# Co-Modality Imbalanced Graph Contrastive Learning

**Anonymous Author(s)**
Affiliation
Address
email

## Abstract

Graph contrastive learning (GCL), leveraging graph augmentations to convert graphs into different views and further train graph neural networks (GNNs), has achieved considerable success on graph benchmark datasets. Yet, there are still some gaps in directly applying existing GCL methods to real-world data. First, handcrafted graph augmentations require expert knowledge as well as trials and errors, but still can not yield consistent performance on multiple tasks. Second, most real-world graph data present imbalanced distribution but existing GCL methods are not immune to data imbalance. Therefore, this work proposes to explicitly tackle these challenges, via a principled framework called *Co-Modality Imbalanced Graph Contratsive Learning with Network Pruning* (**CMI-GCL**) to automatically generate contrastive pairs without expert knowledge and further learn balanced representation over unlabeled data. Specifically, we design inter-modality GCL to automatically generate contrastive pairs (e.g., node-image) based on node content. Inspired by the fact that minority samples can be "forgotten" by pruning deep neural networks, we naturally extend the ad-hoc compression technique, network pruning, to our GCL framework for detecting minority nodes. Hence, we co-train two pruned encoders (e.g., GNN and image encoder) in different modalities by pushing the corresponding node-image pairs together and irrelevant node-image pairs away. Meanwhile, we also propose intra-modality GCL by co-training non-pruned GNN and pruned GNN, to ensure node embeddings with similar attributed features stay closed. By applying pre-trained CMI-GCL to two fine-tuning modes, we demonstrate that our model significantly outperforms state-of-the-art baseline models and learns more balanced representations on real-world graph datasets.

## 1 Introduction

Contrastive learning, aiming to contrast instance pairs generated from unlabeled data to train better representation models, has attracted considerable attention. Inspired by the consistent success of contrastive learning in computer vision and natural language processing [3, 2, 28], an increasing number of works have started to investigate the great potential of contrastive learning on graphs [40, 43, 38, 37]. Most existing graph contrastive learning (GCL) models leverage different graph transformations to augment graph data into different views and further train graph neural networks (GNNs) by maximizing the similarity between positive pairs while minimizing the similarity between negative pairs. In particular, PyGCL [43] summarizes nine categories of graph augmentations from different levels (e.g., node dropping, edge perturbation, and graph diffusion), and then implements different augmentation combinations to transform graph into different views for contrastive learning.

Yet, directly applying the aforementioned GCL models from the controlled benchmark graph datasets to uncontrolled real-word graph datasets still has some gaps. First, notwithstanding suitable handcrafted augmentation methods need professional knowledge as well as trials and errors, the consistent excellent performance on multiple tasks can not be guaranteed [43]. Second, node classes in most

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

real-world graphs present imbalanced distribution (e.g., Zipf long-tail distribution). For instance, the amount of illicit activities (e.g., drug trafficking, malicious repository, and fake review) on social networks is much less than that of normal activities [33]. To tackle these challenges, we aim to design a novel GCL model that automatically generates contrastive pairs without too much efforts, but also can generalize well in real-world imbalanced graph.

Accordingly, inspired by existing cross-modality contrastive learning methods in computer vision [32], we naturally hypothesize that node embedding in graph and the corresponding content embedding (e.g., image embedding) in different modality will be closer if we co-train GNN encoder and content encoder (e.g., image encoder) during contrastive learning. In this way, we can automatically generate contrastive pairs, e.g., (node-image), based on the natural content on nodes, which is called inter-modality GCL. In addition, existing works signal that contrastive loss can guarantee a balanced representation space [39, 16]. However, a recent study [15] demonstrates that contrastive learning can alleviate the data imbalance problem but fails to be fully immune to the data imbalance issue. Motivated by the fact that minority data may be easily "forgotten" by pruning deep neural networks [14], we extend the ad-hoc compression tool, network pruning, to GCL framework and further detect minority samples not yet well represented by encoders during co-modality GCL.

To summarize, we propose a novel framework called *co-modality imbalanced graph contrastive learning with network pruning* (**CMI-GCL**) to automatically generate contrastive pairs from unlabeled data and further learn more balanced node representations. As illustrated in Figure 1, we first prune the graph encoder and content encoder in inter-modality GCL to detect minority samples, and further co-train two encoders via optimizing contrastive loss. Besides, we also aim to ensure that node embeddings with similar attributed features should also stay closed. Therefore, we propose intra-modality GCL that generates contrastive pairs based on the similarities among node attribute features and further co-trains non-pruned graph encoder and pruned graph encoder. By the aforementioned steps, CMI-GCL can boost minority nodes weights in contrastive loss and leads to implicit loss re-balancing. To conclude, our work makes the following contributions:

- As handcrafted graph augmentations are not efficient on real-world data, we devise a co-modality framework to automatically generate contrastive pairs. To our best knowledge, this is the first work on GCL that co-trains encoders in different modalities to facilitate node representation learning.
- To address the data imbalance issue on graphs, our model extends the compression technique, network pruning, to prune encoders during co-modality GCL for detecting minority samples and further learns more balanced node representations.
- Extensive experiments on real-world graph datasets show the effectiveness of our model by comparing with state-of-the-art methods.

## 2 Related Work

**Graph Contrastive Learning.** Most existing GCL models [31, 40, 38] leverage different types of data transformations to augment graph into different views and further train an encoder by discriminating positive pairs and negative pairs generated from unlabeled data. Specifically, GraphCL [40] designs four types of graph augmentations to generate contrastive pairs in different views for contrastive learning. GCC [31] implements random walk for each node to sample subgraph as augmentation. In heterogeneous domain, HeCo [37] employs cross-view (schema view and meta-path view) contrastive learning to train an encoder. These handcrafted data augmentations require trials and errors, and sufficient domain knowledge, but still cannot yield consistent excellent performance on multiple tasks [40]. Hence, we naturally propose to automatically generate contrastive pairs for graph data. Inspired by CLIP [32], a co-modality contrastive learning model on image and text data for image caption generation, we devise cross-modality GCL which leverages the rich content in graph to automatically generate useful contrastive pairs without resorting to handcrafting or expert knowledge.

