Abstract

Graph contrastive learning algorithms have demonstrated remarkable success in various applications such as node classification, link prediction, and graph clustering. However, in unsupervised graph contrastive learning, some contrastive pairs may contradict the truths in downstream tasks and thus the decrease of losses on these pairs undesirably harms the performance in the downstream tasks. To assess the discrepancy between the prediction and the ground-truth in the downstream tasks for these contrastive pairs, we adapt expected calibration error (ECE) to graph contrastive learning. The analysis of ECE motivates us to propose a novel regularization method, \textbf{Contrast-Reg}, to ensure that decreasing the contrastive loss leads to better performance in the downstream tasks. As a plug-in regularizer, Contrast-Reg effectively improves the performance of existing graph contrastive learning algorithms. We provide both theoretical and empirical results to demonstrate the effectiveness of Contrast-Reg in enhancing the generalizability of the Graph Neural Network (GNN) model and improving the performance of graph contrastive algorithms with different similarity definitions and encoder backbones across various downstream tasks.

1 Introduction

Graph structures are widely used to capture abundant information, such as hierarchical configurations and community structures, present in data from various domains like social networks, e-commerce networks, knowledge graphs, the World Wide Web, and semantic webs. By incorporating graph topology along with node and edge attributes into machine learning frameworks, graph representation learning has demonstrated remarkable success in numerous essential applications, such as node classification, link prediction, and graph clustering. Graph contrastive learning has emerged as an effective method for learning graph representations, including algorithms such as DGI \cite{DGI}, GMI \cite{GMI}, GCC \cite{GCC}. 

1 Introduction

Graph structures are widely used to capture abundant information, such as hierarchical configurations and community structures, present in data from various domains like social networks, e-commerce networks, knowledge graphs, the World Wide Web, and semantic webs. By incorporating graph topology along with node and edge attributes into machine learning frameworks, graph representation learning has demonstrated remarkable success in numerous essential applications, such as node classification, link prediction, and graph clustering. Graph contrastive learning has emerged as an effective method for learning graph representations, including algorithms such as DGI \cite{DGI}, GMI \cite{GMI}, GCC \cite{GCC}.
and PyGCL (Zhu et al., 2021a). These algorithms use the noise contrastive estimation loss (NCEloss) to contrast the similarities of similar (or positive) node pairs against those of negative pairs. The algorithms differ in their definition of node similarity, hence the design of contrastive pairs. The design of the encoder backbone can vary from Graph Neural Networks (GNNs) (Kipf & Welling, 2017; Hamilton et al., 2017; Velickovic et al., 2018; Xu et al., 2019) to a skip-gram based model (Tang et al., 2015; Perozzi et al., 2014; Grover & Leskovec, 2016). In this paper, we focus on employing graph contrastive learning algorithms in an unsupervised learning manner and utilize GNNs as the encoder. The learned node embeddings are directly delivered to the downstream tasks.

Although graph contrastive algorithms have demonstrated strong performance in some downstream tasks, we have discovered that directly applying contrastive pairs and GNN models to new tasks or data is not always effective (as shown in Section 6). Nodes with different ground-truth labels can be selected as positive pairs during unsupervised training. Contrastive models that solely minimize the NCE loss can be misled by these pairs and learn certain spurious features that decrease the loss but are detrimental to downstream tasks. To address this issue, we adapt expected calibration error (ECE) to graph contrastive learning to assess the discrepancy between the predicted probability and the ground-truth in downstream tasks for the selected contrastive pairs. High ECE values indicate mis-calibration of the model. We identify two factors that contribute to the model’s mis-calibration: (a) the expectation of the prediction probability for randomly sampled pairs, $E_{(v,v')}[\sigma(h_v \cdot h_{v'})]$, and (b) the probability $q^+$ of $v'$ sharing the same label as $v$ in positive sampling. To address the mis-calibration in existing graph contrastive learning algorithms, we introduce a novel regularization method, denoted as **Contrast-Reg**. Contrast-Reg employs a regularization vector $r$, which consists of a random vector with each entry within the range $(0, 1]$. By ensuring node representations maintain similarity with $r$ and that pseudo-negative representations, generated through shuffled node features, remain dissimilar to $r$, Contrast-Reg guarantees that minimizing the contrastive loss results in high-quality representations that improve accuracy in downstream tasks, rather than overfitting specific spurious features. We provide both theoretical evidence and empirical experiments to support the effectiveness of Contrast-Reg. First, we derive a generalization bound for our contrastive GNN framework, building upon the theoretical framework depending on Rademacher complexity (Saunshi et al., 2019), and demonstrate that Contrast-Reg contributes to a decrease in the upper bound. This result indicates that this term promotes better alignment with the performance of downstream tasks while simultaneously minimizing the training loss, thereby improving the generalizability of the GNN model. Furthermore, we design experiments to examine the empirical performance of Contrast-Reg by formulating the graph contrastive learning framework into four components: a similarity definition, a GNN encoder, a contrastive loss function, and a downstream task. We apply Contrast-Reg to different compositions of these components and achieve superior results across various compositions.

The main contributions of this paper can be summarized as follows:

1. (Section 4.1) We adapt expected calibration error (ECE) to assess the discrepancy between the prediction and the ground-truth in the downstream tasks on contrastive pairs, and identify two key factors that influence the discrepancy.

2. (Section 4.2) We propose a novel regularization method, Contrast-Reg, to ensure that minimizing the contrastive loss results in high-quality representations and improved accuracy in downstream tasks. Contrast-Reg is a plugin regularizer generally effective with respect to various graph contrastive learnings.

3. (Section 4.2 & Section 5 & Section 6) We provide both theoretical results and empirical results to demonstrate the effectiveness of Contrast-Reg in improving the generalizability of the GNN model and achieving superior performance across different existing graph contrastive learning algorithms.

# 2 Related Work

**Graph representation learning.** Many graph representation learning models have been proposed. Factorization-based models (Ahmed et al., 2013; Cao et al., 2015; Qiu et al., 2018) factorize an adjacency matrix to obtain node representations. Random walk-based models such as DeepWalk (Perozzi et al., 2014)
sample node sequences as the input to skip-gram models to compute the representation for each node. Node2vec \cite{grover2016node2vec} balances depth-first and breadth-first random walk when it samples node sequences. HARP \cite{chen2018harp} compresses nodes into super-nodes to obtain a hierarchical graph to provide hierarchical information to random walk. GNN models \cite{kipf2016semi, hamilton2017inductive, velickovic2018graph, xu2019representational, qu2019mcmctree, thomas2023multitable} have shown great capability in capturing both graph topology and node/edge feature information. Most GNN models follow a neighborhood aggregation schema, in which each node receives and aggregates the information from its neighbors in each GNN layer, i.e., for the $k$-th layer, $\widetilde{h}_k^i = \text{aggregate}(h_{k-1}^j, j \in \text{neighborhood}(i))$, and $h_k^i = \text{combine}(\widetilde{h}_k^i, h_{k-1}^i)$. This work employs GNN models as the backbone and tests the representation across various downstream tasks, such as node classification, link prediction, and graph clustering.

**Graph contrastive learning.** Contrastive learning is a self-supervised learning method that learns representations by contrasting positive pairs against negative pairs. Contrastive pairs can be constructed in various ways for different types of data and tasks, such as multi-view \cite{tian2020contrastive, tian2019contrastive}, target-to-noise \cite{vandenOord2018representation, henaff2019data}, instance discrimination \cite{wu2018unsupervised}, mutual information \cite{belghazi2018mutual, hjelm2018learning}, context co-occurrence \cite{mikolov2013distributed, caron2018deep}, clustering \cite{asano2020contextual, caron2018deep}, multiple data augmentation \cite{chen2020simple}, known and novel pairs \cite{sun2023graph}, and contextually relevant \cite{neelakantan2022understanding}. Contrastive learning has been successfully applied to numerous graph representation learning models \cite{velickovic2019deep, peng2020contrastive, you2020graph, qiu2020gcc, zeng2021contrastive} to pseudo subgraph instance discrimination as a contrastive learning training objective and to leverage contrastive learning to empower graph neural networks in learning node representations. We characterize different types of node-level similarity as follows:

- **Structural similarity:** Structural similarity can be captured from various perspectives. From a graph theory viewpoint, GraphWave \cite{donnat2018graphwave} leverages the diffusion of spectral graph wavelets to capture structural similarity, while struc2vec \cite{ribeiro2017struc2vec} uses a hierarchy to measure node similarity at different scales. From an induced subgraph perspective, GCC \cite{qiu2020gcc} treats the induced subgraphs of the same ego network as similar pairs and those from different ego networks as dissimilar pairs. To capture community structure, vGraph \cite{sun2019node} utilizes the high correlation between community detection and node representations to incorporate more community structure information into node representations. To capture global-local structure, DGI \cite{velickovic2019deep} maximizes the mutual information between node representations and graph representations to allow node representations to contain more global information. GDCL \cite{zhao2021graph} incorporates the results of graph clustering to decrease the false-negative samples. DiGCL \cite{tong2021diversifying} generates contrastive pairs by Laplacian perturbations to retain more structural features of directed graphs. GCA \cite{zhu2021node} constructs contrastive pairs by designing adaptive augmentation schemes based on node centrality measures. AD-GCL \cite{suresh2021self} adopts trainable edge-dropping graph augmentation and optimizes both adversarial graph augmentation strategies and contrastive objectives to avoid the model learning redundant graph features.

- **Attribute similarity:** Nodes with similar attributes are likely to have similar representations. GMI \cite{peng2020contrastive} maximizes the mutual information between node attributes and high-level representations, and Pretrain GNN \cite{hu2020pretraining} applies attribute masking to help capture domain-specific knowledge. GCA \cite{zhu2021node} corrupts node features by adding more noise to unimportant node features, to enforce the model to recognize underlying semantic information.

