling the generation of diverse positive and negative graph samples. Leveraging these augmented graph samples, we design both intra-graph and inter-graph contrastive learning strategies to build a robust self-supervised GCL framework. Additionally, this paper provides a theoretical framework for graph augmentation that unifies the processes of positive and negative graph augmentation in self-supervised GCL. We establish a theoretical upper bound for the mutual information of augmented graphs, offering high-level guidance on graph augmentations. Our main contributions can be summarized as follows:

- To the best of our knowledge, this is the first work to design a model-based framework for generating negative samples and provide a theoretical analysis that unifies the processes of positive and negative graph augmentations.
- We propose a novel adversarial contrastive graph augmentation method that generates both positive samples with minimal sufficient information and sufficiently hard negative samples.
- We conduct extensive experiments to validate the effectiveness of the proposed framework across multiple widely used benchmarks for various downstream tasks.

Methodology

In this section, we propose a novel framework ACGA for learning effective node representations for various downstream tasks. We first illustrate the theoretical graph augmentation framework to unify the positive and negative graph augmenting processes and the theoretical motivation of our ACGA framework. Then, we demonstrate the instantiation of ACGA with counterfactual graph approximation to regularize the graph augmentation and adversarial graph augmentation via a Conditional Variational Graph Auto-encoder (C-VGAE) framework.

Theoretical Motivation of ACGA

We let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ denote a graph, where $\mathcal{V} = \{v_1, v_2, ..., v_N\}$, $\mathcal{E} \in \mathcal{V} \times \mathcal{V}$ are the set of nodes and edges of the graph, respectively. Each node in the graph has D-dimensional features, and the feature matrix can be denoted as $X = \{x_1, x_2..., x_N\} \in R^{N \times D}$. The adjacency matrix of a graph is:$A \in \{0, 1\}^{N \times D}$. In this work, we focus on undirected graphs, but the proposed method can be easily extended to directed graphs. Considering a set of above graphs $\mathcal{G}$, we aim to learn an encoder f: $\mathbb{R}^{N_i \times D} \times \{0, 1\}^{N_i \times N_i} \rightarrow \mathbb{R}^{N_i \times d}, d \ll D$ that maps the feature matrix $\mathbf{X}_i$ and adjacency matrix $\mathbf{A}_i$ in graph $G_i \in \mathcal{G}$ to low dimensional node representations $\mathbf{H}_i \in \mathbb{R}^{N_i \times d}$, where $\mathbf{H}_i$ provides highly discriminative features in graph G_i for the downstream task. We associate each graph G_i with label $Y_i \in \mathcal{Y}$ in the task and assume (G_i, Y_i) are IID sampled from a joint distribution $\mathbb{P}_{\mathcal{G} \times \mathcal{Y}} = \mathbb{P}_{\mathcal{Y}|\mathcal{G}} \mathbb{P}_{\mathcal{G}}$.

Traditional adversarial graph augmentation like AD-GCL (Susheel Suresh and Neville 2021) is inspired by Graph Information Bottleneck (GIB) (Wu et al. 2020a; Yu et al. 2021) to learn the minimal information that is sufficient to identify each graph. Different from GIB, AD-GCL does not require the label from the downstream task and thus can be applied to self-supervised training where there exist few or no labels. It optimizes the min-max optimization problem as follows:

$$\text{AD-GCL: } \min_{\psi} \max_{\theta} \mathbb{E}_{G \sim \mathbb{P}_G}[I(f_\theta(G); f_\theta(G^+))], \quad (1)$$

where $G^+ \sim g_\psi(G)$, $f_\theta(\cdot)$ is the encoder and g_ψ is the graph generator. Without limiting the generation process, the augmented graph might drop the essential information related to the task and result in the collapse of training. To guarantee that the generated graph contains useful information, AD-GCL adds the constraint of dropping edges to ensure the generated graph is not too far from the original graph. The hidden assumption is that a slight change in the graph would not destroy the key information related to the label. To demonstrate it clearly, we have the following definition.

Definition 1 *(Positive Graph Set) For a given graph $G \in \mathcal{G}$, we define the graph $G^+ \in \mathcal{G}$ as its positive sample if G^+ and G have the same label. Then the positive graph set of graph G can be denoted as $\mathbb{D}_G^+ = \{G^+ \in \mathcal{G} | Y(G) = Y(G^+)\}$.*

In AD-GCL, since the label cannot be utilized to generate the positive graph set, it adopts an approximate method via regularizing the generated graphs, i.e., $\mathbb{D}_G^+ = \{G^+ \in \mathcal{G} | d_g(G, G^+) \leq \epsilon\}$. Here, the distance function $d_g(G, G^+)$ measures the difference between two graphs. AG-GCL adopts the regularization of total dropped edges,

i.e., $d_g(G, G^+) = \frac{1}{|E_G|}\sum_{e\in E_G}[1 - w_e^+]$, where w_e^+ is the probability of e in G^+. We define the graph distance between a graph G and a graph set $\mathcal{D}$ as $D(G, \mathcal{D}) = \min_{G'\in D} d_g(G, G')$. Then, we can rewrite the AD-GCL as a positive adversarial graph augmentation problem (P-AGA):

$$\min_{\psi}\max_{\theta}\mathbb{E}_{G\sim\mathbb{P}_G}[I(f_\theta(G); f_\theta(G^+)) + \lambda D(G^+, \mathbb{D}_G^+)], \quad (2)$$

where $G^+ \sim g_\psi^+(G)$ and λ is positive constant. Different from GIB, P-AGA indirectly utilizes the label if the label information is considered. It includes the information related to the label via constraining the generated positive graphs, i.e., $D(G^+, \mathbb{D}_G^+)$. On the other hand, adversarial optimization requires the generated positive graph to have minimal mutual information with the original graph, which assists the encoder in learning the most important information related to the label. The following theorem demonstrates the relation between GIB and P-AGA. We put the proofs of all theorems in this paper into the appendix.

