Electronic Research Archive

https://www.aimspress.com/journal/era

ERA, 33(8): 5133–5157.
DOI: 10.3934/era.2025230
Received: 11 June 2025
Revised: 13 August 2025
Accepted: 26 August 2025
Published: 29 August 2025

Research article

Refining conflict in graph contrastive learning via pseudo-labels

Geng Tang, Qiguo Sun, Fengjun Zhang, Keyu Liu*and Xibei Yang

School of Computer, Jiangsu University of Science and Technology, Zhenjiang 212100, China

* **Correspondence:** Email: kyliu@just.edu.cn.

Abstract: Graph contrastive learning (GCL) has emerged as a powerful self-supervised paradigm for graph representation. A critical challenge in GCL arises from the conflict between the contrastive loss function and the message-passing mechanism of graph convolutional network encoders, as the message-passing mechanism pulls neighboring nodes close, while the contrastive loss function pushes negative nodes apart, especially neighboring nodes. Prevalent methods typically tackle the conflict by identifying conflicting node pairs and subsequently ignoring them during training. However, the identification of conflicting pairs in these methods may be one-sided and even incorrect, likely deleting valuable pairs which are inherently non-conflicting. To overcome the limitation, we have proposed conflict refined graph contrastive learning (CR-GCL), a novel framework that leverages pseudo-labels to prevent ignoring non-conflicting negative pairs. Technically, CR-GCL incorporates three key components: 1) a perturbation augmentation module that creates augmentation graphs and generates representations; 2) a conflict quantification module that estimates the gradient impact of negative pairs by tracking representation similarity changes; 3) a conflict refinement module that utilizes pseudo-labels to enhance the identification of conflicting negative pairs, which are then ignored during training. Extensive experiments on six benchmark datasets revealed that CR-GCL significantly outperforms state-of-the-art methods, delivering superior node classification accuracy with various label rates.

Keywords: graph contrastive learning; graph convolutional networks; node representation fusion; pseudo-label; semi-supervised learning

1. Introduction

Graph structure is a crucial and ubiquitous data expression across various domains like electronic recommendation [1], community detection [2], and protein molecule prediction [3]. Graph contrastive learning (GCL) [4,5] has emerged as a powerful self-supervised paradigm for graph representation. It has delivered impressive performances in link prediction [6], node classification [7], graph classification [8], and recommendation systems [9].

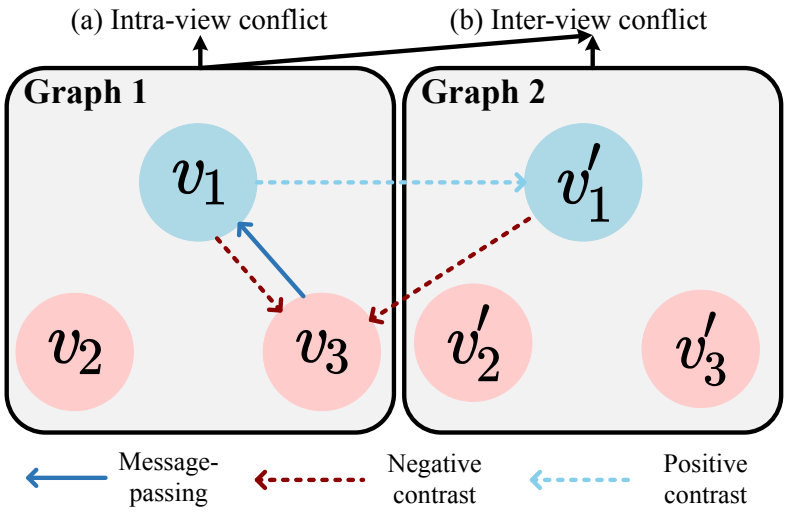
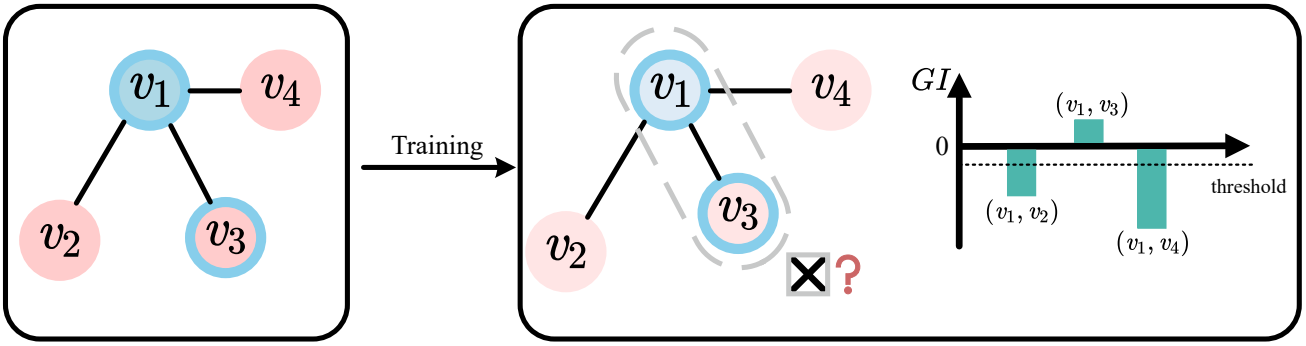