Given the above subgraph instance discrimination objective with GNN backbones, NCEloss \cite{gutmann2012noise, dyer2014noise, mnih2012em, yang2020learning} is applied to optimize the model’s parameters. In addition, Graph Autoencoder, which is another popular graph self-supervised approach by reconstructing useful graph information, has also been extensively studied. For example, GraphMAE \cite{hou2022graphmae} focuses on feature reconstruction with a masking strategy and scaled cosine error. GraphMAE2 \cite{hou2023graphmae2} designs strategies of multi-view random re-mask decoding and latent representation prediction to regularize the reconstruction process. S2GAE \cite{tan2023unsupervised} randomly masks a portion of edges and...
learns to reconstruct these missing edges with a novel masking strategy. SeeGera (Li et al., 2023) adopts a novel hierarchical variational framework.

Numerous studies have demonstrated the success of enhancing a model’s generalizability by devising advanced similarity functions. However, some designs may require extensive tuning, preventing these advanced techniques from adapting to settings beyond the scope of their experiments. This problem is especially prevalent in the unsupervised learning setting, where the ideal protocol is that the learned node embeddings, which are the output of the unsupervised contrastive model, should be applicable to downstream tasks upon convergence. The key difference between Contrast-Reg and other algorithms, such as GCA, GDCL, and DiGCL, lies in the fact that the latter methods are intended to incorporate structures that are critical to the performance of the downstream tasks, while Contrast-Reg ensures that minimizing the contrastive loss results in high-quality representations that enhance the accuracy in downstream tasks. Moreover, Contrast-Reg can naturally function as a plugin to these advanced graph contrastive learning algorithms to improve their performance.

**Expected Calibration Error.** Expected Calibration Error (ECE) (Guo et al., 2017) is a metric employed to quantify the calibration between confidence (largest predicted probability) and accuracy in a model. Calibration refers to the consistency between a model’s predicted probabilities and the actual outcomes. When a model is mis-calibrated, its generalization to unseen data in supervised learning tasks is likely to be poor (Müller et al., 2019; Pereyra et al., 2017; Guo et al., 2017; Zhang et al., 2018). We propose using ECE to evaluate the quality of node embeddings generated by unsupervised graph contrastive learning models. This calibration offers insights into addressing the challenge that applying graph contrastive learning algorithms to downstream tasks does not always yield optimal results.

**Regularization for graph representation learning.** GraphAT (Feng et al., 2019) and BVAT (Deng et al., 2019) introduce adversarial perturbations $\frac{\partial f}{\partial x}$ to the input data $x$ as regularizers to obtain more robust models. GraphSGAN (Ding et al., 2018) generates fake input data in low-density regions by incorporating a generative adversarial network as a regularizer. P-reg (Yang et al., 2021) leverages the smoothness property in real-world graphs to enhance GNN models. Graphnorm (Cai et al., 2021) proposes a novel feature normalization method. ALS (Zhou et al., 2021) employs label propagation to adaptively integrate label smoothing into GNN training. G-Mixup (Han et al., 2022) uses graphons as a surrogate to apply mixup techniques to graph data.

The above regularizers are designed to address the mis-calibration problem in the unsupervised graph contrastive learning optimization process and its performance on downstream tasks. It is worth noting that Contrast-Reg could be used in conjunction with the aforementioned regularization techniques.

### 3 Preliminaries

We begin by introducing the concepts and foundations of graph contrastive learning (GCL). Let a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be denoted, where $\mathcal{V} = v_1, v_2, \ldots, v_n$ and $\mathcal{E}$ represent the vertex set and edge set of $\mathcal{G}$, and the node feature vector of node $v_i$ be $x_i$. Our objective is to learn node embeddings through unsupervised graph contrastive learning and subsequently apply simple classifiers leveraging these embeddings for various downstream tasks, such as node classification, link prediction, and graph clustering.

**Graph contrastive learning** Given a graph $\mathcal{G}$ with node features $\mathcal{X} = (x_1, x_2, \ldots, x_n)$, the aim of graph contrastive learning is to train an encoder $f : (\mathcal{G}, \mathcal{X}) \rightarrow \mathbb{R}^d$ for all input data points $v_i \in \mathcal{V}$ with node feature vector $x_i$ by constructing positive pairs $(v_i, v_i^+)$ and negative pairs $(v_i, v_{i1}, \ldots, v_{iK})$.

The encoder $f$ is typically implemented using Graph Neural Networks (GNNs). Specifically, let $h_i$ represent the output embedding of the encoder $f$ for node $v_i$, $h_i^{(k)}$ as the embedding at the $k$-th layer, and $h_i^0 = x_i$. 
The output of the encoder $f$ is then iteratively defined as:

$$
m_i^{(k)} = \text{Aggregate}_k \left( \left\{ h_j^{(k-1)} : v_j \in \mathcal{N}(v_i) \right\} \right)$$
$$h_i^{(k)} = \text{RELU} \left( W^k \cdot \text{Update} \left( h_i^{(k-1)}, m_i^{(k)} \right) \right),$$

(1)

where $\mathcal{N}(v_i)$ the set of nodes adjacent to $v_i$, and Aggregate and Update are the aggregation and update function of GNNs [Gilmer et al., 2017]. $h_i$ is the last layer of $h_i^{(k)}$ for node $v_i$.

Leveraging various types of similarity as pseudo subgraph instance discrimination labels, graph contrastive learning constructs positive and negative pairs to train the embedding $h_i$ by optimizing the loss on these pairs. The most commonly employed loss is the NCEloss:

$$\hat{L}_{nce} = \frac{1}{M} \sum_{i=1}^{M} \left[ -\log \sigma(h_i^{T} h_i^+) + \sum_{k=1}^{K} \log \sigma(h_i^{T} h_i^k) \right]$$

(2)

with $M$ samples $(v_i, v_i^+, v_i^-, \ldots, v_i^M)_{i=1}^{M}$ in empirical setting where $\sigma(\cdot)$ is the sigmoid function.

**Calibrating graph contrastive learning by the expected calibration error (ECE)**

Expected Calibration error measures the degree to which the model output probabilities match ground-truth accuracy in supervised tasks [Naeini et al., 2015; Guo et al., 2017]. It’s defined as the expectation of absolute difference between the largest predicted probability (confidence) and its corresponding accuracy,

$$\text{ECE} = \mathbb{E}_{(v, v') \in S} \left[ |p(v, v') - \text{acc}(v, v')| \right]$$

(3)

In this paper, we extend the use of ECE to evaluate the quality of graph contrastive learning. Specifically, we can compare the predicted probability of a positive pair (i.e., a pair of nodes with the same label) with the true probability that the pair belongs to the same class. We can also compare the predicted probability of a negative pair (i.e., a pair of nodes with different labels) with the true probability that the pair belongs to different classes. By calculating the differences between the predicted and true probabilities for both positive and negative pairs, we can quantify the overall mis-calibration of the model and identify areas for improvement. The formal definition is presented as follows. The largest predicted probability, $p(v, v')$, is determined as

$$p(v, v') = \begin{cases} 
\sigma(h_v \cdot h_{v'}), & \text{as positive pair} \\
1 - \sigma(h_v \cdot h_{v'}), & \text{as negative pair}
\end{cases}$$

(4)

where $h_v$ and $h_{v'}$ are the embeddings for the target node $v$ and the selected sample $v'$, respectively, and $\sigma(\cdot)$ is the sigmoid function. The corresponding accuracy, $\text{acc}(v, v')$, is defined as follows:

$$\text{acc}(v, v') = \begin{cases} 
\mathbb{I}(v, v'), & \text{as positive pair} \\
1 - \mathbb{I}(v, v'), & \text{as negative pair}
\end{cases}$$

(5)

where $\mathbb{I}$ is an indicator function denoting whether $v$ and $v'$ has the same label, the positive/negative pairs are pseudo-labels for training the contrastive learning algorithm which may not equal to the true label. In this case, when two nodes are selected as positive pairs, $\text{acc}(v, v')$ is equal to 1 if $v'$ and $v$ belong to the same class and equal to 0 otherwise; when the two nodes are selected as negative pairs, $\text{acc}(v, v') = 1 - \mathbb{I}(v, v')$.

Using the ECE metric in this way allows us to evaluate the quality of the embeddings outputted by the model ($\sigma(h_v \cdot h_{v'})$) and their accuracy in downstream tasks ($\text{acc}(v, v')$), such as node classification. We can improve the general performance and utility of graph contrastive learning algorithms by detecting regions of mis-calibration and developing novel techniques for enhancing calibration performance.
Methodology

To calibrate the model, we first adapt the expected calibration error (ECE) formula for the graph contrastive learning setting (Section 4.1). In order to ensure that minimizing the contrastive loss leads to high-quality representations that improve accuracy in downstream tasks, we propose a regularization term designed to lower the ECE value and enhance the model’s generalizability (Section 4.2). We also provide theoretical evidence to support the effectiveness of the proposed regularization term.

4.1 Empirical Calibration Reveals Limitations of Existing Graph Contrastive Learning Algorithms

To calibrate the existing graph contrastive learning algorithms, we measure the degree of calibration using the expected calibration error (ECE) metric, which is defined by Equation 3:

\[
ECE = r^+ \cdot \left( 1 - \mathbb{E}_{v,v' \sim \mathcal{D}} [\pi(v,v') \cdot (1 - q^+)] \right) + \mathbb{E}_{v,v' \sim \mathcal{D}} [\pi(v,v') \cdot (1 - q^-)]
\]

where \( q^+ \) and \( q^- \) are the probabilities that the node \( v' \) has the same label as node \( v \) in positive and negative sampling, respectively; \( r^+ \) and \( r^- \) are the ratios of sampling positive and negative samples, with \( r^+ + r^- = 1 \). In this study, we set \( r^+ = r^- = 0.5 \).

Building upon this formulation, we propose the following claim that takes into account the positive and negative pair construction assumptions to analyze the potential issues that lead to the mis-calibration between the decrease in graph contrastive learning training loss and the degradation of downstream task performance.

Claim 4.1. Under the assumption that negative sampling is uniformly sampled, and positive sampling is sampled based on the calculated distance between pairwise embeddings, ECE is positively correlated with the expectation of the prediction value for randomly sampled pairs \( \mathbb{E}_{v,v' \sim \mathcal{D}} [\pi(v,v') \cdot (1 - q^-)] \), and negatively correlated with the probability \( q^+ \) of \( v' \) having the same label as \( v \) in positive sampling.