Theorem 1 *Denote the optimal encoder for GIB as h^* and that for P-AGA as f_P^*. If the positive graph set is obtained by Definition 1, $\gamma \leq 1$ and $\lambda \geq 1$, then both encoders extract the same information related to the label, i.e., $I(h^*(G); Y) = I(f_P^*(G); Y)$.*

Theorem 1 states that both the optimal encoder of GIB and P-AGA can capture the most important information related to the label, although they extract such information from different views. It also shows that our P-AGA provides a unified framework for both supervised GRL methods like GIB and self-supervised GRL methods like AD-GCL.

As previously mentioned, the negative pairs are important since they can assist the encoder in learning the key information from the negative view. The recent model-based methods like AD-GCL only consider positive graph augmentations. The recent work R-GCL (Li et al. 2022) attempted to learn the negative graph (complement graph), but it is not hard enough. When using the "house" example in Fig. 1, their negative graph (complement graph) is the rest graph excluding the "house" while what we expect is the incomplete "house" shown in Fig. 1(b). The challenge of negative graph augmentations is to discover high-quality negative pairs.

Definition 2 *(Negative Graph Set) For a given graph $G \in \mathcal{G}$, we define the graph $G^- \in \mathcal{G}$ as its negative sample if G^- and G have different labels. Then the negative graph set of graph G can be denoted as $\mathbb{D}_G^- = \{G^- \in \mathcal{G} | Y(G) \neq Y(G^-)\}$.*

Similar to the positive graph set, when we consider self-supervised GRL with few or no labels, the negative graph set cannot be calculated directly, but should be approximated instead. In the approximation of the negative graph set, we also take the hidden assumption similar to the positive graph set, i.e., a slight change of the negative graph would not destroy the key information related to the label. However, we cannot apply the method in the positive graph set since the original set can be one special case of the positive graph but we cannot know the negative graph without knowing the label. To address this issue, we propose a counterfactual graph augmentation method to generate a typical negative graph in the later subsection. The typical negative graph $\bar{G}$ should be far enough from the original graph G to make sure the typical graph is a real negative sample. However, if the typical negative graph can be justified easily, the generation of hard negative samples based on this graph might be difficult. Then, given the typical negative graph, we can approximate the negative graph set as $\mathbb{D}_G^- = \{G^- \in \mathcal{G} | d_g(\bar{G}, G^-) \leq \epsilon\}$.

For a negative pair (G, G^-), the encoder function $f(\cdot)$ should ensure minimal mutual information between the original graph G and the negative graph G^-. On the other hand, the harder negative sample would be more useful for training a better representation function. Thus, the negative adversarial graph augmentation (N-AGA) problem can be formulated as the min-max optimization:

$$\min_{\phi}\max_{\theta}\mathbb{E}_{G\sim\mathbb{P}_G}[-I(f_\theta(G); f_\theta(G^-)) + \lambda D(G^-, \mathbb{D}_G^-)], \quad (3)$$

where $G^- \sim g_\phi^-(G)$. The encoder $f_\theta(\cdot)$ focuses on discovering the difference between the original graph G and the generated negative graph G^- while the adversarial process aims to generate the hard negative samples similar to the original graph. The following theorem states the upper bound of mutual information between the original graph and the generated negative graph.

Theorem 2 *Denote the optimal encoder and generator in N-AGA as f_N^* and g^{*-} accordingly. For any $G \in \mathcal{G}$, there exist a constant $c > 0$ such that if $\lambda \geq c$, then the mutual information between G and its generated negative graph $G^- \sim g^{*-}(G)$ is upper bounded by $I(f_N^*(G); f_N^*(G^-)) \leq I(G; f_N^*(G)) - I(f_N^*(G); Y) \leq I(G; G^-|Y)$.*

Theorem 2 states that an upper bound of the mutual information is guaranteed by the encoder function to learn from the original graph and the generated negative graph, which matches our aim. It also shows that the augmented negative graph includes enough information irrelevant to the downstream task, which makes the negative graph samples hard enough.

P-AGA and N-AGA optimize the encoder from positive and negative views, both benefiting the training of the encoder. To utilize the advantage of the P-AGA and N-AGA optimizations, we combine them into an adversarial contrastive graph augmentation problem (ACGA). Formally, we have the following min-max optimization:

$$\min_{\phi,\psi}\max_{\theta}\mathbb{E}_{G\sim\mathbb{P}_G}\big[U_\theta(G, G^+, G^-) + \lambda(V(G^+) + V(G^-))\big], \quad (4)$$

where $G^+ \sim g_\psi^+(G)$, $G^- \sim g_\phi^-(G)$, $U_\theta(\cdot)$ represents a contrastive learning function and $V(G^s) = D(G^s, \mathbb{D}_G^s)$.