Figure 1. Conflict illustration. (a) Intra-view conflict occurs in a single view. For instance, v_1 aggregates features from its neighbor v_3, increasing their representation similarity. However, (v_1, v_3) is a negative pair, and the contrastive loss function aims to reduce the similarity, leading to a conflict. (b) Inter-view conflict arises across views. Here, (v_1, v'_1) is a positive pair, with the goal of maximizing the similarity. However, since (v_3, v'_1) forms a negative pair, the similarity of the positive pair (v_1, v'_1) may be unintentionally decreased when v_3 is aggregated into v_1, resulting in a conflict.

The core spirit of GCL is to maximize the similarity of positive pairs and minimize the similarity of negative pairs. It should be noticed that such a mechanism may cause conflict with graph convolutional network (GCN)-based encoders including intra-view conflict and inter-view conflict as presented in Figure 1. As reported in the existing literature [10, 11], the conflict lies between GCN encoders' message-passing mechanism and GCL's contrastive loss principle. Specifically, the message-passing mechanism pulls neighboring nodes close by aggregating their features, whereas the contrastive loss function pushes negative nodes apart, especially neighboring nodes. A detailed theoretical analysis of inter-view conflict is in Appendix A. To alleviate the conflict, existing solutions [10, 11] mostly depend on a simple threshold-based approach to detect and ignore the conflicting pairs whose conflict suspicions are measured by gradient impact (GI) during model training, which is approximately characterized with representation similarity change. Nevertheless, such a strategy is a one-stage approach so that the identification of conflicting pairs may be one-sided and even incorrect. As illustrated in Figure 2, existing methods usually identify pairs like (v_1, v_3) as conflicting since its GI exceeds the threshold; however, v_3 may suggest a significant homogeneous relationship, especially a positive pair relationship to v_1. Intuitively, in such a case, it is reasonable that GI between v_1 and v_3 is large, as the similarity between v_1 and v_3 is high. Therefore, in this paper, we argue that special pairs like (v_1, v_3) should not be ignored. If conflict identification can be refined to help perceive those pairs, then the final identification results would be more reliable for model training.

In this paper, we propose a framework called conflict refined graph contrastive learning (CR-GCL), which leverages pseudo-labels to enhance the identification of conflicting negative pairs, thereby enhancing training quality. Technically, CR-GCL consists of three key modules: the perturbation augmentation module, conflict quantification module, and conflict refinement module. The perturbation augmentation



(a) Original topology (b) Ignoring process

Figure 2. Limitation illustration. (a) In original topology, v_2, v_3 and v_4 are all adjacent to the target node v_1. (b) During training, (v_1, v_3) is identified as a conflicting pair.

module randomly masks features and drops edges simultaneously to generate augmentation graphs from the input graph. Moreover, a local-to-global approach fuses node representations from both the augmentation graphs and the input graph, enabling the model to learn high-quality representations. The conflict quantification module estimates the gradient impact of each negative pair by tracking the change in representation similarity across training epochs. In the conflict refinement module, by utilizing pseudo-labels, we enhance the identification of conflicting negative pairs and then ignore them during training. Additionally, to comprehensively process pairs throughout training, the ignored nodes undergo iterative updates at each optimization step. The key contributions of this work are summarized as follows.