We provide a thorough analysis of this claim in Appendix A.1. To investigate the changes in these two factors and the ECE value over the process of minimizing the NCELoss (Equation 2), we conduct an illustrative
experiment with an existing graph contrastive learning algorithm and present the relevant values in Figure 1. In Figure 1a, we show that as the NCE loss (red solid line) is minimized over the epochs, the expectation of the prediction value for randomly sampled pairs \( \mathbb{E}(v, v') [\sigma(h_v, h_{v'})] \) (blue dashed line) increases. Moreover, Figure 1b demonstrates the model’s challenge in identifying genuine positive pairs, with the probability \( q^+ \) initially increasing before gradually declining (red solid line), and the ECE value first decreasing before rising (blue dashed line). These findings suggest that, as the epochs progress, merely reducing the NCE loss is insufficient to enhance the accuracy of downstream tasks. Our empirical calibration shows that the model learns certain spurious features that reduce the loss but are ultimately harmful to downstream tasks when solely minimizing the NCE loss. The above analysis emphasizes the need for graph contrastive learning algorithms to effectively quantify the relationships between learned embeddings and the accuracy of downstream tasks. In the next section, we present our approach to mitigate the possible risks associated with spurious feature learning in graph contrastive learning algorithms and show its effectiveness through a number of experiments.

### 4.2 Proposed Regularization Term: Contrast-Reg

To ensure that minimizing the NCE loss aligns with downstream task accuracy, we propose a contrastive regularization term, denoted as Contrast-Reg, given by:

\[
L_{\text{reg}} = - \mathbb{E}_{h, h'} \left[ \log \sigma(h^T v W r) + \log \sigma(-h^T v W r) \right],
\]

where \( r \) is a random vector uniformly sampled from \( (0, 1) \), \( W \) is a trainable parameter, and \( h \) is the noisy feature generated by different data augmentation techniques, such as those used in the previous literature. In Section 5, we will discuss how we calculate the noisy features in the GNN setting. In the following, we will introduce the impact of Contrast-Reg on the ECE value and the generalizability of the GNN model.

#### ECE decreases by incorporating Contrast-Reg

In Claim 4.1 and Appendix A.1, it is proven that the ECE value is positively correlated with \( \mathbb{E}(v, v') [\sigma(h_v, h_{v'})] \) and inversely correlated with \( q^+ \). In addition, Theorem 2 in Appendix A.3 shows that Contrast-Reg reduces the variance of the learned node embeddings' norm \( ||h|| \), thereby preventing the loss from decreasing due to exploding embedding norms. As a result, the norm values are lower than those obtained with the vanilla contrastive loss, leading to a reduction in \( \mathbb{E}(v, v') [\sigma(h_v, h_{v'})] \) and an increase in representation quality. With the improvement in the representation quality, \( \mathbb{E}(v, v') [\sigma(h_v, h_{v'})] \) decreases and \( q^+ \) increases, resulting in a decreased ECE value. Furthermore, we conducted an empirical study to examine the impact of Contrast-Reg on the ECE value and investigate the two factors that cause miscalibration. Figure 2a shows that the expectation of the prediction value for randomly sampled pairs \( \mathbb{E}(v, v') [\sigma(h_v \cdot h_{v'})] \) with Contrast-Reg increased much more slowly compared to the vanilla NCE loss (green solid line compared to the red solid line). Figure 2b presents the changes in positive sampling accuracy \( q^+ \) over the epochs of the vanilla NCE loss and NCE loss with Contrast-Reg, where Contrast-Reg helps \( q^+ \) increase, while \( q^- \) trained by vanilla NCE loss decreased after the initial increases. These two factors contribute to the different ECE changes in the vanilla NCE loss and NCE loss with Contrast-Reg. The red dashed line indicates that the ECE value increases after the initial decreases, while the green dashed line shows that the ECE value decreases slightly. The comparison demonstrates that applying Contrast-Reg alleviates the miscalibration in representation learning, ensuring that minimizing the contrastive loss results in high-quality representations with increased accuracy in downstream tasks, rather than overfitting certain spurious features. We provide a detailed comparison of the impact on the ECE value between models with and without Contrast-Reg in Appendix A.2 across different datasets.

#### Generalizability improved by incorporating Contrast-Reg

In Section 6.3, we present empirical evidence supporting the effectiveness of Contrast-Reg’s ability to enhance the generalizability of the GNN model through an ablation study. Furthermore, we analyze the impact of Contrast-Reg on the generalizability of graph contrastive learning algorithms using the Rademacher complexity. In Theorem 1, we offer performance guarantees for the learned graph embeddings outputted by the GNN function class \( F = f \) with the unsupervised loss function \( L_{\text{sup}}(f) \) on the downstream average classification task \( L^d_{\text{sup}}(f) \). Detailed settings can be found in Appendix A.4. Assume that \( f \) is bounded, i.e., \( ||h_i|| \leq R \)
In Equation (8), \( \sigma_{\text{max}}(M(f,c)) \) and \( \mathbb{E}_{v_i \sim D_c} \| h_i \|^2 \) are upper bounds of \( \mathbb{E}_{v_i \sim D_c} [\| h_i \|^2] \) and \( \mathbb{E}_{v_i \sim D_c} [\| h_i \|^2] \), respectively. The advantage of incorporating Contrast-Reg into graph contrastive learning lies in its ability to reduce both the largest singular value of matrix \( M(f,c) \) (\( \sigma_{\text{max}}(M(f,c)) \)) and the expectation of the embedding norm within the same class, \( \mathbb{E}_{v_i \sim D_c} [\| h_i \|^2] \) (as illustrated in Figure 2c). This reduction leads to a decrease in the upper bound of \( s(f) \), ultimately yielding a lower value for the term \( \beta s(f) + \eta \text{Gen}_M \) compared to vanilla graph contrastive learning. The reduction in this term promotes better alignment with the performance of downstream tasks while simultaneously minimizing the training loss. Further experiments on the generalizability of the Contrast-Reg approach can be found in Section 6.3 and a more detailed explanation on lowering the term \( s(f) \) is provided in Appendix A.3.

5 A Contrastive GNN Framework

Algorithm 1 presents the graph contrastive learning framework and how Contrast-Reg is plugged into the framework. The framework employs a GNN model, such as GCN [Kipf & Welling 2017], GAT [Velickovic]...
et al., 2018), and GIN (Xu et al., 2019), to obtain node embeddings. To train the parameters of the GNN backbone, we generate contrastive pairs using the functions SeedSelect and Contrast. In addition, we incorporate the Contrast-Reg term into the optimization process, along with the contrastive loss on these pairs. After the model convergence, the embeddings are directly delivered to the downstream tasks.

Algorithm 1: Graph Contrastive Learning Framework

Input: Graph $G = (V, E)$, node attributes $X$, a GNN model $f : V \rightarrow \mathbb{R}^H$, the number of epochs $e$;
Output: A trained GNN model $f$;
1 Initialize training parameters;
2 for epoch $\leftarrow 1$ to $e$
3 \hspace{1em} $C = \text{SeedSelect}(G, X, f, \text{epoch})$;
4 \hspace{1em} $P = \text{Contrast}(C, G, X, f)$;
5 \hspace{1em} loss = NCEloss($P$) + Contrast-Reg($G, X, f$);
6 \hspace{1em} Back-propagation and update $f$;
7 end

The functions SeedSelect and Contrast are responsible for selecting the seed nodes and sampling the positive/negative pairs for these seeds while incorporating various priors for both structural and attribute aspects of the graph. To illustrate the graph contrastive learning framework, we utilize two simple graph contrastive learning algorithms (ML and LC).

- **LC: Structural similarity** We adopt clustering methods to detect communities in a graph and generate positive/negative pairs in the node representation space (Newman, 2006). The SeedSelect function is implemented using curriculum learning, following the design of AND (Huang et al., 2019). Initially, SeedSelect utilizes node embeddings to select a node set $C$ with low entropy, reducing randomness and uncertainty during early training stages. As training progresses, SeedSelect gradually expands $C$ by adding more nodes. For each node $x_i$ in $C$, Contrast generates a positive node $x_i^+$ that is highly similar, and a randomly sampled negative node $x_i^-$, returning a tuple of their representations: $\{(f(x_i), f(x_i^+), f(x_i^-))\}_{x_i \in C}$.

- **ML: Attribute similarity** In ML, we assume that nodes with similar attributes should have similar representations to preserve the attribute information. Following the design of DIM (Hjelm et al., 2019) and GMI (Peng et al., 2020), we adopt a multi-level representation approach. Initially, SeedSelect includes all nodes in the node set $G$. For each seed node $x_i$, Contrast uses the node itself as the positive node and randomly samples a negative node. The positive representation of the seed node $x_i$ is generated by adding an additional GNN layer upon $f$, denoted as $g$. The tuple containing the representations is denoted as $\{(f(x_i), g(x_i), f(x_i^-))\}_{x_i \in C}$.

Advanced graph contrastive learning algorithms can also be easily implemented using this framework. As an example, Graph Contrastive Analysis (GCA) (Zhu et al., 2021b) algorithm can be implemented by making a simple modification to the Contrast function found in Appendix B. Overall, Contrast-Reg serves as a highly effective plugin regularizer for graph contrastive learning with various backbones, and with different Contrast and SeedSelect implementations.

6 Experimental Results

We begin by introducing the experimental settings in Section 6.1. Section 6.2 presents the main results across various downstream tasks. Moreover, we assess the benefits of Contrast-Reg through ablation studies.

6.1 Experiment Settings

**Downstream tasks** We conduct our experiments on three distinct downstream tasks, namely node classification, graph clustering, and link prediction. The experimental procedure consists of two stages: first, we utilize positive and negative contrastive pairs to train the GNN models in an unsupervised manner, obtaining the node embeddings. Subsequently, we apply these embeddings to the downstream tasks by integrating
Training details The datasets we employ encompass citation networks, web graphs, co-purchase networks, and social networks. Comprehensive statistics for these datasets can be found in Appendix D. For Cora, Citeseer, Pubmed, ogbn-arxiv, ogbn-products, and Reddit, we adhere to the standard dataset splits and randomly divide the train/validation/test sets, allocating 20/30/all remaining nodes per class, in accordance with the recommendations in the previous literature (Shchur et al., 2018). We measure performance across 25 (5x5) different runs, comprising 5 random splits and 5 fixed-seed runs (from 0 to 4) for each random split. The hyperparameter configurations can be found in Appendix D.