ACGA provides higher requirements for the encoder such that it should get rid of the abundant information unrelated to the label and also capture the complete information related to the label. The first part in ACGA provides the contrastive learning for the graph encoder and the second part provides the regularization for the augmented positive and negative graphs. We would provide an instantiation about the ACGA, which adopts a novel counterfactual graph approximation to conduct the regularization, and a novel C-VGAE framework to generate positive and negative graph samples.

Instantiation of ACGA

In this subsection, we introduce the instantiation of the ACGA, including a novel counterfactual graph approximation method to regularize the augmented positive and negative graphs, the adversarial graph augmentation to generate the positive and negative graphs, and the graph contrastive learning to assist in the self-supervised GRL. The overall framework of our ACGA is demonstrated in Fig. 2

Counterfactual Graph Approximation: As previously mentioned, since we cannot directly use labels to generate positive and negative graph sets, we must rely on an approximation method. For the positive graph set, we propose using a typical positive graph that maintains high semantic similarity to the original graph but differs from G in more meaningful ways than simply adding or removing a few edges. For the negative graph set, we generate a typical negative sample for graph G, which would have a counterfactual label but remains sufficiently similar to graph G.

We introduce the counterfactual graph approximation framework used to obtain these typical positive and negative graphs. They are employed as regularization for graph augmentations, ensuring that generated samples belong to the positive and negative graph sets. The generated samples, along with the original graph, are then used to construct the graph contrastive learning framework. We propose to maintain high semantic similarity (or dissimilarity) between the original graph and its typical graphs by imposing additional constraints on an auxiliary variable, denoted as S. The intuition behind the auxiliary variable is that, while not a label itself, it is closely related to the label or graph. Changing the auxiliary variable can significantly alter the graph structure. In this paper, we use the degree of each node as the auxiliary variable, which has proven effective across various tasks in our experiments. Other auxiliary variables could also be used; for example, from a graph spectral perspective, the eigenvalue of each node could serve as an auxiliary variable (Lin, Chen, and Wang 2022). In specific scenarios like recommendation systems, the number of clicks/comments on each item could be used as the auxiliary variable.

We illustrate the causal graph for graph augmentations in Fig.3. The adjacency matrix A in graph G and the auxiliary variable S both influence the label variable Y. They are driven by a common hidden variable Z, which represents the hidden embedding for each node. Based on this causal graph, we outline the three steps of counterfactual reasoning as described by Judea Pearl(Pearl 2009). In the abduction step, we use the observed data (i.e., the graph set) to infer the distribution of the hidden variable Z (e.g., graph embedding). In the action step, we manipulate the graph structure (e.g., by adding or removing edges). In the prediction step, we predict the label Y (e.g., graph label) for a new graph. The typical positive graph augmentation can be viewed as an intervention on A without altering the auxiliary variable S. For typical negative graph augmentation, we prefer to reverse the auxiliary variable S by intervening in A. We define the former as a P-counterfactual intervention and the latter as an N-counterfactual intervention.

To conduct the counterfactual graph approximation more conveniently, we transfer the auxiliary variable S for each node to an edge auxiliary variable. More specifically, the edge auxiliary variable T is defined as $T_{i,j} = 1$ if $S_i = S_j$, and $T_{i,j} = 0$ otherwise. In addition, we calculate the similarity $d_n(v_i, v_j)$ between v_i and v_j via node features or the embedding learned from the graph structure. Then, we can define the P-counterfactual and N-counterfactual links as:

$$(v_a^+, v_b^+) = \arg\min_{(v_i,v_j)\in E}\{d_n(v_i, v_a) + d_n(v_j, v_b) \mid T_{a,b} = T_{i,j}\} \tag{5}$$

$$(v_a^-, v_b^-) = \arg\min_{(v_i,v_j)\in E}\{d_n(v_i, v_a) + d_n(v_j, v_b) \mid T_{a,b} = \tilde{T}_{i,j}\} \tag{6}$$

where $\tilde{T}_{i,j} = 1 - T_{i,j}$ and we exclude the original nodes v_a and v_b when finding the P-counterfactual and N-counterfactual links. We utilize the P-counterfactual link to generate the P-counterfactual graph $\bar{G}^+$ as the typical positive graph and the N-counterfactual link to generate the N-counterfactual graph $\bar{G}^-$ as the typical negative graph. Then, we approximate the positive graph set as $\mathbb{D}_G^+ = \{G^+ \in \mathcal{G} | d(\bar{G}^+, G^+) \leq \epsilon_1\}$ and the negative graph set as $\mathbb{D}_G^- = \{G^- \in \mathcal{G} | d_g(\bar{G}^-, G^-) \leq \epsilon_2\}$. We simplify the regularization for positive and negative graphs as $D(G+, \mathbb{D}_G^+) = d_g(\bar{G}^+, G^+)$ and $D(G^-, \mathbb{D}_G^-) = d_g(\bar{G}^-, G^-)$.