**Data Imbalance in Self-supervised Learning.** Data imbalance is very common in real-world applications (e.g., anomaly detection [5, 7]). Most existing methods against data imbalance focus on supervised learning (e.g., re-sampling [1, 25], re-weighting [17, 35], and GraphSMOTE [42]). Some recent works [39, 16, 15] start to explore the benefits of the balanced feature space from self-supervised learning. Yang and Xu [39] first studied the label bias in self-supervised learning and concluded that classifiers that are pre-trained in a self-supervised manner consistently outperform their corresponding supervised baselines. Kang et al. [16] found that contrastive learning can present

a more balanced feature space over imbalanced datasets by comparison with supervised learning. Based on this, SDCLR [15] thinks contrastive learning is not fully immune to data imbalance. All aforementioned works study the imbalance problem on image. Motivated by these works, we devise an algorithm that can well mitigate the data imbalance in our proposed co-modality GCL framework.

**Network Pruning.** Network pruning [20, 23, 21] has been considered as a popular compression tool for DNN models. In recent, some works [10, 14, 15] begin to explore the deeper connection with DNN memorization and generalization. Specifically, Frankle and Carbin [10] found that pruning a dense DNN can identify "significant subnetwork" that testing accuracy is comparable to the one in original network. Hooker et al. [14] thought a small data subset, i.e., data belonging to ambiguous classes and minority classes, are more likely to be impacted by the introduction of sparsity via network pruning. Based on the conclusion, Jiang et al. [15] constructed a self-competitor via network pruning to study imbalanced image in contrastive learning. In this work, we introduce network pruning in co-modality contrastive learning to detect minority nodes and further gain more imbalanced representation.

# 3 Preliminary

**Graph Neural Networks.** Given a graph $G = (\mathcal{V}, \mathcal{E}, \mathcal{X})$, where $\mathcal{V}$ is the set of different types of nodes, $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the set of edges, and $\mathcal{X}$ is the attributed feature set. We aim to learn the node representation by considering both structure and node attributed features. Current GNNs (e.g., GCN [19], GAT [36], and GraphSage [11]) follow a messaging-passing framework and has gained consistent performance on various graph data. In this paper, we choose GCN as the graph encoder to learn the node embedding $h_G^i \in \mathbb{R}^{d_G}$ corresponding to node $v_i$. In particular, GCN is formulated as $H^{l+1} = \sigma(\widetilde{A}\, H^l\, W^l)$, where $H^{l+1}$ denotes the node representations at $l+1$ layer, $\widetilde{A}$ is a symmetric normalization of $A$ with self-loop, $W^l$ is the weight matrix at $l$-th layer, and $\sigma$ is the activation function. For simplicity, we use $f_G$ to denote the graph encoder and $h_G$ to denote the node embedding. Besides GCN, we also discuss GAT and GraphSage as graph encoder in experiments.

**Text Pre-Trained Model.** Existing graph representation learning methods [9, 41, 29] generally implement the pre-trained language model (e.g., BERT [6]) to extract features from text content as the attributed feature for each node. Instead of this, we first adopt DistillBERT [34] (a distilled version of BERT) as the pre-trained text encoder to obtain the attributed features for each node, and then fine-tune DistillBERT with contrastive loss. The structure of DistillBERT we implemented is a base size with a 12-layer 512-wide model along with 8 attention heads. Text sequence is bracketed with [SOS] and [EOS] tokens and we take the activation of the highest layer of transformer at [EOS] token as text representation $h_T \in \mathbb{R}^{d_T}$. We utilize $f_T$ to denote text encoder. After fine-tuning $f_T$ via optimizing co-modality contrastive loss, we update fine-tuned attributes for all nodes. Besides DistillBERT, we also discuss BERT as the text encoder and more details can be found in Section 5.5.

**Image Pre-Trained Model.** We implement two different architectures (i.e., ResNets and Transformer) as image encoder. As ResNets is widely used and has been proved with excellent performance, we implement ResNet50 [12] as the base architecture. Following [13], we make the similar modifications on ResNet and replace the global average pooling layer with an attention pooling mechanism. Besides, we consider the recently introduced Swin Transformer [24] as the image encoder and closely follow their implementation. In this paper, with the consideration of performance and efficiency (as discussed in Section 5.5), we take Swin Transformer as the image encoder denoted as $f_I$ to obtain the image representation $h_I \in \mathbb{R}^{d_I}$. After training $f_I$ via contrastive loss optimization, we attach the obtained image embedding to the corresponding nodes as attributed features for downstream tasks.

**Problem 1.** ***Co-Modality Imbalanced Graph Contrastive Learning.*** *Given a graph $G = (\mathcal{V}, \mathcal{E}, \mathcal{X})$ built on multi-modality data along with imbalanced label $Y$, we aim to build a GCL model over $G$ that can be further fine-tuned or applied to the downstream tasks with imbalanced labels (e.g., imbalanced node classification).*

# 4 The Proposed Model

In this section, we present the details of CMI-GCL (Figure 1) which includes three steps: (i) graph construction (Figure 1.(a)); (ii) co-modality imbalanced graph contrastive learning with network pruning (Figure 1.(b)); (iii) two fine-tuning modules for downstream tasks(Figure 1.(c)).

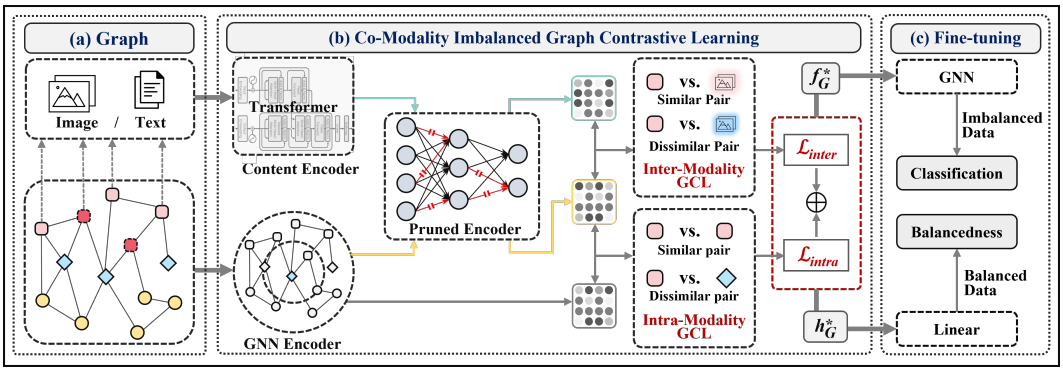


Figure 1: The overall framework of CMI-GCL: (a) It constructs graph to depict content data (e.g., image and text) and relation data in networks; (b) It co-trains pruned GNN and pruned content encoder in inter-modality GCL to ensure the embeddings of node and content referring to the same entity stay together; Intra-modality GCL that co-trains non-pruned GNN and pruned GNN is further proposed to ensure embeddings of nodes with similar attributes stay closed. (c) The pre-trained encoder $f_G^*$ and embedding $h_G^*$ from (b) are fine-tuned to downstream tasks.