6.2 Main Results

Our primary results involve comparing our proposed Contrast-Reg, along with the chosen similarity definitions, against state-of-the-art algorithms across various downstream tasks.

6.2.1 Node Classification

We evaluate node classification performance on all datasets, utilizing both full-batch training and stochastic mini-batch training. Our methods are compared with DGI (Veličković et al., 2019), GMI (Peng et al., 2020), node2vec (Grover & Leskovec, 2016), GCA (Zhu et al., 2021b), GraphMAE (Hou et al., 2022) and supervised GCN (Kipf & Welling, 2017). GCA, DGI, and GMI represent state-of-the-art algorithms in unsupervised graph contrastive learning. Node2vec is an exemplary algorithm for random walk-based graph representation algorithms (Grover & Leskovec, 2016), GraphMAE (Hou et al., 2022), while GCN is a classic semi-supervised GNN model. We incorporate the Contrast-Reg into ML, LC, GCA and GraphMAE models and compare them against the baseline models. The results, presented in Table 1, demonstrate that GCA and GraphMAE outperform Contrast-Reg achieve excellent performance across all the datasets that are trained in the full-batch mode and even surpass the performance of the supervised GCN.

Table 1: Downstream task: node classification

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Contrast-Reg</th>
<th>Cora</th>
<th>Citeseer</th>
<th>Pubmed</th>
<th>Wiki</th>
<th>Computers</th>
<th>Photo</th>
<th>ogbn-arxiv</th>
<th>ogbn-products</th>
<th>Reddit</th>
</tr>
</thead>
<tbody>
<tr>
<td>GCN</td>
<td>81.54±0.08</td>
<td>71.25±0.67</td>
<td>79.20±0.38</td>
<td>72.40±0.01</td>
<td>70.82±2.04</td>
<td>88.75±1.99</td>
<td>71.74±0.20</td>
<td>75.64±0.21</td>
<td>94.02±0.05</td>
<td></td>
</tr>
<tr>
<td>node2vec</td>
<td>71.07±0.06</td>
<td>47.37±0.99</td>
<td>66.24±1.60</td>
<td>58.74±1.35</td>
<td>75.37±1.32</td>
<td>83.63±1.35</td>
<td>70.65±0.13</td>
<td>72.89±0.10</td>
<td>93.26±0.04</td>
<td></td>
</tr>
<tr>
<td>DGI</td>
<td>81.90±0.61</td>
<td>71.85±0.37</td>
<td>76.89±3.13</td>
<td>63.70±1.43</td>
<td>69.42±1.93</td>
<td>77.19±2.00</td>
<td>69.66±1.98</td>
<td>77.00±2.21</td>
<td>94.14±0.00</td>
<td></td>
</tr>
<tr>
<td>GMI</td>
<td>80.95±0.05</td>
<td>71.11±0.11</td>
<td>77.97±0.04</td>
<td>63.35±0.03</td>
<td>79.27±1.44</td>
<td>87.08±1.21</td>
<td>68.36±1.09</td>
<td>75.75±1.20</td>
<td>94.19±0.04</td>
<td></td>
</tr>
<tr>
<td>GCA</td>
<td>82.51±0.75</td>
<td>70.83±0.69</td>
<td>85.66±1.01</td>
<td>76.39±1.42</td>
<td>87.47±0.40</td>
<td>92.07±1.00</td>
<td>OOM</td>
<td>OOM</td>
<td>OOM</td>
<td></td>
</tr>
<tr>
<td>GraphMAE</td>
<td>84.14±0.30</td>
<td>73.06±0.34</td>
<td>80.98±0.41</td>
<td>77.40±1.18</td>
<td>80.77±1.42</td>
<td>88.93±1.18</td>
<td>71.47±0.27</td>
<td>71.18±0.18</td>
<td>94.43±0.03</td>
<td></td>
</tr>
</tbody>
</table>

Table 2: Downstream task: graph clustering

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Contrast-Reg</th>
<th>Cora</th>
<th>Citeseer</th>
<th>Wiki</th>
<th>Acc</th>
<th>NMI</th>
<th>F1</th>
<th>Acc</th>
<th>NMI</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>node2vec</td>
<td>61.78±0.30</td>
<td>44.47±0.21</td>
<td>62.55±0.26</td>
<td>39.58±0.37</td>
<td>24.23±0.27</td>
<td>37.54±0.39</td>
<td>43.29±0.58</td>
<td>37.39±0.52</td>
<td>36.35±0.51</td>
<td></td>
</tr>
<tr>
<td>DGI</td>
<td>71.81±1.01</td>
<td>54.90±0.66</td>
<td>69.88±0.90</td>
<td>68.60±0.47</td>
<td>43.75±0.50</td>
<td>64.64±0.41</td>
<td>44.37±0.92</td>
<td>42.20±0.96</td>
<td>40.16±0.72</td>
<td></td>
</tr>
<tr>
<td>AGC</td>
<td>68.83±0.02</td>
<td>53.72±0.64</td>
<td>65.62±0.91</td>
<td>68.37±0.02</td>
<td>42.44±0.03</td>
<td>63.73±0.02</td>
<td>49.54±0.07</td>
<td>47.02±0.09</td>
<td>42.16±0.11</td>
<td></td>
</tr>
<tr>
<td>GMI</td>
<td>63.44±2.18</td>
<td>50.31±1.18</td>
<td>62.21±1.46</td>
<td>63.75±1.05</td>
<td>38.14±0.84</td>
<td>60.23±0.79</td>
<td>42.81±0.40</td>
<td>41.53±0.26</td>
<td>38.52±0.22</td>
<td></td>
</tr>
<tr>
<td>LC</td>
<td>✓</td>
<td>70.94±2.04</td>
<td>55.90±0.75</td>
<td>67.36±2.17</td>
<td>67.90±1.74</td>
<td>43.63±0.57</td>
<td>64.21±0.60</td>
<td>50.12±0.86</td>
<td>49.70±0.89</td>
<td>47.74±0.97</td>
</tr>
<tr>
<td>ML</td>
<td>✓</td>
<td>71.59±1.07</td>
<td>56.01±0.64</td>
<td>68.11±1.32</td>
<td>69.17±1.63</td>
<td>44.47±0.46</td>
<td>64.74±0.41</td>
<td>53.13±1.01</td>
<td>51.81±0.57</td>
<td>46.11±0.93</td>
</tr>
</tbody>
</table>

We employ multiclass logistic regression for the node classification, k-means for graph clustering, and a single MLP layer for link prediction. Moreover, we conduct experiments in the pretrain-finetune paradigm, as this approach constitutes a significant component of graph contrastive learning (Qiu et al., 2020). We first employ a large graph to train the GNN models, followed by finetuning such models and train a simple downstream classifier on a separate graph.
6.2.2 Graph Clustering

We assess clustering performance using three metrics: accuracy (Acc), normalized mutual information (NMI), and F1-macro (F1), following the previous literature [Xia et al., 2014]. Higher values indicate better clustering performance. We compare our methods with DGI, node2vec, GMI, and AGC [Zhang et al., 2019] on the Cora, Citeseer, and Wiki datasets. AGC is a state-of-the-art graph clustering algorithm that leverages high-order graph convolution for attribute graph clustering. For all models and datasets, we employ k-means to cluster both the labels and representations of nodes. The clustering results of labels are considered as the ground truth. To reduce dimensionality, we apply PCA to the representations before using k-means, since high dimensionality can negatively impact clustering [Chen, 2018]. The random seed setting for model training is consistent with that in the node classification task. To minimize randomness, we set the clustering random seed from 0 to 4 and compute the average result for each learned representation. Table 2 presents improved results with and without PCA for each cell. Our algorithms, particularly ours (ML), exhibit superior performance in all cases, demonstrating the effectiveness of Contrast-Reg. It is worth noting that the superior results of ours (ML) compared to ours (LC) suggest that attributes play a crucial role in clustering, as graph clustering is applied to attribute graphs.

6.2.3 Link Prediction

In order to circumvent the data linkage issue in link prediction, we employ an inductive setting for graph representation learning. We randomly extract induced subgraphs (comprising 85% of the edges) from each original graph for training both the representation learning model and the link predictor, while reserving the remaining edges for validation and testing (10% for the test edge set and 5% for the validation edge set). We assess performance across 25 (5x5) different runs, utilizing a fixed-seed random split scheme with five distinct induced subgraphs and five fixed-seed runs (ranging from 0 to 4). We compare our results with the baseline algorithms, including DGI, GMI, node2vec, ML-GCN [Shiao et al., 2022], GRACE [Zhu et al., 2020], and unsupervised GCN (GCN-neg in Table 3). In addition, we incorporate Contrast-Reg into LC, ML-GCN, and GRACE against the aforementioned baselines. Our results, presented in Table 3, demonstrate that Contrast-Reg consistently improves the performance of graph contrastive learning models, as evidenced by comparing the results of ML-GCN and GRACE with and without Contrast-Reg. Furthermore, we achieve the state-of-the-art results with the algorithms incorporating Contrast-Reg.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Contrast-Reg Cora</th>
<th>Citeseer</th>
<th>Pubmed</th>
<th>Wiki</th>
</tr>
</thead>
<tbody>
<tr>
<td>GCN-neg</td>
<td>92.40±0.51</td>
<td>92.27±0.90</td>
<td>97.24±0.19</td>
<td>93.27±0.31</td>
</tr>
<tr>
<td>node2vec</td>
<td>86.33±0.87</td>
<td>79.60±1.58</td>
<td>81.74±0.57</td>
<td>92.41±0.35</td>
</tr>
<tr>
<td>DGI</td>
<td>93.62±0.98</td>
<td>95.03±1.73</td>
<td>97.24±0.13</td>
<td>95.55±0.35</td>
</tr>
<tr>
<td>GMI</td>
<td>91.31±0.88</td>
<td>92.23±0.80</td>
<td>95.14±0.25</td>
<td>95.30±0.29</td>
</tr>
<tr>
<td>ML-GCN</td>
<td>94.81±0.55</td>
<td>96.38±0.63</td>
<td>92.86±0.26</td>
<td>89.84±0.69</td>
</tr>
<tr>
<td>GRACE</td>
<td>92.82±0.65</td>
<td>93.14±0.73</td>
<td>96.05±0.91</td>
<td>87.72±1.16</td>
</tr>
<tr>
<td>LC</td>
<td>✓</td>
<td>94.61±0.64</td>
<td>95.63±0.88</td>
<td>97.26±0.15</td>
</tr>
<tr>
<td>ML-GCN</td>
<td>✓</td>
<td>95.18±0.31</td>
<td>96.83±0.55</td>
<td>95.10±0.22</td>
</tr>
<tr>
<td>GRACE</td>
<td>✓</td>
<td>93.60±0.57</td>
<td>93.24±0.62</td>
<td>96.60±1.56</td>
</tr>
</tbody>
</table>