Adversarial Graph Augmentation: To generate the positive and negative graphs, we adopt a Variational Graph Auto-encoder (VGAE) (Kipf and Welling 2016b) framework. VGAE is designed to learn low-dimensional representations of graphs in an unsupervised manner. It uses a probabilistic encoder-decoder architecture, with the encoder network learning a distribution over latent variables that encode the input graph and the decoder network reconstructing the input graph via latent variables. Our target is to generate the positive graph pair $(\bar{G}^+, G^+)$ and the negative graph pair $(\bar{G}^-, G^-)$ via VGAE framework. Traditional VGAE utilizes $q(Z|G)$ to encode the graph and $p(G^*|Z)$ to decode the hidden embedding. In this work, we propose a novel Conditional VGAE (C-VGAE) that considers an extra conditional graph. For the positive graph augmentation, we have the encoder $q(Z^+|\bar{G}^+, G)$ and decoder $p(G^{*+}|Z^+)$. For the negative graph generation, we have the encoder $q(Z^-|\bar{G}^-, G)$ and decoder $p(G^{*-}|Z^-)$. Note that the graph G is the common condition for encoders in both positive and negative graph augmentations.

The encoders learn a distribution over latent variables Z^s based on the typical graphs G^s and the conditional graph G for different nodes independently:

$$q^s(Z^s \mid G, G^s) = \prod_{i=1}^{N} q^s\left(z_i^s \mid G, G^s\right). \tag{7}$$

Specifically, we assume a Gaussian distribution for the latent variable of each node:

$$q^s\left(z_i^s \mid G, G^s\right) = \mathcal{N}\big(z_i^s \mid [\mu_i, \mu_{i_s}], diag(\sigma_i^2, \sigma_{i_s}^2)\big), \tag{8}$$

where $s \in \{+, -\}$ represents the positive and negative indicator, (μ_i, σ_i) are the mean and variance for node i in conditional graph G learned by $\mu = GCN_\mu(G)$, and $\log\sigma = GCN_\sigma(G)$ respectively. $(\mu_{i_s}, \sigma_{i_s})$ are the mean

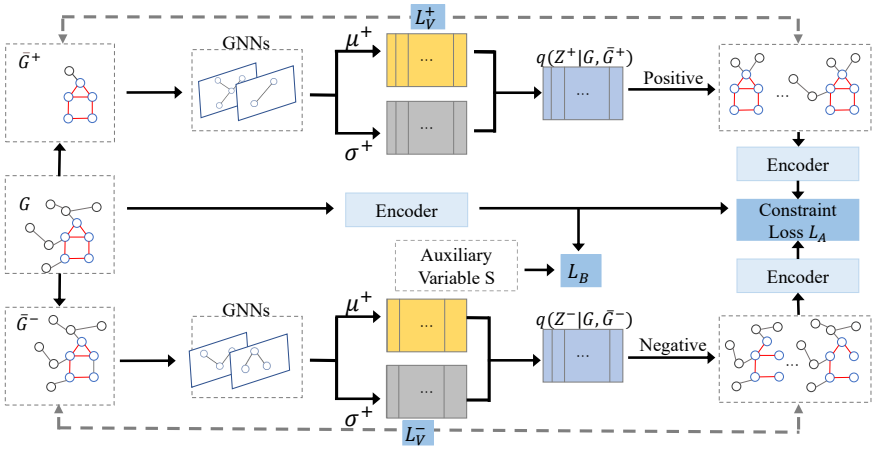


Figure 2: Instantiated framework of ACGA

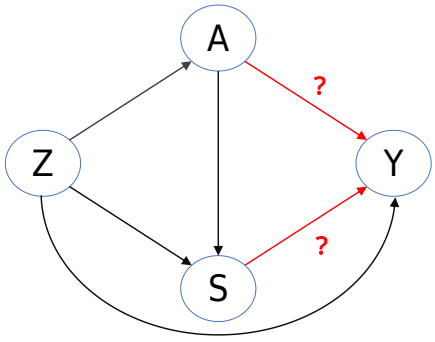


Figure 3: Causal graph for counterfactual augmentation

and the variance for node i in the positive/negative graph learned by $\mu_s = GCN_{\mu_s}(G^s)$ and $\log \sigma_s = GCN_{\sigma_s}(G^s)$ respectively.

The decoder calculates the probability of edges to reconstruct the graph based on the latent variables of node pairs:

$$p(G^{*s} \mid Z) = \prod_{i=1}^{N} \prod_{j=1}^{N} p^s \left(A_{ij}^s \mid z_i^s, z_j^s\right), \tag{9}$$

where $p(A_{ij} = 1 \mid z_i, z_j) = f_G(z_i, z_j)$, $A_{i,j}^s = 1$ indicates the existence of an edge between nodes i and j for the generated positive and negative graph, and $f_G = \sigma(z_i^T, z_j)$ generates the probability of this edge.

The loss function includes the distance measurement between the generated graph and the original graph and the divergence of the node representation distribution and the normal distribution:

$$\mathcal{L}_V^s = \mathbb{E}[\log p^s(\tilde{G}^s \mid Z^s)] - \beta KL[q^s(Z^s \mid G, G^s) \| p(Z)]. \tag{10}$$

The first item is the reconstruction loss with the cross-entropy function, which is responsible for predicting the probability of edges' existence. The second term is the KL divergence between the variational and prior distributions. We utilize the isotropic Gaussian distribution $p(Z) = \prod_i N(0, I)$ as the prior. β reweighs the KL-divergence, which learns the disentangled factors in Z^s.