### 4.1 Graph Construction

Graph has been proved to be effective in modeling real-world networked data (e.g., social network). Thus, to model real-world data for certain tasks (e.g., malware detection on GitHub data), we introduce graph to depict both content and structure information in the data. In this work, by following previous studies [30, 29] that provides GitHub and Instagram graph datasets, we introduce four types of entities (i.e., user, repository, file, and keyword) and seven types of relations in GitHub graph; three types of entities (i.e., user, post, and keyword) and eight types of relations in Instagram graph. In addition, we implement the pre-trained language model DistillBERT [34] and Swin Transformer [24] to encode the text and image respectively, and apply them to the corresponding nodes as feature attributes. Different from previous works, we use original content (e.g., image and text) to facilitate contrastive pairs generation and further co-train graph encoder and content encoder via contrastive learning.

### 4.2 Co-Modality Imbalanced Graph Contrastive Learning

After graph construction, we first propose inter-modality contrastive learning to automatically generate contrastive pairs. Then we introduce network pruning to detect minority nodes during contrastive learning. Last, we design intra-modality contrastive learning to ensure that node embeddings with similar semantics (attributed features) stay closed.

#### 4.2.1 Inter-Modality Graph Contrastive Learning

The main idea of contrastive learning is to make representations maximize the agreement on sample pairs under proper graph augmentations. Yet, finding proper graph augmentation for every task is always resource-consuming [40]. Inspired by existing work [32] of multiple-modality learning, we propose to fully utilize nodes' content to automatically generate contrastive pairs, and further co-train encoders in different modalities via contrastive learning. That is to say, we aim to learn two encoders such that embeddings in two modalities are close to each other in the learned space if they refer to the same entity, otherwise far away. In this paper, we consider two combinations of modalities for graphs constructed in Section 4.1 (i.e., graph-image for Instagram graph, graph-text for GitHub graph). For simplicity, we discuss graph-image modalities as an example in this section, and will also conduct experiments on graph-text modalities. Specifically, given a sample pair $(x_G^i, x_I^i)$, where $x_G^i$ is the attributed feature of node $v_i \in \mathcal{V}$ and $x_I^i$ is the image input corresponding to node $v_i$, we feed $x_G^i$ to graph encoder $f_G$ and $x_I^i$ to image encoder $f_I$ respectively and further obtain the corresponding embedding $(h_G^i, h_I^i)$. Note that, as different types of nodes in $G$ may have different feature dimensions, we first transform the input attributed features of all nodes to a common space. After getting the embedding pair $(h_G^i, h_I^i)$, we apply nonlinear projection heads to convert embedding from different modalities to the same space (denoted as $(z_G^i, z_I^i)$) for comparison. We consider $(z_G^i, z_I^i)$ as a similar pair (positive pair) if the node and the image refer to the same entity. Otherwise, $(z_G^i, z_I^i)$ is viewed as a dissimilar pair (negative pair).

#### 4.2.2 Network Pruning for Imbalance Data

Whether contrastive learning is immune to data imbalance or not has remained controversial. Recent study [15] claim that contrastive learning can alleviate the data imbalance issue in representation learning but is not fully immune to data imbalance. To validate that, we define a metric to describe the balanced property of a feature space. Typically, the linear separability is evaluated by the accuracy of a linear classifier over representations. A feature space $U$ can be considered as balanced if the representation $u_i$ from different classes has similar degrees of linear separability. Typically, the linear separability is evaluated by the accuracy of a linear classifier over representations. Specifically, we utilize $a_1, a_2, ..., a_c$ to denote the classification accuracy of a linear classifier over representation $u_i \in U$ from $c$ classes. Then the balancedness metric $\mathcal{B}$ of feature space $U$ is defined as:

$$\mathcal{B}(U) = \frac{1}{c}\sum_{j,k}^{c} \exp\left(-\frac{|a_j - a_k|^2}{\xi}\right), \text{where } a_k = \frac{\#\{u_i|\hat{y}_i = y_i = k, u_i \in U\}}{\#\{u_i|y_i = k, u_i \in U\}}, \tag{1}$$

where $\xi$ is a scaling hyper-parameter, and $\hat{y}$ is the classification result. $\mathcal{B}(U)$ achieves its maximum when all class-wise accuracy are equal, which means there is no separability bias of learned representations to any class. We conduct experiments with $\mathcal{B}(U)$ in Section 5.2.2 and further conclude that existing GCL will not be able to produce balanced representations if data is heavily imbalanced.

Based on the above finding, we propose to introduce a method to address the data imbalance issue during co-modality GCL. The recent work [14] finds that certain examples are particularly sensitive to sparsity via network pruning and these samples mostly impacted after pruning are termed as Pruning Identified Exemplars (PIEs). They further demonstrate that PIEs often show up at the minority classes. Inspired by this, we extend network pruning to our proposed framework to detect minority samples. With PIEs dynamically generated during contrastive training, more minority samples will be exposed along with training epochs, as we will show in Figure 2 of experiments. We implement magnitude pruning method that uses absolute value of weights to rank their importance and removes weights that are below a certain threshold. Specifically, the detailed process is described as follows:

1. Randomly initialize encoders $f_G$ and $f_I$ with parameters $\theta_G^0$ and $\theta_I^0$ respectively.
2. At each iteration $l$, prune $e\%$ of smallest-magnitude parameters in $\theta_G^l$ and $\theta_I^l$, creating the corresponding masks $m_G^l$ and $m_I^l$.
3. Apply masks $m_G^l$ and $m_I^l$ to the feed-forward encoders and calculate the contrastive loss.
4. Release masked weights and update encoders' parameters via optimizing the contrastive loss.