Table 3: Downstream task: link prediction

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Contrast-Reg Reddit</th>
<th>ogbn-products</th>
</tr>
</thead>
<tbody>
<tr>
<td>No pretraining</td>
<td>90.44±1.62</td>
<td>84.69±0.79</td>
</tr>
<tr>
<td>DGI</td>
<td>92.09±1.05</td>
<td>86.37±0.19</td>
</tr>
<tr>
<td>GMI</td>
<td>92.13±1.16</td>
<td>86.14±0.16</td>
</tr>
<tr>
<td>ML</td>
<td>✓</td>
<td>92.18±0.97</td>
</tr>
<tr>
<td>LC</td>
<td>✓</td>
<td>92.52±0.55</td>
</tr>
</tbody>
</table>

Table 4: Pretraining
Reg, as well as ML-GCN with ML-GCN+Contrast-Reg in Table 3. These experiments support that Contrast-
of various models, including GCA with GCA+Contrast-Reg in Table 1, GRACE with GRACE+Contrast-
backbones and graph contrastive learning algorithms. This conclusion is further supported by the comparison
demonstrate that Contrast-Reg consistently enhances graph contrastive learning performance across various
The ablation study of Contrast-Reg on the GAT and GIN backbone is presented in Table 6. The results
6.4 Generality of Contrast-Reg

Table 5: Graph contrastive learning with regularization

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Regularization</th>
<th>Cora</th>
<th>Wiki</th>
<th>Computers</th>
<th>Reddit</th>
</tr>
</thead>
<tbody>
<tr>
<td>ML</td>
<td></td>
<td>73.22±0.77</td>
<td>58.70±1.51</td>
<td>77.08±2.48</td>
<td>94.33±0.97</td>
</tr>
<tr>
<td>ML</td>
<td>$\ell_2$-normalization</td>
<td>80.14±0.92</td>
<td>61.83±1.36</td>
<td>80.80±1.45</td>
<td>94.07±0.19</td>
</tr>
<tr>
<td>ML</td>
<td>Weight-decay</td>
<td>81.65±0.42</td>
<td>63.67±1.47</td>
<td>79.09±0.32</td>
<td>94.34±0.06</td>
</tr>
<tr>
<td>ML</td>
<td>Contrast-Reg</td>
<td><strong>82.65±0.57</strong></td>
<td><strong>67.20±0.96</strong></td>
<td><strong>82.11±1.47</strong></td>
<td><strong>94.38±0.04</strong></td>
</tr>
<tr>
<td>LC</td>
<td></td>
<td>79.73±0.75</td>
<td>65.14±1.48</td>
<td>79.80±1.49</td>
<td>94.42±0.03</td>
</tr>
<tr>
<td>LC</td>
<td>$\ell_2$-normalization</td>
<td>81.09±0.59</td>
<td>63.96±0.64</td>
<td>80.73±1.36</td>
<td>94.20±0.06</td>
</tr>
<tr>
<td>LC</td>
<td>Weight-decay</td>
<td>81.94±0.44</td>
<td>65.17±1.34</td>
<td>81.89±1.58</td>
<td><strong>94.44±0.03</strong></td>
</tr>
<tr>
<td>LC</td>
<td>Contrast-Reg</td>
<td><strong>82.33±0.41</strong></td>
<td><strong>69.19±1.13</strong></td>
<td><strong>81.98±1.52</strong></td>
<td><strong>94.43±0.03</strong></td>
</tr>
</tbody>
</table>

6.2.4 Pretraining

We further assess the performance of Contrast-Reg in the context of the pretrain-finetune paradigm. For the Reddit dataset, we naturally partition the data by time, pretraining the models using the first 20 days. We generate an induced subgraph based on the pretraining nodes and divide the remaining data into three parts: the first part produce a new subgraph for fine-tuning the pre-trained model and training the classifier, while the second and third parts are designated for validation and testing. For the ogbn-products dataset, we split the data according to node ID, pretraining the models using a subgraph generated by the initial 70% of the nodes. The data splitting scheme for the remaining data mirrors that of the Reddit dataset. We conduct baseline experiments on DGI and GMI, employing the same GraphSAGE with GCN-aggregation encoder as in our model. Table 4 reveals that pretraining the model facilitates convergence to a more robust representation model with reduced variance, and Contrast-Reg can enhance the transferability of the pre-trained model.

6.3 Discussion of Contrast-Reg

To evaluate the benefits of Contrast-Reg, we conduct experiments comparing it to two approaches, specifically, $\ell_2$-normalization and weight decay, on four networks from different domains, with a focus on node classification task performance. It is important to note that some regularization techniques, such as those mentioned in Section 2, are not explicitly designed to address the miscalibration problem. $\ell_2$-normalization (Chen et al., 2020) mitigates the potential risk of the expectation of the prediction value for randomly sampled pairs $E_{(v, v')}[\sigma(v \cdot v')]$ exploding by explicitly eliminating the embedding norm for each node’s embeddings to tackle the miscalibration problem, while weight decay strives to achieve the same result by implicitly restricting the gradient descent step length. Table 5 presents a comparison of the accuracy achieved by different contrastive learning algorithms, ML and LC, when incorporating $\ell_2$-normalization, weight decay, and Contrast-Reg, as opposed to using the vanilla algorithms. The results indicate that integrating $\ell_2$-normalization, weight decay, and Contrast-Reg into graph contrastive learning algorithms improves the accuracy of downstream tasks, suggesting that addressing miscalibration enhances the generalization of learned embeddings for downstream tasks. However, the performance gains provided by Contrast-Reg exceed those of $\ell_2$-normalization and weight decay. This implies that while alternative algorithms exist to address miscalibration, Contrast-Reg emerges as the most effective method for improving the generalization of learned embeddings for downstream tasks. It is crucial to acknowledge the existence of other algorithms that may also contribute to mitigating the miscalibration problem, and further research is needed to explore and compare their effectiveness.

6.4 Generality of Contrast-Reg

The ablation study of Contrast-Reg on the GAT and GIN backbone is presented in Table 6. The results demonstrate that Contrast-Reg consistently enhances graph contrastive learning performance across various backbones and graph contrastive learning algorithms. This conclusion is further supported by the comparison of various models, including GCA with GCA+Contrast-Reg in Table 1, GRACE with GRACE+Contrast-Reg, as well as ML-GCN with ML-GCN+Contrast-Reg in Table 3. These experiments support that Contrast-
Reg serves as an effective regularization term for graph contrastive learning algorithms that utilize different similarity definitions and GNN encoder backbones. Furthermore, we have also conducted experiments on a graph autoencoder algorithm, GraphMAE [Hou et al., 2022], which is also graph self-supervised learning. The empirical results presented in Table 4 reveal that when combined with Contrast-Reg, GraphMAE achieves superior empirical performance, thus highlighting the potential benefits of Contrast-Reg for graph autoencoder methods. A comprehensive analysis of the impact of Contrast-Reg on graph autoencoder algorithms is an interesting direction and will be left as future work, which includes the theoretical investigations and empirical evaluations involving a broader range of graph autoencoder algorithms [Hou et al., 2022, 2023, Tan et al., 2023, Li et al., 2023).

7 Conclusions

We adapted expected calibration error (ECE) to the graph contrastive learning framework and identified the key factors that influence the discrepancy between the prediction in unsupervised training and the accuracy in downstream tasks. Motivated by ECE, we proposed a novel regularization regularizer, Contrast-Reg, to ensure that decreasing the contrastive loss leads to better performance in the downstream tasks. Our theoretical and empirical results both demonstrate the effectiveness of Contrast-Reg in improving the generalizability of the base graph contrastive learning model and achieving superior performance across different existing graph contrastive learning algorithms.

References


Fuli Feng, Xiangnan He, Jie Tang, and Tat-Seng Chua. Graph adversarial training: Dynamically regularizing based on graph structure. CoRR, abs/1902.08226, 2019.


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 ’20, pp. 1150–1160, 2020.


express the following relationship: Considering that positive sampling is based on the calculated distance between pairwise embeddings, we can

Thus, Equation 6 can be reformulated as:

Here, \( E \) sampled, we can write:

When the loss converges, \( \mathbb{E}_{\text{topk}} \sigma(h_v \cdot h_{v'}) \rightarrow 1 \).

Furthermore, by considering the definition in Equation 4 and the fact that negative samples are uniformly sampled, we can write:

Thus, Equation 6 can be reformulated as:

When the loss converges, \( \mathbb{E}_{\text{topk}} \sigma(h_v \cdot h_{v'}) \rightarrow 1 \), indicating that the ECE value is negatively correlated with the probability \( q^+ \) of \( v'_+ \). In addition, negative samples are uniformly sampled, so that \( q^- = 1/K \), where \( K \) represents the number of classes when \( K > 2 \). As a result, ECE is positively correlated with the expectation of the confidence value for randomly sampled pairs \( \mathbb{E}_{(v,v')} \sigma(h_v \cdot h_{v'}) \).