Graph Contrastive Learning: The original graph G and its augmented positive graph G^+ and negative graph G^- can be utilized to provide the inter-graph contrastive learning(inter-GCL). We use the GCN network as the encoder $f(\cdot)$. More formally, we have:

$$H = GCN_\theta(G); H_+ = GCN_\theta(G^+); H_- = GCN_\theta(G^-). \tag{11}$$

Note that G^+ and G^- are generated via sampling from the probability $f_G(\cdot, \cdot)$. The sampling process and the normalizing process $\hat{D}^{-\frac{1}{2}}\hat{A}\hat{D}^{-\frac{1}{2}}$ take lots of computation. Denote the estimated probability of the augmented matrix as $\hat{A}^s$ learned by $f_G(\cdot, \cdot)$. We substitute the normalizing process with $\hat{A}^s$ and simplify the GCN encoder as:

$$H_s^{(l+1)} = \sigma\left((\tilde{A}^s + I_N)H_s^{(l)}W^{(l)}\right). \tag{12}$$

Only the encoders for augmented positive and negative graphs use the above simplification, which is different from the encoder for graph G. The parameters of encoders for all these graphs are still the same. With the encoder to capture the hidden information, we construct the graph contrastive learning via a triplet loss as follows:

$$\mathcal{L}_A = \frac{1}{N}\sum_{i=1}^{N}[\|h_i - h_i^+\|_2^2 - \gamma \left\|h_i - h_i^-\right\|_2^2 + \xi]_+, \tag{13}$$

where ξ is the threshold that controls the effective difference and γ balances the importance from the positive view and the negative view. We measure the mutual information between graph G and augmented graph G^s simply by their difference, i.e., $I(h, h^s) = \|h - h^s\|_2^2$.

Besides inter-graph contrastive learning, we also introduce intra-graph contrastive learning(intra-GCL). For each node in the graph G, we treat the other nodes with the same auxiliary variable as the positive nodes and negative nodes otherwise. More formally, we have the intra-graph contrastive learning loss as:

$$\mathcal{L}_B = \frac{1}{N}\sum_{i=1}^{N}\log\frac{\exp\left(sim(h_i, \tilde{h}_i)\right)}{\sum_{k=1, s_i \neq s_k}^{N}\exp\left(sim(h_i, h_k)\right)}, \tag{14}$$

where $\tilde{h}_i$ is a positive sample of node i with the same auxiliary variable. We instantiate the optimization of ACGA as:

$$\min_{\phi,\psi} \max_{\theta} \lambda_1 \mathcal{L}_A(\theta, \psi, \phi) + \lambda_2 \mathcal{L}_B(\theta) + (\mathcal{L}_V^+(\phi) + \mathcal{L}_V^-(\psi)). \tag{15}$$

We instantiate the contrastive learning function $U_\theta(\cdot)$ in Eq. 4 via both the inter-graph contrastive learning loss and intra-graph contrastive learning loss.

Algorithm Time Complexity Analysis: Our work introduces two additional components that result in increased time and memory usage. The first component is the counterfactual graph approximation. Calculating distance metrics between different node pairs has a time complexity of $O(N^2)$ and a space complexity of $O(N^2)$. A typical graph augmentation requires the calculation of all N^2 node pairs. For each pair, one node must find the most similar nodes, requiring a check of N nodes. This leads to a time complexity of $O(N^2 \times N)$, resulting in a total time complexity of $O(N^2 + N^3)$ and a space complexity of $O(N^2)$. It's important to note that generating positive and negative typical graphs occurs before training, similar to the calculation of $D^{-1/2}$ in the GCN model with a time complexity of $O(N^3)$, which does not impact the training process. The second component is involved in the training process, where we introduce an additional negative adversarial graph augmentation. It takes the same amount of time as the positive adversarial graph augmentation. In our framework, we offer a non-trivial optimization by avoiding any recalculation of $D^{-1/2}$ in Eq. 11 - 12. A detailed comparison of the average running time per epoch is provided in the Appendix.

Experiments

In this section, we evaluate our ACGA method in node classification tasks and link prediction tasks respectively, and conduct ablation experiments to verify the role of each component. We put the parameter analysis in the appendix.

Experimental Setups

Datasets: We evaluate our method on total seven benchmark datasets across domains: citation networks (CORA, CITESEER, PUBMED (Sen et al. 2008)), social networks (BLOGCATALOG, FLICKR (Huang, Li, and Hu 2017)), and air traffic (AIR USA (Wu et al. 2020b)), co-authorship networks(Coauthor-CS (Shchur et al. 2018)).

Baselines: We consider total eight baselines including AD-GCL (Suresh et al. 2021), JOAO (You et al. 2021), GAUG (Zhao et al. 2021),MVGRL (Hassani and Khasahmadi 2020),GCA (Zhu et al. 2021), AFGRL (Lee, Lee, and Park 2022), GAE (Kipf and Welling 2016b) and GCNs.