In the above steps, the contrastive loss in inter-modality GCL is formulated as:

$$\mathcal{L}_{\text{inter}} = -\log \sum_{v_i \in \mathcal{V}} \frac{\exp\left[\text{sim}\left(\widetilde{z}_G^i, \widetilde{z}_I^i\right)/\tau_{\text{inter}}\right]}{\sum_{p=1}^{2n} \mathbb{1}_{[i \neq p]} \exp\left[\text{sim}\left(\widetilde{z}_G^i, \widetilde{z}_I^p\right)/\tau_{\text{inter}}\right]}, \tag{2}$$

where $(\widetilde{z}_G^i, \widetilde{z}_I^i)$ is the projected embedding generated by pruned encoders $\widetilde{f}_G$ and $\widetilde{f}_I$, $\text{sim}(\widetilde{z}_G^i, \widetilde{z}_I^i)$ is the cosine similarity between $\widetilde{z}_G^i$ and $\widetilde{z}_I^i$, $\tau_{\text{inter}}$ is the temperature value, and $n$ is the size of mini-batch.

Following the iterative pruning process that repeatedly trains, prunes and releases weights in each round, it effectively boosts minority samples' weights in $\mathcal{L}_{\text{inter}}$ and leads to implicit loss re-balancing.

#### 4.2.3 Intra-Modality Graph Contrastive Learning

Inter-modality graph contrastive learning ensures that embedding from different modalities w.r.t. the same entity stay closed in the projected space but fails to ensure that embedding with similar original features from the same modality stay closed. We believe nodes with similar attributed features naturally have similar semantics and their embedding should be closer, while most existing GNNs are mainly dominated by relation among nodes [29]. In light of this, we propose intra-modality graph contrastive learning to balance the importance of local neighbor relations and attributed features, and further ensure that nodes with similar attributed features will stay closed in the projected space. Note that, in intra-modality graph contrastive learning, we mainly focus on graph modality as we believe features among nodes have rich semantics. Specifically, we first define $\mathcal{S}_t$ and $\mathcal{S}_o$ as sets of node pairs having the top-R similarity and the bottom-R similarity respectively, which is formally defined as:

$$\begin{aligned} \mathcal{S}_t &= \{(v_i, v_p) \mid \text{sim}(x_G^i, x_G^p) \text{ in top-R of } [\text{sim}(x_G^i, x_G^p)]_{p=1}^N, \ \forall v_i \in \mathcal{V}\} \\ \mathcal{S}_o &= \{(v_i, v_p) \mid \text{sim}(x_G^i, x_G^p) \text{ in bottom-R of } [\text{sim}(x_G^i, x_G^p)]_{p=1}^N, \ \forall v_i \in \mathcal{V}\}, \end{aligned} \tag{3}$$

where $i \neq p$, $\text{sim}(x_G^i, x_G^p)$ measures the cosine similarity of node attributes between node $v_i$ and $v_p$, $R$ is the number of top/bottom pairs chosen for each node, and $N$ is the number of nodes in $G$.

The main difference between intra-modality GCL and inter-modality GCL lies in that, intra-modality GCL implements one graph encoder and another pruned graph encoder for contrastive training, while inter-modality GCL implements one graph encoder and one image (or text) encoder for co-modality contrastive training. Hence, we have a dense graph encoder $f_G$ and a sparse graph encoder $\widetilde{f}_G$ pruned by iterative magnitude pruning. Given a sample pair $(x_G^i, x_G^p)$, it will be encoded by $f_G$ and $\widetilde{f}_G$ respectively. The projected embeddings $(z_G^i, \widetilde{z}_G^p)$ is regarded as a positive pair if $(v_i, v_p)$ in $\mathcal{S}_t$, while it is a negative pair if $(v_i, v_p)$ in $\mathcal{S}_o$. The intra-modality GCL loss is defined as follows:

$$\mathcal{L}_{\text{intra}} = -\log \sum_{v_i \in \mathcal{V}} \frac{\sum_{(v_i, v_p) \in \mathcal{S}_t} \exp\left[\text{sim}\left(\widetilde{z}_G^i, \widetilde{z}_G^p\right)/\tau_{\text{intra}}\right]}{\sum_{(v_i, v_q) \in \mathcal{S}_o} \exp\left[\text{sim}\left(\widetilde{z}_G^i, \widetilde{z}_G^q\right)/\tau_{\text{intra}}\right]}. \tag{4}$$

By this way, for atypical and minority nodes, our model can amplify the prediction differences between the pruned and non-pruned graph encoders in intra-level, and further these nodes' weights will be implicitly increased in the intra-modality contrastive loss $\mathcal{L}_{\text{intra}}$.

After performing inter-modality pruning contrastive learning and intra-modality pruning contrastive learning, the overall objective of co-modality pruning contrastive learning can be formulated as:

$$\mathcal{L} = \lambda \mathcal{L}_{\text{inter}} + (1 - \lambda) \mathcal{L}_{\text{intra}}, \tag{5}$$

where $\lambda$ is the trade-off hyper-parameter for balancing two loss terms. Pseudo-code of CMI-GCL is provided in Section A of Appendix.

### 4.3 Fine-tuning

Inspired by GCC [31] we introduce two fine-tuning strategies, i.e., full fine-tuning mode for downstream tasks and freezing mode for balancedness (linear separability) evaluation.

**Full fine-tuning.** In full fine-tuning mode, we adopt a multi-step protocol to evaluate the classification performance. (i) Representation learning via CMI-GCL: we pre-train encoders by optimizing the pre-training loss $\mathcal{L}$. (ii) Fine-tuning : we adopt the pre-trained weights from $f_G^*$ (pre-trained graph encoder) as the initialization of the fine-tuning encoder and update the attributed features generated by $f_I^*$ (pre-trained image encoder). Then we train the fine-tuning encoder together with the classifier (i.e., MLP) using focal loss, a specific loss for imbalanced classification [22]. (iii) Classification evaluation: we evaluate the classification performance over imbalanced nodes with different imbalance ratios.