A.2 Contrast-Reg Leads to a Decreased ECE Value

We empirically investigate the impact of loss convergence on the ECE value, with and without Contrast-Reg, across various datasets and contrastive strategies, in Table 7. The results demonstrate that Contrast-Reg leads to a decrease in ECE values for all tested datasets and contrastive losses. This reduction in ECE across various datasets and contrastive strategies, in Table 7. The results demonstrate that Contrast-Reg

A.3 Explanation of Contrast-Reg’s Impact on the Term \( s(f) \)

In this section, we provide an explanation for why Contrast-Reg leads to a decrease in the upper bound of \( s(f) \). Firstly, we will prove that minimizing Eq. 7 results in a decrease in \( \text{Var}([h_i]) \) when \( h_i^T W_r > c \), as stated in Theorem 2. Then, based on the assumption that models with lower \( \text{Var}([h_i]) \) inherently favor lower values of \( \mathbb{E}([h_i]) \), both \( \mathbb{E}_{\nu_i} [||h_i||^2] \) and \( \sigma_{\text{max}}(M(f,c)) \) will decrease. Consequently, the upper bound of \( s(f) \) decreases with the implementation of Contrast-Reg. In the subsequent proof, \( h_i = f(x) \), and the notations \( h_i \) and \( f(x) \) may be used interchangeably for the sake of presentation clarity.

Theorem 2. Minimizing Eq. 7 induces the decrease in \( \text{Var}([h_i]) \) when \( h_i^T W_r > c \).

Proof. We minimize \( \mathcal{L}_{\text{reg}} \) by gradient descent with learning rate \( \beta \).

\[
\frac{\partial}{\partial f(x)} \mathcal{L}_{\text{reg}} = -\sigma(-f(x)^T W_r)W_r
\]
The embedding of $f(x)$ is updated as the following after adding $\mathcal{L}_reg$:

$$f(x) \leftarrow f(x) + \beta \left( \sigma(-f(x)^TWr)W \right)$$  \hspace{1cm} (13)

Eq. (13) shows that in every optimization step, $f(x)$ extends by $\beta\sigma(-f(x)^TWr)\|W\|$ along $r_0 := \frac{W_r}{\|W\|}$. If we do orthogonal decomposition for $f(x)$ along $r_0$ and its unit orthogonal hyperplain $\Pi(r_0)$, $f(x) = (f(x)^T r_0) r_0 + (f(x)^T \Pi(r_0)) \Pi(r_0)$. Thus we have

$$\|f(x)\| = \sqrt{(f(x)^T r_0)^2 + (f(x)^T \Pi(r_0))^2}. \hspace{1cm} (14)$$

The projection of $f(x)$ along $r_0$ is $f(x)^T r_0 = \frac{f(x)^T W r}{\|W\|}$, while the projection of $f(x)$ plus the Contrast-Reg update along $r_0$ is

$$\left( f(x)^T r_0 \right)_{reg} = \frac{f(x)^T W r}{\|W\|} + \frac{\beta}{1 + e^{\|x\|^2}} \|W\| .$$

Note that $\left( f(x)^T \Pi(r_0) \right)_{reg} = f(x)^T \Pi(r_0)$.

Based on Lemma 1 and Eq. (14),

when $\beta \|W\|^2 \leq 1$ and $f(x)^T W r > 1.5$, we have

$$\text{Var} \left( \left\| f(x) \right\| \right) < \text{Var} \left( \|f(x)\| \right). \hspace{1cm} (15)$$

**Lemma 1.** For a random variable $X \in [1.5, +\infty)$, a constant $\tau \in (0, 1]$ and a constant $c^2$, we have

$$\text{Var} \left( \sqrt{\left( X + \frac{\tau}{1 + e^X} \right)^2 + c^2} \right) < \text{Var} \left( \sqrt{X^2 + c^2} \right). \hspace{1cm} (16)$$

**Proof.** First, we consider

$$h(x) = \sqrt{\left( x + \frac{\tau}{1 + e^x} \right)^2 + c^2} - \sqrt{x^2 + c^2},$$

where $h(x)$ is strictly decreasing in $[x_0, +\infty)$ and strictly increasing in $(-\infty, x_0]$, and $x_0$ is the solution of $h'(x) = \frac{4h(x)}{dx} = 0$. Thus, we can approximate the range of $x_0 \in (0, 1.5)$ by the fact that $h'(0)h'(1.5) < 0$ for all $\tau$ and $c^2$.

Thus, for $x > y \geq 1.5$,

$$\sqrt{\left( x + \frac{\tau}{1 + e^x} \right)^2 + c^2} - \sqrt{x^2 + c^2} < \sqrt{\left( y + \frac{\tau}{1 + e^y} \right)^2 + c^2} - \sqrt{y^2 + c^2}$$

and since $(x + \frac{\tau}{1+e^x})$ is monotonically increasing, we get

$$0 < \sqrt{\left( x + \frac{\tau}{1 + e^x} \right)^2 + c^2} - \sqrt{\left( y + \frac{\tau}{1 + e^y} \right)^2 + c^2} < \sqrt{x^2 + c^2} - \sqrt{y^2 + c^2}.$$

When $y > x \geq 1.5$,

$$\sqrt{x^2 + c^2} - \sqrt{y^2 + c^2} < \sqrt{\left( x + \frac{\tau}{1 + e^x} \right)^2 + c^2} - \sqrt{\left( y + \frac{\tau}{1 + e^y} \right)^2 + c^2} < 0.$$
Further, we assume that $X$ and $Y$ are i.i.d. random variables sampled from $[1.5, +\infty)$,

$$
\text{Var} \left( \sqrt{(X + \frac{\tau}{1 + e^X})^2 + c^2} \right) \\
= \frac{1}{2} \times \mathbb{E}_{X,Y} \left[ \left( \sqrt{(X + \frac{\tau}{1 + e^X})^2 + c^2} - \sqrt{(Y + \frac{\tau}{1 + e^Y})^2 + c^2} \right)^2 \right] \\
= \frac{1}{2} \times \int \left( \sqrt{x^2 + c^2} - \sqrt{y^2 + c^2} \right)^2 p(x)p(y)dx\,dy \\
< \frac{1}{2} \times \int \left( \sqrt{x^2 + c^2} - \sqrt{y^2 + c^2} \right)^2 p(x)p(y)dx\,dy \\
= \text{Var}(\sqrt{X^2 + c^2})
$$

Remark 1. $\beta \|Wr\|_2^2 \leq 1$, which is the condition of Eq. (15), is not difficult to satisfy, since the magnitude of $r$ could be tuned. In practice, $r \in (0,1]$ can fit in all our experiments.

Remark 2. The range of $f(x)^Twr$ in Theorem 3 is not a tight bound for $x_0$ in Lemma 4. Since when Eq. (7) converges, $f(x)^Twr$ is much larger than 1.5 for almost all the samples empirically, we prove the case for $f(x)^Twr \in [1.5, +\infty)$.

A.4 Comprehensive Explanation of Theorem 1: Notations and Proof

To formally analyze the behavior of contrastive learning, we introduce the following concepts as the previous literature (Saunshi et al., 2019) does.

- **Latent classes**: Data are considered as drawn from latent classes $C$ with distribution $\rho$. Further, distribution $D_c$ is defined over feature space $X$ that is associated with a class $c \in C$ to measure the relevance between $x$ and $c$.

- **Semantic similarity**: Positive samples are drawn from the same latent classes, with distribution

$$
D_{\text{sim}}(x, x^+) = \mathbb{E}_{c \in \rho} \left[ D_c(x)D_c(x^+) \right],
$$

while negative samples are drawn randomly from all possible data points, i.e., the marginal of $D_{\text{sim}}$, as

$$
D_{\text{neg}}(x^-) = \mathbb{E}_{c \in \rho} \left[ D_c(x^-) \right]
$$

- **Supervised tasks**: Denote $K$ as the number of negative samples. The object of the supervised task, i.e., feature-label pair $(x, c)$, is sampled from

$$
D_T(x, c) = D_c(x)D_T(c),
$$

where $D_T(c) = \rho(c|c \in T)$, and $T \subseteq C$ with $|T| = K + 1$.

Mean classifier $W^\mu$ is naturally imposed to bridge the gap between the representation learning performance and linear separability of learn representations, as

$$
W^\mu_c := \mu_c = \mathbb{E}_{x \sim D_c}[f(x)].
$$

- **Empirical Rademacher complexity**: Suppose $F : X \rightarrow [1,0]$. Given a sample $S$,

$$
\mathcal{R}_S(F) = \mathbb{E}_e \left[ \sup_{f \in F} e^T f(S) \right],
$$

where $e = (e_1, \cdots, e_m)^T$, with $e_i$ are independent random variables taking values uniformly from $\{-1, +1\}$. 

20
In addition, the theoretical framework (Saunshi et al., 2019) makes an assumption: encoder $f$ is bounded, i.e., $\max_{x \in X} \|f(x)\| \leq R^2$, $R \in \mathbb{R}$.

To prove Theorem 1 we first list some key lemmas.

**Lemma 2.** For all $f \in \mathcal{F}$,

$$L_{\sup}^\mu(f) \leq \frac{1}{1-\tau} (L_{nce}(f) - 2\tau \log 2).$$

(19)

This bound connects contrastive representation learning algorithms and its supervised counterpart. This lemma is achieved by Jensen’s inequality. The details are given in Appendix A.6

**Lemma 3.** With probability at least $1 - \delta$ over the set $\mathcal{S}$, for all $f \in \mathcal{F}$,

$$L_{nce}(\hat{f}) \leq L_{nce}(f) + \text{Gen}_M.$$  

(20)

This bound guarantees that the chosen $\hat{f} = \arg\min_{f \in \mathcal{F}} L_{nce}^\mu$ cannot be too much worse than $f^* = \arg\min_{f \in \mathcal{F}} L_{nce}$. The proof applies Rademacher complexity of the function class (Mohri et al., 2018) and vector-contraction inequality (Maurer, 2016). More details are given in Appendix A.7

**Lemma 4.** $L_{nce}(f) \leq 4s(f) + 2 \log 2$.

This bound is derived by the loss caused by both positive and negative pairs that come from the same class, i.e., class collision. The proof uses Bernoulli’s inequality (details in Appendix A.8).

**Proof to Theorem 1.** Combining Lemma 2 and Lemma 3 we obtain with probability at least $1 - \delta$ over the set $\mathcal{S}$, for all $f \in \mathcal{F}$,

$$L_{\sup}^\mu(\hat{f}) \leq \frac{1}{1-\tau} (L_{nce}(f) + \text{Gen}_M).$$  

(21)

Then, we decompose $L_{nce} = \tau L_{nce}^\mu(f) + (1 - \tau)L_{nce}^\mu(f)$, apply Lemma 4 to Eq. (21), and obtain the result of Theorem 1.