Implementation Details: Our experimental device is an NVIDIA Tesla T4 16GB graphics card. Large datasets like Pubmed, Coauthor-CS lead to OOM (Out of Memory) for most baselines. Thus, we sampled subgraphs of 60% of nodes with graph sizes more than 10k. For all baselines and our models, the hidden layer dimension is set to 128 for node classification tasks and 32 for link prediction tasks, and the rest of the parameters remain as the original parameters of Git Hub. We use Adam optimizer with learning rate set to 0.01. The statistics of each dataset and more details of implementation for different tasks are shown in the appendix.

Overall Performance

Node Classification: We observe performance improvements with ACGA over most baselines across all datasets, as shown in Table 1. Unlike AD-GCL, which focuses solely on positive graph augmentations, our approach optimizes the encoder from both positive and negative perspectives. Notably, JOAO learns to select predefined graph augmentation operations, such as edge/node dropping and subgraph sampling. Our results indicate that the reliance on predefined operations may hinder the learning of meaningful graph patterns. While MVGRL achieves the best performance on the BlogCatalog dataset, it struggles to generalize effectively to other datasets like CORA. On the Co-CS dataset, the enhancement effect is less pronounced, possibly because the Co-CS edges are sparse, and further sampling exacerbates this sparsity. Additionally, ACGA outperforms the non-adversarial baseline (GAUG) across all datasets, highlighting advantages of adversarial augmentation methods.

Link Prediction: Table 2 shows that our ACGA achieves the best performance in link prediction tasks across most datasets. We note that GCA-EV delivers the top performance on both datasets using eigenvector centrality. This may be because link prediction is calculated based on the feature similarity of each node after aggregating information from its neighbors, and eigenvector centrality effectively captures and retains this crucial neighbor information for link prediction. Additionally, we observe a significant drop in performance on the BlogCatalog dataset compared to other datasets. This may be due to bloggers tending to follow individuals from different fields, who may not share high similarity, making link prediction more challenging. Despite this difficulty, model-based methods like ACGA, AD-GCL, and GCA still achieve strong results, demonstrating their robust generalization capabilities.

Ablation Study

We present the results of ablation studies for both node classification and link prediction tasks. To evaluate the importance of various modules, we systematically remove different components, including regularization (NO REG), KL loss (NO KL), and intra-GCL (NO INTRA). Additionally, we highlight the impact of two key components, P-AGA (Eq.2) and N-AGA (Eq.3). Our ablation experiments are conducted on the CORA, CITESEER, and AIR USA datasets. The ablation experiments for the link prediction task are provided in the Appendix.

Table 3 shows that the model's performance drops significantly without regularization, highlighting its importance. This also demonstrates that P-counterfactual and N-counterfactual graphs play a crucial role in generating effective positive and negative graph samples. We further observe that relying solely on positive or negative augmentation is insufficient for optimal graph representation learning (GRL). Positive augmentation primarily drives GRL performance, while negative augmentation provides essential support in the learning process. Additionally, when we replace

Algorithm	Cora	Citeseer	Air-usa	Blogc	Flickr	Pubmed	Co-CS
GCN	81.1±0.7	69.4±1.2	58.7±1.1	75.9±0.3	58.7±0.9	61.7±1.3	60.3±0.4
AD-GCL	82.1±0.8	71.1±0.9	59.1±0.3	75.8±0.7	60.1±0.9	63.1±0.6	60.1±0.8
JOAO	82.2±0.6	70.1±0.8	56.9±1.5	76.7±0.5	58.8±1.0	62.7±1.4	60.9±0.6
GAUG	83.4±0.4	72.3±1.3	59.8±0.7	77.1±0.3	62.3±0.9	64.7±1.1	61.2±0.3
MVGRL	82.2±0.3	71.7±0.3	58.1±0.2	**79.7±0.4**	60.9±0.2	65.2±0.2	60.8±0.2
GCA-EV	82.4±0.3	72.2±0.0	60.2±0.2	77.4±0.1	44.5±0.5	64.6±0.9	59.4±3.5
AFGRL	81.6±0.4	71.6±0.2	59.2±0.5	73.7±2.1	58.1±1.8	63.6±0.6	60.4±1.1
ACGA	**84.4±0.6**	**74.2±0.4**	**61.1±0.4**	77.5±0.8	**64.2±0.6**	**65.5±0.8**	**61.7±0.3**

Table 1: Overall comparisons for node classification tasks (mean ± std. over 10 runs).