**Freezing.** In freezing mode, we also adopt a three-stage protocol to evaluate the balancedness of learned representations. (i) Representation learning: similar to full fine-tuning mode, we obtain $f_G^*$ via CMI-GCL; (ii) Classifier learning with pre-trained embedding: we freeze parameters of $f_G^*$, and further train a linear classifier $\hat{y} = (w^\top h_G^* + \epsilon)$ with the supervised cross-entropy loss. Here $w$ denotes the classifier weight matrix, $\epsilon$ denotes the bias term, $h_G^*$ is the pre-trained node embedding. (iii) Representation evaluation: we evaluate the classification accuracy on $h_G^*$ and further calculate the proposed balancedness metric $\mathcal{B}(U)$.

## 5 Experiments

In this section, we first introduce datasets that we use to evaluate CMI-GCL and baselines models. Then we provide the detailed analysis to show the effectiveness of CMI-GCL and its strong applicability to real-world graph datasets. More about experiments, i.e., implement details, hyper-parameter sensitivity, and ablation study with different imbalanced ratios are provided in Appendix.

### 5.1 Experimental Setup

#### 5.1.1 Dataset

In this paper, we adopt two multi-modality graph datasets from existing works, i.e., GitHub graph [29] and Instagram graph [30], which include raw content and graph information on these two platforms. Specifically, GitHub graph provided by [29] aims to detect malicious repositories, which contains

all content of repositories, users, and files on GitHub. The GitHub graph consists of 174,661 nodes (i.e., 20,895 repository nodes, 52,530 keyword nodes, 82,451 file nodes, and 18,785 user nodes) and 696,464 edges among nodes. Instagram graph provided by [30], having all image and text information about users and posts, aims to detect illicit drug traffickers on Instagram. The Instagram graph contains 129,894 nodes (i.e., 8,651 user nodes, 79,705 post nodes, 41,538 keyword nodes) and 218,039 edges among nodes. Both datasets have two types of node labels (i.e., 6,965 malicious and 13,930 benign repositories on GitHub, 3,242 drug traffickers and 5,409 normal users on Instagram).

#### 5.1.2 Baseline Method

To evaluate the performance of CMI-GCL, we implement two tasks, i.e., binary classification over imbalanced data (Task 1) and balancedness evaluation over balanced data (Task 2). Specifically, for Task 1, we compare CMI-GCL with fourteen baseline methods which are divided into four groups (Table 1): feature-based method (G1), graph learning models (G2), GCL models (G3), and methods against imbalanced data (G4). For G1, we feed feature vector to a two-layer MLP [26]. For G2, we implement seven graph learning methods, i.e., Deepwalk [27], mp2vec [8], GCN [19], GAT [36], and GraphSage [11], to learn the node representations. Meta-AHIN [29] is the work that provides the GitHub data and develops a graph model to detect malicious repositories. MetaHG [30] is the work that provides the Instagram data and develops a meta-learning model to detect illicit drug traffickers. For G3, we implement three popular approaches, i.e., over-sampling [18], re-weighting [4], and GraphSMOTE [42], to handle the imbalanced node representations generated by GCN. GraphSMOTE is the extension of SMOTE [1] on graph data, which leverages node similarities to generate synthesized nodes and edges. For G4, we reproduce three GCL methods, i.e., GCC [31], GraphCL [40], and HeCo [37]. To compare them with CMI-GCL fairly, we fine-tune the pre-trained models with Focal loss as well. Besides, for Task 2, we compare CMI-GCL with SimCLR [2] (CMI-GCL without pruning), GCC, GraphCL, and HeCo to evaluate the balancedness of representations (Table 2). Details of baseline settings are described in Section B of Appendix.

#### 5.1.3 Experimental Settings

All experiments are conducted under the environment of the Ubuntu 16.04 OS, plus Intel i9-9900k CPU, two GeForce GTX 2080 Ti Graphics Cards, and 64 GB of RAM. We train all methods with a fixed epoch (1000) and use the early stopping strategy with a patient of 50. Besides, all methods are trained ten times, and the average performance multiplied by 100 on testing data is reported. We use 70% samples for training, 10% for validation, and the remaining 20% for testing. Following existing works in evaluating imbalanced classification [1, 42], we adopt two metric, i.e., F-score (F1) and AUC-ROC score (AUC) to evaluate all models. Besides, we define $\mathcal{B}(U)$ in Eq. 1 to evaluate the balancedness of representations learned from GCL. In this paper, the initial imbalance ratio $\beta$ of repository data and user data are 0.5 (1:2) and 0.6 (1:1.7), respectively. To evaluate model performance in more imbalanced scenarios, we set $\beta$ as 0.1 and 0.01 and further utilize $D_{\text{init}}$, $D_{\text{b}}$, $D_{0.1}$, and $D_{0.01}$ to denote dataset with initial imbalance ratio, 1.0, 0.1, and 0.01, respectively. Note that for Task 1, we leverage $D_{\text{init}}$ to train CMI-GCL and fine-tune the pre-trained encoder over $D_{\text{init}}$, $D_{0.1}$, and $D_{0.01}$ to get the classification results. For Task 2, we leverage $D_{\text{init}}$, $D_{\text{b}}$, $D_{0.1}$, and $D_{0.01}$ to train all GCL models respectively, and further apply the pre-trained embeddings as features to a linear classifier with $D_{\text{b}}$. When pre-train CMI-GCL, with the grid search, the scaling parameter $\xi$ in $\mathcal{B}(U)$ is set as 0.01, pruning ratio $e$ is set as 20%, the number of contrastive pairs for each node $R$ in intra-modailty GCL is set as 10, and $n$ for each mini-batch is set as 200. Besides, the temperature parameter $\tau_{\text{inter}}$ and $\tau_{\text{intra}}$ are set as 1.0 and the trade-off hyper-parameter $\lambda$ among co-modality GCL is set as 0.6. For fine-tuning, $\alpha$ and $\gamma$ in Focal loss [22] are set as 1.0 and 2.0, respectively.