**A.5 Contrastive Learning with NCELoss**

The contrastive loss is

$$L_{\text{un}} := \mathbb{E}_{(x,x^+) \sim \mathcal{D}_{\text{sim}}, (x_1^-, \cdots, x_K^-) \sim \mathcal{D}_{\text{neg}}} \left[ \ell(\{f(x)^T (f(x^+) - f(x_i^-))\}_{i=1}^K) \right],$$

where $\ell$ can be the hinge loss as $\ell(v) = \max\{0, 1 + \max_i \{-v_i\}\}$ or the logistic loss as $\ell(v) = \log_2 (1 + \sum_i \exp(-v_i))$. And its supervised counterpart is defined as

$$L_{\sup}^\mu := \mathbb{E}_{(x, c) \sim \mathcal{D}_{\text{tr}(x, c)}} \left[ \ell \left( f(x)^T \mu_c - f(x)^T \mu_{c'} \right) | c' \neq c \right].$$

A more powerful loss function, NCELoss, used in the previous literature (Velickovic et al., 2019; Yang et al., 2020; Mnih & Teh, 2012; Dyer, 2014), can be framed as

$$L_{nce} :=$$

$$- \mathbb{E}_{(x,x^+) \sim \mathcal{D}_{\text{sim}}, (x_1^-, \cdots, x_K^-) \sim \mathcal{D}_{\text{neg}}} \left[ \log \sigma(f(x)^T f(x^+)) + \sum_{k=1}^{K} \log \sigma(-f(x)^T f(x_k^-)) \right],$$

(22)

and its empirical counterpart with $M$ samples $(x_i, x_i^+, x_{1i}^-, \cdots, x_{Ki}^-)_{i=1}^{M}$ is given as

$$\hat{L}_{nce} := -\frac{1}{M} \sum_{i=1}^{M} \left[ \log \sigma(f(x_i)^T f(x_i^+)) + \sum_{k=1}^{K} \log \sigma(-f(x_i)^T f(x_k^-)) \right],$$

(23)

where $\sigma(\cdot)$ is the sigmoid function.

For its supervised counterpart, it is exactly the cross entropy loss for the $(K+1)$-way multi-class classification task:

$$L_{\sup}^\mu := - \mathbb{E}_{(x, c) \sim \mathcal{D}_{\text{tr}(x, c)}} \left[ \log \sigma(f(x)^T \mu_c) + \log \sigma(-f(x)^T \mu_{c'}) | c' \neq c \right].$$

(24)
A.6 Proof of Lemma 2

First, we prove that \( \ell(f(x^+), \{f(x_i^-)\}) = -(\log \sigma(f(x)^T f(x^+)) + \sum_{i=1}^{K} \log(\sigma(f(x)^T f(x^-))) \) is convex w.r.t. \( f(x^+), f(x_i^-), \ldots, f(x_K^-) \). Consider that \( \ell_1(z) = -\log \sigma(z) \) and \( \ell_2(z) = -\log \sigma(-z) \) are both convex functions since \( \ell_1' > 0 \) and \( \ell_2' > 0 \) for \( z \in \mathbb{R} \). Given \( f(x) \in \mathbb{R}, z^+ = f(x)^T f(x^+) \) and \( z^- = f(x)^T f(x^+) \) are affine transformation w.r.t. \( f(x^+) \) and \( f(x^-) \). Thus, when \( f(x) \) is fixed, \( \ell_1(f(x^+)) = -\log \sigma(f(x)^T f(x^+)) \) and \( \ell_2(f(x^-)) = -\log \sigma(-f(x)^T f(x^-)) \) are convex functions. As \( \ell_1 > 0 \) and \( \ell_2 > 0 \), we obtain \( \ell(f(x^+), \{f(x_i^-)\}) = -(\log \sigma(f(x)^T f(x^+)) + \sum_{i=1}^{K} \log(\sigma(f(x)^T f(x^-))) \) is convex since non-negative weighted sums preserve convexity [Boyd & Vandenberghe 2014]. By the definition of convexity,

\[
\mathcal{L}_{nce}(f) = \mathbb{E}_{x \sim D_x^+} \mathbb{E}_{e \sim D_e^+} \left[ \ell(f(x^+), \{f(x_i^-)\}) \right] = \mathbb{E}_{x \sim D_x^-} \mathbb{E}_{e \sim D_e^-} \left[ \ell(f(x^-), \{f(x_i^-)\}) \right] = (1 - \tau) \mathcal{L}_{nce}(f) + \tau \mathbb{E}_{x \sim D_x^- \cup D_x^+} \left[ -\log \sigma(f(x)^T \mu_{c+}) - \log \sigma(-f(x)^T \mu_{c-}) \right] \geq (1 - \tau) \mathcal{L}_{nce}(f) + 2\tau \log 2
\]

A.7 Generalization bound

Denote

\[
\tilde{\mathcal{F}} = \{ \tilde{f}(x_1, x_2, \ldots, x_K) \mid f(x_1), f(x_2), \ldots, f(x_K) \mid f \in \mathcal{F} \}.
\]

Let \( q = h \circ \tilde{f} \), and its function class,

\[
Q = \{ q = h \circ \tilde{f} \mid \tilde{f} \in \tilde{\mathcal{F}} \}.
\]

Denote \( z_i = (x_1, x_2, \ldots, x_K) \), suppose \( \ell \) is bounded by \( B \), then we can decompose \( h = \frac{1}{B} \ell \circ \phi \). Then we have \( q_j(z_i) = \frac{1}{B} \ell(\phi(f(z_i))) \), where

\[
\phi(\tilde{f}(z_i)) = \left( \sum_{t=1}^{d} f(x_i)_t f(x_i') \right) \left( \sum_{t=1}^{d} f(x_i)_t f(x_i') \right) \cdot \cdot \cdot ,
\]

\[
\ell(x) = -\left( \log \sigma(x_0) + \sum_{i=1}^{K} \log \sigma(-x_i) \right).
\]

From Eq. [25], we know that \( \phi: \mathbb{R}^{(K+2)d} \rightarrow \mathbb{R}^{K+1} \). Then we will prove that \( h \) is \( L \)-Lipschitz by proving that \( \phi \) and \( \ell \) are both Lipschitz continuity. First,

\[
\frac{\partial \phi(\tilde{f}(z_i))}{\partial f(x_t)} = f(x_{ik}) = \begin{cases} f(x_{ik})_t, & k = 0 \\ f(x_{ik})_t, & k = 1, \ldots, K \end{cases}
\]

\[
\frac{\partial \phi(\tilde{f}(z_i))}{f(x_{ik})} = \frac{\partial \phi(\tilde{f}(z_i))}{f(x_{ik})} = \frac{\partial \phi(\tilde{f}(z_i))}{f(x_{ik})} = f(x_t).
\]

If we assume \( \sum_{t=1}^{d} f(x_{ik})_t^2 \leq R^2 \) and \( \sum_{t=1}^{d} f(x_t)^2 \leq R^2 \),

\[
\|J\|_F = \sqrt{\sum_{t=1}^{d} f(x_{ik})_t^2 + \sum_{k=1}^{K} \sum_{t=1}^{d} f(x^-_{ik})_t^2 + (K + 1) \sum_{t=1}^{d} f(x_t)^2} \leq \sqrt{2(K + 1)R^2} = \sqrt{2(K + 1)R}
\]

22
Algorithm 2: ML
Parameter: Parameters of an (additional) GNN layer g.

1. Function Contrast(C, G, X, f):
   1. Let $g(x_i)$ be the representation of $x_i$ by stacking $g$ upon $f$;
   2. Randomly pick a negative node $x_i^-$ from $V$ for each $x_i \in C$;
   3. return $\{(g(x_i), f(x_i), f(x_i^-))\}_{i \in G}$

2. Function SeedSelect(G, X, f, epoch):
   1. return $V$

Combining $\|J\|_2 \leq \|J\|_F$, we obtain that $\phi$ is $\sqrt{2(K+1)}R$-Lipschitz. Similarly, $\ell$ is $\sqrt{K+1}$-Lipschitz. Since we assume that the inner product of embedding is no more than $R^2$. Thus, $l$ is bounded by $B = -(K + 1)\log(\sigma(-R^2))$. Above all, $h$ is $L$-Lipschitz with $L = \sqrt{2(K+1)}R$. Applying vector-contraction inequality (Maurer 2016), we have:

$$E_{\sigma \sim \{\pm 1\}^M}[\sup_{f \in F}[\sigma, (h \circ \hat{f})|_S]] \leq \sqrt{2}LE_{\sigma \sim \{\pm 1\}^{(K+1)M}}[\sup_{f \in F}[\sigma, \hat{f}|_S]].$$

If we write it in Rademacher complexity manner, we have:

$$R_S(\mathcal{Q}) \leq \frac{2(K+1)R}{B}R_S(\mathcal{F}).$$

Applying generalization bounds based on Rademacher complexity (Mohri et al. 2018) to $q \in \mathcal{Q}$. For any $\delta > 0$, with the probability of at least $1 - \frac{\delta}{2}$,

$$\mathbb{E}[q(z)] \leq \frac{1}{M} \sum_{i=1}^{M} q(x_i) + \frac{2R_S(\mathcal{Q})}{M} + 3 \sqrt{\frac{\log \frac{1}{\delta}}{2M}}$$

Thus for any $f$,

$$\mathcal{L}_{nce}(f) \leq \hat{\mathcal{L}}_{nce}(f) + \frac{4(K+1)RR_S(\mathcal{F})}{M} + 3B \sqrt{\frac{\log \frac{1}{\delta}}{2M}} \quad (26)$$

Let $\hat{f} = \arg \min_{f \in \mathcal{F}} \hat{\mathcal{L}}_{nce}(f)$ and $f^* = \arg \min_{f \in \mathcal{F}} \mathcal{L}_{nce}(f)$. By Hoeffding’s inequality, with probability of $1 - \frac{\delta}{2}$,

$$\hat{\mathcal{L}}_{nce}(f^*) \leq \mathcal{L}_{nce}(f^*) + B \sqrt{\frac{\log \frac{1}{\delta}}{2M}} \quad (27)$$