Algorithm	Cora		Citeseer		Air-usa		Blogc		Flickr	
	AUC	AP	AUC	AP	AUC	AP	AUC	AP	AUC	AP
GAE	91.4±0.3	92.3±0.4	92.3±0.8	92.5±0.6	92.4±0.5	93.0±0.8	77.0±2.0	76.6±2.1	92.6±0.1	92.4±0.1
AD-GCL	92.2±0.5	92.8±0.5	92.4±0.6	92.9±0.5	93.8±0.8	93.4±0.4	85.6±1.2	85.2±1.2	93.0±0.2	92.8±0.2
JOAO	93.6±0.3	93.8±0.7	93.8±0.5	94.0±0.6	92.8±0.2	93.1±0.1	81.0±2.7	80.6±2.8	93.4±0.1	93.2±0.1
MVGRL	93.5±0.2	92.8±0.3	92.8±0.7	93.1±0.8	92.3±0.4	93.1±0.3	83.2±0.6	83.2±0.6	93.3±0.1	93.4±0.1
GCA-DE	93.1±0.7	93.2±0.4	93.2±0.9	93.8±0.9	93.1±0.3	94.7±0.2	86.0±0.4	85.8±0.4	90.5±0.6	91.1±0.4
GCA-PR	93.1±0.4	93.1±0.4	93.3±1.4	93.9±1.3	93.2±0.3	94.9±0.2	85.8±0.4	85.7±0.4	90.6±0.5	91.2±0.3
GCA-EV	93.0±0.6	93.2±0.4	93.7±1.7	94.5±1.8	92.9±0.2	94.6±0.1	85.9±0.4	**86.2±0.5**	90.3±0.5	91.0±0.3
ACGA	**94.7±0.2**	**94.9±0.2**	**94.8±0.4**	**94.7±0.3**	**94.6±0.1**	**95.3±0.1**	**86.3±0.2**	86.0±0.2	**94.3±0.1**	**94.5±0.2**

Table 2: Overall comparisons for link prediction tasks (mean ± std. over 10 runs).

Algorithm	Cora	Citeseer	Air-usa
N-AGA	71.3±8.3	65.0±1.1	49.1±2.2
P-AGA	82.6±0.7	71.7±0.5	59.8±0.8
RAND GRAPH	82.1±0.8	71.4±0.7	59.3±0.9
NO REG	81.9±0.9	71.3±0.5	59.1±1.1
NO INTRA	82.9±0.8	72.1±0.3	60.1±0.4
NO KL	83.5±0.7	72.5±0.8	60.5±0.8
ACGA	**84.4±0.6**	**74.2±0.4**	**61.1±0.4**

Table 3: Ablation study for node classification

our typical graphs with randomly generated ones (RAND GRAPH), the results confirm that typical graphs are indeed vital for effective augmentations. Moreover, both the KL loss and intra-GRL are critical components of our ACGA. Intra-GRL contributes to more stable performance, while KL loss enhances the diversity of generated graphs, thereby improving the model's generalization ability.

Related Work

Graph Augmentation: Most previous works augment graph samples based on prior rule-based knowledge. For example, DropEdge (Rong et al. 2019) randomly drops a certain number of edges, while AdaEdge (Chen et al. 2020a) adaptively controls between-class and within-class edges. DGI (Hjelm et al. 2018) constructs positive samples by aggregating node representations from the original graph and negative samples by shuffling node order. GMI (Peng et al. 2020) and SUBG-CON (Jiao et al. 2020) propose more efficient GCL methods based on local subgraphs. Recently, model-based methods have garnered significant attention. GAUG (Zhao et al. 2021) introduces a neural edge predictor as an augmentation module. ARIEL (Shengyu Feng and Tong 2021) presents an adversarial augmentation method that regularizes mutual information among positive pairs. AD-GCL (Susheel Suresh and Neville 2021) allows GNNs to capture key information by optimizing adversarial graph augmentation strategies with edge regularization. Both Rep2Vec (Qian et al. 2022) and another work by Wu et al. (Wu et al. 2023) generate positive graphs similar to the original through a discriminator or small perturbations. However, these model-based methods only focus on positive graph augmentations with similar semantic meanings. We have included more related works on graph contrastive learning in the appendix.

Conclusion

In this paper, we proposed a novel adversarial contrastive graph augmentation method for generating both positive samples with minimal sufficient information and challenging negative graph samples. Specifically, we introduced a counterfactual graph approximation method to generate typical positive and negative graphs, which are then used to regularize graph augmentation through a novel Conditional Variational Graph Auto-encoder (CVGAE) framework. Additionally, we incorporated inter-GCL and intra-GCL to extract effective information from both the original and augmented graphs. We also provided a theoretical framework for graph augmentation that unifies the self-supervised processes of positive and negative graph augmentation. In future work, we aim to apply our method to additional tasks, such as graph classification.

Acknowledgments

This work is supported by the National Key R&D Program of China under Grant NO.2022YFA1003900, the National Natural Science Foundation of China under Grants NO.U23B2026, NO.62372305, and NO.62272317, the Guangdong Provincial Key Laboratory of Mathematical Foundations for Artificial Intelligence under Grant NO.2023B1212010001, the Guangdong Basic and Applied Basic Research Foundation under Grants NO.2023A1515110750 and NO.2024B1515040012, the Research Team Cultivation Program of Shenzhen University under Grant NO.2023QNT015, and the Shenzhen Science and Technology Program under Grant NO.JCYJ20220531103402006.

References

Chen, D.; Lin, Y.; Li, W.; Li, P.; Zhou, J.; and Sun, X. 2020a. Measuring and relieving the over-smoothing problem for graph neural networks from the topological view. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, 3438–3445.

Chen, T.; Kornblith, S.; Norouzi, M.; and Hinton, G. 2020b. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, 1597–1607. PMLR.

Hamilton, W.; Ying, Z.; and Leskovec, J. 2017. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30.