### 5.2 Performance Comparison

#### 5.2.1 CMI-GCL Helps Downstream Tasks with Imbalanced Classes

Table 1 shows the performances of all models on GitHub and Instagram graphs with different imbalance ratios. The best performance are highlighted in bold and the best baseline results are emphasized by underline. In Table 1, we can conclude that: (i) Merely considering content data (G1) or relations among nodes (Deepwalk and mp2vec in G2) is not supportive to learn node representations in imbalanced graphs. Integrating rich content and relations in graph can learn better representations. (ii) Leveraging unlabeled data to train graph encoder via contrastive learning can enhance node

Table 1: Performance comparison of all methods on imbalanced datasets with different ratios.

| Setting | | GitHub (Node ⇔ Text) | | | | | | Instagram (Node ⇔ Image) | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | $D_{0.01}$ | | $D_{0.1}$ | | $D_{\text{init}}$ | | $D_{0.01}$ | | $D_{0.1}$ | | $D_{\text{init}}$ | |
| Group | Model | F1 | AUC | F1 | AUC | F1 | AUC | F1 | AUC | F1 | AUC | F1 | AUC |
| G1 | Feature+MLP [26] | 23.88 | 57.91 | 64.92 | 79.69 | 66.74 | 78.21 | 4.35 | 51.04 | 36.02 | 62.71 | 66.77 | 76.83 |
| G2 | Deepwalk [27] | 25.47 | 73.49 | 66.71 | 82.54 | 68.73 | 84.25 | 6.25 | 52.87 | 38.29 | 72.34 | 68.24 | 82.42 |
| | mp2vec [8] | 27.15 | 75.34 | 67.24 | 83.35 | 69.55 | 85.52 | 8.57 | 54.61 | 39.64 | 75.37 | 70.52 | 84.38 |
| | GCN [19] | 32.33 | 80.09 | 69.15 | 86.85 | 72.55 | 88.28 | 12.51 | 58.71 | 42.62 | 81.45 | 72.55 | 87.28 |
| | GAT [36] | 32.51 | 79.25 | 69.42 | 86.25 | 73.27 | 88.24 | 12.58 | 58.91 | 42.32 | 81.98 | 72.32 | 87.12 |
| | GraphSAGE [11] | 29.41 | 77.87 | 66.14 | 84.25 | 70.48 | 86.82 | 10.29 | 56.27 | 40.54 | 79.27 | 71.24 | 85.39 |
| | Meta-AHIN [29] | 34.15 | 81.53 | 69.98 | 87.45 | 73.46 | 89.52 | 13.74 | 59.45 | 43.15 | 82.53 | 74.57 | 88.58 |
| | MetaHG [30] | 35.55 | 81.78 | 70.24 | 87.29 | 73.98 | 90.05 | 14.25 | 59.54 | 44.07 | 82.74 | 74.98 | 88.98 |
| G3 | Over-sampling [18] | 35.73 | 82.97 | 71.64 | 89.76 | 73.18 | 90.99 | 15.28 | 60.24 | 44.11 | 83.83 | 73.05 | 88.68 |
| | Re-weighting [4] | 37.77 | 84.66 | 71.78 | 91.48 | 74.12 | 92.84 | 17.39 | 62.75 | 45.87 | 84.64 | 73.83 | 89.07 |
| | GraphSMOTE [42] | 40.59 | 85.21 | 72.59 | 92.54 | 74.51 | 92.98 | 19.52 | 65.24 | 46.52 | 85.36 | 73.45 | 90.13 |
| G4 | GCC+Focal [31] | 39.25 | 83.58 | 71.29 | 90.48 | 78.25 | 92.14 | 18.12 | 63.25 | 46.12 | 84.86 | 76.51 | 90.24 |
| | GraphCL+Focal [40] | 40.48 | 84.27 | 73.15 | 91.48 | 79.34 | 93.25 | 18.42 | 64.28 | 47.57 | 86.16 | 77.41 | 91.64 |
| | HeCo+Focal [38] | 39.68 | 83.84 | 72.26 | 90.76 | 78.87 | 92.94 | 18.17 | 63.59 | 46.52 | 85.29 | 76.98 | 90.82 |
| Ours | **CMI-GCL** | **45.89** | **88.51** | **76.35** | **95.27** | **83.35** | **96.48** | **22.47** | **67.34** | **51.24** | **89.14** | **80.71** | **94.25** |

representations (G3). Note that all models in G3 are based on the embeddings generated from a two-layer GCN. (iii) Some popular methods against imbalanced data in supervised learning can alleviate the influence of imbalance and further improve the performance to some extent. However, they fail to handle heavily imbalanced data ($D_{0.01}$). (iv) CMR-CLR has the best performance when dealing with heavily imbalanced data. By comparing CMR-CLR with GCL models in G4, we can conclude that CMR-CLR can learn better representations for real-world imbalanced graphs.

### 5.2.2 CMI-GCL Improves Balancedness on Imbalanced Data

We now investigate if contrastive learning is vulnerable to data imbalance. Based on Eq. 1, the maximum value of $\mathcal{B}(U)$ is 0.5 in this paper when the accuracy of the two classes is the same. That is to say, the classification doesn't have any imbalance. As shown in Table 2, we can conclude that: (i) For both datasets, models pre-trained on $D_b$ show larger balancedness than imbalanced subsets. (ii) The balancedness is worse when the imbalance ratio $\beta$ is smaller. Even so, contrastive learning can still alleviate the influence of data imbalance if the data is not heavily imbalanced (i.e., $D_{\text{init}}$ and $D_{0.1}$). (iii) GCL models become fragile when trained over a heavily imbalanced dataset ($D_{0.01}$), which validates the necessity of addressing the imbalance problem in GCL. (iv) CMI-GCL can learn more balanced representations over heavily imbalanced data, which demonstrates the effectiveness of CMI-GCL in learning balanced representations.