Substituting $\hat{f}$ into Eq. (26), combining $\hat{\mathcal{L}}_{nce}(\hat{f}) \leq \mathcal{L}_{nce}(f^*)$ and applying union bound, with probability of at most $\delta$

$$\mathcal{L}_{nce}(\hat{f}) \leq \mathcal{L}_{nce}(\hat{f}) + \frac{4(K+1)RR_S(\mathcal{F})}{M} + 3B \sqrt{\frac{\log \frac{1}{\delta}}{2M}} + B \sqrt{\frac{\log \frac{1}{\delta}}{2M}}$$

$$\leq \mathcal{L}_{nce}(f^*) + \frac{4(K+1)RR_S(\mathcal{F})}{M} + 4B \sqrt{\frac{\log \frac{1}{\delta}}{2M}}$$

$$\leq \mathcal{L}_{nce}(f) + \frac{4(K+1)RR_S(\mathcal{F})}{M} - 4(K+1)\log(\sigma(-R^2)) \sqrt{\frac{\log \frac{1}{\delta}}{2M}}$$

fails. Thus, with probability of at least $1 - \delta$, Eq. (28) holds.
A.8 Class collision loss

Let $p_i = |f(x)^T f(x_i)|$ and $p = \max_{i \in \{0, \ldots, K\}} p_i$. Considering

$$
\mathcal{L}_{ncc}^+(f) = -\mathbb{E} \left[ \log(\sigma(f(x)^T f(x_0^+))) + \sum_{i=1}^{K} \log(\sigma(-f(x)^T f(x_i^-))) \right]
$$

$$
= \mathbb{E} \left[ \log(1 + e^{-f(x)^T f(x_0^+)}) + \sum_{i=1}^{K} \log(1 + e^{f(x)^T f(x_i^-)}) \right]
\leq (K + 1)\mathbb{E} \left[ \log(1 + e^{p}) \right]
\leq (K + 1) \log 2 + (K + 1) \mathbb{E} [p]

(29)
$$

Since $x, x_0^+, x_1^-, \ldots, x_K^-$ are sampled i.i.d. from the same class,

$$
\mathbb{E}[p] = \int P[p \geq x] dx = \int (1 - (1 - P[p_0 \geq x])^{K+1}) dx.
\leq (K + 1) \mathbb{E} [\log(1 + e^{p})]
\leq (K + 1) \mathbb{E} [p]

(30)
$$

Applying Bernoulli’s inequality, we have

$$
\mathbb{E}[p] \leq \int (1 - (1 - (K + 1)P[p_0 \geq x])) dx
= \int (K + 1)P[p_0 \geq x] dx
= (K + 1)\mathbb{E}[p_0]
\leq (K + 1)\mathbb{E}[|f(x)^T f(x_0^+)|]
\leq (K + 1)\sqrt{\mathbb{E}[(f(x)^T f(x_0^+))^2]}.
$$

(31)

Therefore,

$$
\mathcal{L}_{ncc}^+(f) \leq (K + 1) \log 2 + (K + 1)^2 s(f).
$$

(32)
Algorithm 3: LC

Hyperparameter: $R$: curriculum update epochs; $k$: the number of candidate positive samples for seed node;

```plaintext
Function Contrast($C, G, X, f$):
  1. For $x_i \in C$, let $N_i^+ = \{x_j \in \text{Neighbor}(x_i)\}$ with largest $f(x_i)^T f(x_j)$;
  2. Randomly pick one positive node $x_i^+$ from $N_i^+$ for each $x_i \in C$;
  3. Randomly pick one negative node $x_i^-$ from $V$ for each $x_i \in C$;
  4. return $\{(f(x_i), f(x_i^+), f(x_i^-))\}_{x_i \in C}$;
end
```

Function SeedSelect($G, X, f, epoch$):

```plaintext
  1. if $\text{epoch} \% R \neq 1$ then
  2.   return the same set of seed nodes $C$ as in the last epoch;
  3. end
  4. $p_{i,j} \leftarrow \frac{f(i)^T f(j)}{\sum_{k \in V} f(i)^T f(k)}$ for $i, j \in V$;
  5. $H(i) \leftarrow -\sum_{j \in V} (p_{i,j} \log(p_{i,j}))$ for $i \in V$;
  6. return $(\lceil \frac{\text{epoch}}{R} \rceil + 1) \frac{R}{|G|}$ nodes with smallest $H$;
end
```

Algorithm 4: GCA

Hyperparameter: two stochastic augmentation functions set $T$ and $T'$;

```plaintext
Function Contrast($C, G, X, f$):
  1. Sample two stochastic augmentation functions $t \sim T$ and $t' \sim T'$;
  2. Generate two graph views $\tilde{G}_1 = t(G)$ and $\tilde{G}_2 = t'(G)$ by performing corruption on $G$;
  3. return $\{(f(\tilde{G}_1(x_i)), f(\tilde{G}_2(x_i)), f(\tilde{G}_2(x_j)))\}_{x_i \in G, x_j \neq x_i}$;
end
```

Function SeedSelect($G, X, f, epoch$):

```plaintext
  1. return $V$;
end
```

B Details of Graph Contrastive Algorithms

Algorithm 1 presents the graph contrastive learning framework and how Contrast-Reg is plugged into it. The functions SeedSelect and Contrast in the graph contrastive learning framework are responsible for selecting the seed nodes and sampling the positive/negative pairs for these seeds while incorporating various priors for both structural and attribute aspects of the graph. The detailed SeedSelect and Contrast implementations for ML, LC, GCA are referred to Algorithm 2, 3 and 4.

C Supplementary Experiments

C.1 Plug Contrast-Reg into DiGCL and GDCL

We applied Contrast-Reg to two graph contrastive learning algorithms, namely DiGCL (Tong et al., 2021) and GDCL (Zhao et al., 2021). GDCL employs graph clustering to reduce the number of false-negative samples, whereas DiGCL generates contrastive pairs using Laplacian perturbations to better preserve the structural attributes of directed graphs. In Table C we present a comparative analysis of the performance of Contrast-Reg, applied to both algorithms, as well as their performance without Contrast-Reg. We used the official implementations and the parameters of both algorithms, and conducted experiments with random seeds ranging from 0 to 9. Our results demonstrate that Contrast-Reg serves as an effective plugin to these advanced graph contrastive learning algorithms, even when applied to directed graphs (DiGCL).
Table 8: DiGCL & GDCL with and without Contrast-Reg

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Contrast-Reg</th>
<th>Cora-ml</th>
<th>Citeseer</th>
<th>Cora</th>
</tr>
</thead>
<tbody>
<tr>
<td>DiGCL</td>
<td></td>
<td>76.62±1.23</td>
<td>65.54±1.39</td>
<td></td>
</tr>
<tr>
<td>DiGCL ✓</td>
<td>✓</td>
<td>79.74±0.62</td>
<td>68.74±1.13</td>
<td></td>
</tr>
<tr>
<td>GDCL</td>
<td></td>
<td>72.98±0.73</td>
<td>85.41±0.53</td>
<td></td>
</tr>
<tr>
<td>GDCL ✓</td>
<td>✓</td>
<td>76.39±1.16</td>
<td>85.80±0.46</td>
<td></td>
</tr>
</tbody>
</table>

Table 9: Datasets

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Node #</th>
<th>Edge #</th>
<th>Feature #</th>
<th>Class #</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cora (Yang et al., 2016)</td>
<td>2,708</td>
<td>5,429</td>
<td>1,433</td>
<td>7</td>
</tr>
<tr>
<td>Citeseer (Yang et al., 2016)</td>
<td>3,327</td>
<td>4,732</td>
<td>3,703</td>
<td>6</td>
</tr>
<tr>
<td>Pubmed (Yang et al., 2016)</td>
<td>19,717</td>
<td>44,338</td>
<td>500</td>
<td>3</td>
</tr>
<tr>
<td>ogbn-arxiv (Hu et al., 2020a)</td>
<td>169,343</td>
<td>1,166,243</td>
<td>128</td>
<td>40</td>
</tr>
<tr>
<td>Wiki (Yang et al., 2015)</td>
<td>2,405</td>
<td>17,981</td>
<td>4,973</td>
<td>3</td>
</tr>
<tr>
<td>Computers (Shchur et al., 2018)</td>
<td>13,381</td>
<td>245,778</td>
<td>767</td>
<td>10</td>
</tr>
<tr>
<td>Photo (Shchur et al., 2018)</td>
<td>7,487</td>
<td>119,043</td>
<td>745</td>
<td>8</td>
</tr>
<tr>
<td>ogbn-products (Hu et al., 2020a)</td>
<td>2,449,029</td>
<td>61,859,140</td>
<td>100</td>
<td>47</td>
</tr>
<tr>
<td>Reddit (Hamilton et al., 2017)</td>
<td>232,965</td>
<td>114,615,892</td>
<td>602</td>
<td>41</td>
</tr>
</tbody>
</table>

D Experiment details

Dataset statistics: The datasets we employed encompass citation networks, web graphs, co-purchase networks, and social networks. The detailed dataset statistics are shown in Table 9.

Hardware Configuration: The experiments are conducted on Linux servers installed with an Intel(R) Xeon(R) Silver 4114 CPU @ 2.20GHz, 256GB RAM and 8 NVIDIA 2080Ti GPUs.

Software Configuration: Our models, as well as the DGI, GMI and GCN baselines, were implemented in PyTorch Geometric (Fey & Lenssen, 2019) version 1.4.3, DGL (Wang et al., 2019) version 0.5.1 with CUDA version 10.2, scikit-learn version 0.23.1 and Python 3.6. Our codes and datasets will be made available.

Hyper-parameters: For full batch training, we used 1-layer GCN as the encoder with prelu activation, for mini-batch training, we used a 3-layer GCN with prelu activation. We conducted grid search of different learning rate (from 1e-2, 5e-3, 3e-3, 1e-3, 5e-4, 3e-4, 1e-4) and curriculum settings (including learning rate decay and curriculum rounds) on the fullbatch version. We used 1e-3 or 5e-4 as the learning rate; 10,10,15 or 10,10,25 as the fanouts and 1024 or 512 as the batch size for mini-batch training.