Hassani, K.; and Khasahmadi, A. H. 2020. Contrastive multi-view representation learning on graphs. In *International Conference on Machine Learning*, 4116–4126. PMLR.

Hjelm, R. D.; Fedorov, A.; Lavoie-Marchildon, S.; Grewal, K.; Bachman, P.; Trischler, A.; and Bengio, Y. 2018. Learning deep representations by mutual information estimation and maximization. *arXiv preprint arXiv:1808.06670*.

Huang, X.; Li, J.; and Hu, X. 2017. Label informed attributed network embedding. In *Proceedings of the tenth ACM international conference on web search and data mining*, 731–739.

Jiao, Y.; Xiong, Y.; Zhang, J.; Zhang, Y.; Zhang, T.; and Zhu, Y. 2020. Sub-graph contrast for scalable self-supervised graph representation learning. In *2020 IEEE International Conference on Data Mining (ICDM)*, 222–231. IEEE.

Khosla, P.; Teterwak, P.; Wang, C.; Sarna, A.; Tian, Y.; Isola, P.; Maschinot, A.; Liu, C.; and Krishnan, D. 2020. Supervised contrastive learning. *Advances in Neural Information Processing Systems*, 33: 18661–18673.

Kipf, T. N.; and Welling, M. 2016a. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Kipf, T. N.; and Welling, M. 2016b. Variational graph auto-encoders. *arXiv preprint arXiv:1611.07308*.

Lee, J.; Lee, I.; and Kang, J. 2019. Self-attention graph pooling. In *International conference on machine learning*, 3734–3743. PMLR.

Lee, N.; Lee, J.; and Park, C. 2022. Augmentation-free self-supervised learning on graphs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, 7372–7380.

Li, S.; Wang, X.; Zhang, A.; Wu, Y.; He, X.; and Chua, T.-S. 2022. Let invariant rationale discovery inspire graph contrastive learning. In *International conference on machine learning*, 13052–13065. PMLR.

Lin, L.; Chen, J.; and Wang, H. 2022. Spectral augmentation for self-supervised learning on graphs. *arXiv preprint arXiv:2210.00643*.

Pearl, J. 2009. *Causality*. Cambridge university press.

Peng, Z.; Huang, W.; Luo, M.; Zheng, Q.; Rong, Y.; Xu, T.; and Huang, J. 2020. Graph representation learning via graphical mutual information maximization. In *Proceedings of The Web Conference 2020*, 259–270.

Qian, Y.; Zhang, Y.; Wen, Q.; Ye, Y.; and Zhang, C. 2022. Rep2vec: Repository embedding via heterogeneous graph adversarial contrastive learning. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 1390–1400.

Rong, Y.; Huang, W.; Xu, T.; and Huang, J. 2019. Dropedge: Towards deep graph convolutional networks on node classification. *arXiv preprint arXiv:1907.10903*.

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*.

Shengyu Feng, Y. Z., Baoyu Jing; and Tong, H. 2021. Adversarial Graph Contrastive Learning with Information Regularization. In *WWW*. ACM.

Suresh, S.; Li, P.; Hao, C.; and Neville, J. 2021. Adversarial graph augmentation to improve graph contrastive learning. *Advances in Neural Information Processing Systems*, 34: 15920–15933.

Susheel Suresh, C. H., Pan Li; and Neville, J. 2021. Adversarial Graph Augmentation to Improve Graph Contrastive Learning. *Advances in Neural Information Processing Systems*.

Tang, J.; Zhang, J.; Yao, L.; Li, J.; Zhang, L.; and Su, Z. 2008. Arnetminer: extraction and mining of academic social networks. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, 990–998.

Vogel, H. G. 1997. Drug Discovery and Evaluation: Pharmacological Assays.

Wu, C.; Wang, C.; Xu, J.; Liu, Z.; Zheng, K.; Wang, X.; Song, Y.; and Gai, K. 2023. Graph Contrastive Learning with Generative Adversarial Network. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2721–2730.

Wu, T.; Ren, H.; Li, P.; and Leskovec, J. 2020a. Graph information bottleneck. *Advances in Neural Information Processing Systems*, 33: 20437–20448.

Wu, Z.; Pan, S.; Chen, F.; Long, G.; Zhang, C.; and Philip, S. Y. 2020b. A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems*, 32(1): 4–24.

Ying, R.; He, R.; Chen, K.; Eksombatchai, P.; Hamilton, W. L.; and Leskovec, J. 2018. Graph convolutional neural networks for web-scale recommender systems. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, 974–983.

You, Y.; Chen, T.; Shen, Y.; and Wang, Z. 2021. Graph contrastive learning automated. In *International Conference on Machine Learning*, 12121–12132. PMLR.

Yu, J.; Xu, T.; Rong, Y.; Bian, Y.; Huang, J.; and He, R. 2021. Recognizing predictive substructures with subgraph information bottleneck. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Zhang, M.; and Chen, Y. 2018. Link prediction based on graph neural networks. *Advances in neural information processing systems*, 31.

Zhao, T.; Liu, Y.; Neves, L.; Woodford, O.; Jiang, M.; and Shah, N. 2021. Data augmentation for graph neural networks. In *Proceedings of the aaai conference on artificial intelligence*, volume 35, 11015–11023.

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.