Table 2: Balancedness comparison of contrastive models learned on $D_b$, $D_{\text{init}}$, $D_{0.1}$, and $D_{0.01}$.

| Dataset | Ratio | CMI-GCL | SimCLR | GCC | GraphCL | HeCo |
|---|---|---|---|---|---|---|
| GitHub | $D_{\text{b}}$ | 49.51 | 49.42 | 49.22 | **49.57** | 49.40 |
| | $D_{\text{init}}$ | **49.45** | 48.47 | 47.91 | 48.32 | 48.47 |
| | $D_{0.1}$ | **48.52** | 46.21 | 45.23 | 46.14 | 46.27 |
| | $D_{0.01}$ | **47.52** | 38.60 | 37.71 | 38.45 | 38.52 |
| Instagram | $D_{\text{b}}$ | 49.49 | 49.43 | 49.01 | **49.51** | 49.10 |
| | $D_{\text{init}}$ | **49.18** | 48.91 | 48.52 | 48.92 | 48.66 |
| | $D_{0.1}$ | **46.31** | 44.52 | 43.51 | 44.66 | 43.80 |
| | $D_{0.01}$ | **44.47** | 34.61 | 32.19 | 34.68 | 33.40 |

## 5.3 Ablation Study

To show the effectiveness of different components in CMI-GCL, we conduct a set of ablation experiments over subset $D_{0.1}$ and further analyze the contribution of each component (i.e., co-modality GCL, pruning, and intra-modality GCL) by removing it separately. First, we remove co-modality GCL from CMI-GCL, which means we directly employ a two-layer GCN to learn the node embedding for both datasets. As shown in Table 3, we can conclude that our model is effective as the performances on both datasets drop significantly. In addition, we re-

Table 3: Comparison among model variants of CMI-GCL on $D_{0.1}$.

| Dataset | Method | F1 | AUC |
|---|---|---|---|
| GitHub | CMI-GCL | **76.35** | **95.27** |
| | - Co-Modality | 69.15 | 86.85 |
| | - Pruning | 74.04 | 93.08 |
| | - Intra-Modality | 74.69 | 93.74 |
| Instagram | CMI-GCL | **51.24** | **89.14** |
| | - Co-Modality | 42.62 | 81.45 |
| | - Pruning | 48.53 | 87.18 |
| | - Intra-Modality | 49.25 | 87.69 |

move network pruning and intra-modality on CMI-GCL. We find the performance on both datasets decreases obviously, validating the effectiveness of network pruning and intra-modality in CMI-GCL.

### 5.4 CMI-GCL Mines More Samples from Minority Class

We further investigate whether network pruning is effective to detect minority samples during contrastive learning. Based on the pruning steps introduced in Section 4.2.2, we measure the distribution of PIEs mined by CMI-GCL. Specifically, when pre-train on different subsets (i.e., $D_{\text{init}}$, $D_{0.1}$, $D_{0.01}$), we sample top 1% training data that are most easily affected by pruning, and further calculate the percentages of PIEs that belong to the minority class. As shown in Figure 2, minority samples in $D_{0.1}$ and $D_{0.01}$ are more easily to be detected by CMI-GCL along with the training epochs. Notwithstanding the minority samples in $D_{0.01}$ are less likely to be detected at the beginning, it shows a general increasing trend along with training epochs. $D_{\text{init}}$ keeps a relatively low percentage during the pre-training process. Hence, we again demonstrate that network pruning in CMI-GCL can detect minority samples and further learns more balanced representations.

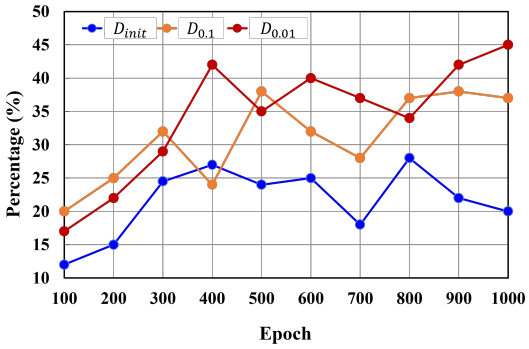


Figure 2: The percentage of PIEs that belong to minority classes under different training epochs.

### 5.5 CMI-GCL Are Applicable to Mainstream Models in Different Modalities

Our proposed CMI-GCL aims to study various real-world graphs: text-based graph (e.g., GitHub), image-based graph (e.g., Instagram). To this end, CMI-GCL is designed to be a plug-and-play tool which is applicable to most mainstream models in different modalities. Hence, we adopt at least two models in each modality (i.e., graph: GCN, GAT, and GraphSAGE (SAGE); text: BERT and DistillBERT (DBERT); image: ResNet50 (ResNet) and Swin Transformer (ST)) to show the strong applicability and the effectiveness of CMI-GCL. Figure 3 shows the model size and the performance of encoders (i.e., F1) on Task 1 with dataset $D_{\text{init}}$. We find that all combinations have excellent performances on imbalanced dataset. Due to the efficiency and resource concerns, We choose GCN as the graph encoder, Swin Transformer as the image encoder, and DistillBERT as the text encoder in this paper.

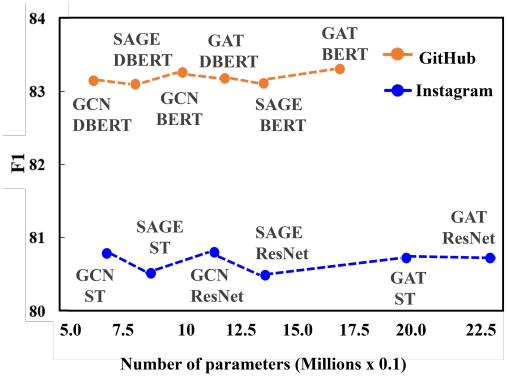


Figure 3: Model size vs. F1 score over $D_{\text{init}}$.

## 6 Conclusion

In this work, we aim to develop a graph contrastive learning model (CMI-GCL) on real-world multi-modality imbalanced graphs. Specifically, we propose co-modality GCL including inter GCL and intra GCL to automatically generate contrastive pairs based on the rich content. In addition, we propose network pruning to detect minority samples during co-modality GCL. Our proposed CMI-GCL is designed as a plug-and-play tool which is applicable to most mainstream models in different modalities. Extensive experiments across real-world graph datasets demonstrate that CMI-GCL outperforms state-of-the-art baselines for multiple applications and can significantly alleviate the data imbalance problem in graph. The **limitation** of this work lies in that, most benchmark graph datasets merely provide graph data but the raw content (e.g., image and text) in graph is not available. In the future, we will extend CMI-GCL to more real-world graph datasets.

# References


[1] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 2002.

[2] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *ICML*, 2020.

[3] Ting Chen, Simon Kornblith, Kevin Swersky, Mohammad Norouzi, and Geoffrey Hinton. Big self-supervised models are strong semi-supervised learners. In *NeurIPS*, 2020.

[4] Yin Cui, Menglin Jia, Tsung-Yi Lin, Yang Song, and Serge Belongie. Class-balanced loss based on effective number of samples. In *CVPR*, 2019.

[5] Ailin Deng and Bryan Hooi. Graph neural network-based anomaly detection in multivariate time series. In *AAAI*, 2021.

[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[7] Kaize Ding, Jundong Li, Rohit Bhanushali, and Huan Liu. Deep anomaly detection on attributed networks. In *SIAM*, 2019.

[8] Yuxiao Dong, Nitesh V Chawla, and Ananthram Swami. metapath2vec: Scalable representation learning for heterogeneous networks. In *KDD*, 2017.

[9] Yujie Fan, Yiming Zhang, Yanfang Ye, and Xin Li. Automatic opioid user detection from twitter: Transductive ensemble built on different meta-graph based similarities over heterogeneous information network. In *IJCAI*, 2018.

[10] Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *ICLR*, 2019.

[11] William L Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *NeurIPS*, 2017.

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[13] Tong He, Zhi Zhang, Hang Zhang, Zhongyue Zhang, Junyuan Xie, and Mu Li. Bag of tricks for image classification with convolutional neural networks. In *CVPR*, 2019.

[14] Sara Hooker, Aaron Courville, Gregory Clark, Yann Dauphin, and Andrea Frome. What do compressed deep neural networks forget? *arXiv preprint arXiv:1911.05248*, 2019.

[15] Ziyu Jiang, Tianlong Chen, Bobak J Mortazavi, and Zhangyang Wang. Self-damaging contrastive learning. In *ICML*, 2021.

[16] Bingyi Kang, Yu Li, Sa Xie, Zehuan Yuan, and Jiashi Feng. Exploring balanced feature spaces for representation learning. In *ICLR*, 2020.

[17] Bingyi Kang, Zhuang Liu, Xin Wang, Fisher Yu, Jiashi Feng, and Trevor Darrell. Few-shot object detection via feature reweighting. In *ICCV*, 2019.

[18] Bingyi Kang, Saining Xie, Marcus Rohrbach, Zhicheng Yan, Albert Gordo, Jiashi Feng, and Yannis Kalantidis. Decoupling representation and classifier for long-tailed recognition. In *ICLR*, 2020.

[19] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *ICLR*, 2017.

[20] Yann LeCun, John Denker, and Sara Solla. Optimal brain damage. In *NeurIPS*, 1989.

[21] Hao Li, Asim Kadav, Igor Durdanovic, Hanan Samet, and Hans Peter Graf. Pruning filters for efficient convnets. In *ICLR*, 2017.

[22] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *ICCV*, 2017.

[23] Ning Liu, Xiaolong Ma, Zhiyuan Xu, Yanzhi Wang, Jian Tang, and Jieping Ye. Autocompress: An automatic dnn structured pruning framework for ultra-high compression rates. In *AAAI*, 2020.

[24] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, 2021.

[25] Roweida Mohammed, Jumanah Rawashdeh, and Malak Abdullah. Machine learning with oversampling and undersampling techniques: overview study and experimental results. In *ICICS*, 2020.

[26] Fionn Murtagh. Multilayer perceptrons for classification and regression. *Neurocomputing*, 1991.

[27] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. In *KDD*, 2014.

[28] Senthil Purushwalkam and Abhinav Gupta. Demystifying contrastive self-supervised learning: Invariances, augmentations and dataset biases. In *NeurIPS*, 2020.

[29] Yiyue Qian, Yiming Zhang, Yanfang Ye, and Chuxu Zhang. Adapting meta knowledge with heterogeneous information network for covid-19 themed malicious repository detection. In *IJCAI*, 2021.

[30] Yiyue Qian, Yiming Zhang, Yanfang Ye, and Chuxu Zhang. Distilling meta knowledge on heterogeneous graph for illicit drug trafficker detection on social media. In *NeurIPS*, 2021.

[31] Jiezhong Qiu, Qibin Chen, Yuxiao Dong, Jing Zhang, Hongxia Yang, Ming Ding, Kuansan Wang, and Jie Tang. Gcc: Graph contrastive coding for graph neural network pre-training. In *KDD*, 2020.

[32] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *ICML*, 2021.

[33] Shebuti Rayana and Leman Akoglu. Collective opinion spam detection: Bridging review networks and metadata. In *KDD*, 2015.

[34] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019.

[35] Zheyan Shen, Peng Cui, Tong Zhang, and Kun Kunag. Stable learning via sample reweighting. In *AAAI*, 2020.

[36] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. In *ICLR*, 2018.

[37] Xiao Wang, Nian Liu, Hui Han, and Chuan Shi. Self-supervised heterogeneous graph neural network with co-contrastive learning. In *KDD*, 2021.

[38] Xiao Wang and Guo-Jun Qi. Contrastive learning with stronger augmentations. *arXiv preprint arXiv:2104.07713*, 2021.

[39] Yuzhe Yang and Zhi Xu. Rethinking the value of labels for improving class-imbalanced learning. In *NeurIPS*, 2020.

[40] Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. Graph contrastive learning with augmentations. In *NeurIPS*, 2020.

[41] Yiming Zhang, Yiyue Qian, Yujie Fan, Yanfang Ye, Xin Li, Qi Xiong, and Fudong Shao. dstyle-gan: Generative adversarial network based on writing and photography styles for drug identification in darknet markets. In *ACSAC*, 2020.

[42] Tianxiang Zhao, Xiang Zhang, and Suhang Wang. Graphsmote: Imbalanced node classification on graphs with graph neural networks. In *WSDM*, 2021.

[43] Yanqiao Zhu, Yichen Xu, Qiang Liu, and Shu Wu. An empirical study of graph contrastive learning. In *NeurIPS*, 2021.

## Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] Detailed discussion of limitation of this work is introduced in Section 6.
   (c) Did you discuss any potential negative societal impacts of your work? [No] There is no potential negative societal impact of our work.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] Please refer to the Appendix. We provide a URL about our source code.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] Please refer to experimental settings in Section 5.1.3.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] All reported results are the average of ten times.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] Please refer to experimental settings in Section 5.1.3.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [N/A] We use datasets/code provided in original papers.
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes] Code is provided in the attachment.
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A] We use datasets/code provided in existing papers.
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A] We use datasets/code provided in existing papers.
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]