1) We develop a high-quality pseudo-label generation strategy. We utilize a local-to-global fusion mechanism in the perturbation augmentation module to capture node representations that integrate local and global information. The representations guide the production of high-quality pseudo-labels, establishing a strong foundation for subsequent processing.

2) We design a conflicting pair refining mechanism. We leverage pseudo-labels to refine the conflicting pairs in the conflict refinement module. Pseudo-labels assist perceiving significant node pairs formed by homogeneous nodes, especially positive pairs, thereby preventing them from being ignored during training. This method improves the model's performance, and its effectiveness is confirmed through ablation studies.

3) We demonstrate the effectiveness of the proposed method with abundant comparative studies. Experimental results on six publicly available datasets highlight its superior performance.

2. Related works

2.1. Graph convolutional networks

Graph convolutional networks (GCNs) have demonstrated remarkable performances on modeling graph-structured data. Studies advancing GCNs can be broadly classified into four key categories [12]: sampling-based models [13], multi-scale information fusion-based models [14, 15], attention-based models [16, 17], and contrastive learning-based models [18].

Sampling-based models accelerate GCN training through node sampling and graph segmentation. For instance, the FastGCN [19] introduces importance sampling to select nodes with significant gradient contributions, while GRL [13] uses neighboring sampling to aggregate information from a fixed number of sampled neighbors for each node. The graph dropout self-learning hierarchical graph convolution network (DHGCN) [20] uses a self-learning hierarchical sampling strategy with graph dropout, and the progressive granular ball sampling fusion model (PGBSF) [21] proposes a granular-ball-based sampling framework to optimize training efficiency.

Multi-scale information fusion-based models aim to enrich node representations by integrating information across different scales. The adaptive multi-channel graph convolutional network (AM-GCN) [22] proposes an adaptive multi-channel fusion strategy, fusing the topology graph and a feature-based k-nearest neighbors graph. The mixed-order graph convolutional network (MOGCN) [23] fuses multi-order graph embeddings from different adjacency matrices using an ensemble strategy based on negative correlation learning. The sequential attention layer-wise fusion network (SLFNet) [24] applies sequential attention across layer-wise node embeddings within each view and then aggregates fused representations from all views. The cross-graph interaction network (GInterNet) [25] constructs learnable cross-graph topologies and performs inter-graph message-passing with attention-based interaction to fuse multiple graphs.

Attention-based models integrate attention mechanisms to optimize feature aggregation in GCNs. The graph attention network (GAT) [26] employs self-attention to weigh neighboring contributions, and the multi-View graph convolutional network with attention mechanism (MAGCN) [16] extends this with multi-head attention for broader relational modeling. The high-order graph attention network (HGRN) [17] focuses on high-order attention for long-range dependencies, whereas The collaborative graph neural network for augmented graphs (LoGo-GNN) [27] combines local and global information to prevent noise.

Contrastive learning-based models utilize self-supervised learning to reduce the reliance on labels. The graph contrastive representation learning with adaptive augmentation model (GCA) [18] introduces an adaptive augmentation strategy for contrastive graph learning, which perturbs less informative edges and feature dimensions while preserving important structural and semantic patterns, and the model graph contrastive coding (GCC) [28] proposes a structural pre-training framework based on subgraph instance discrimination, which enables the graph neural networks (GNNs) to capture universal structural patterns across multiple graphs and transfer them to diverse downstream tasks.

2.2. Graph contrastive learning

Graph contrastive learning (GCL) has emerged as a pivotal methodology in graph neural networks, significantly improving models' generalization capabilities through self-supervised representation learning. By generating augmentation graphs from original graphs, GCL enables the autonomous extraction of latent supervised information, thereby reducing dependency on manually annotated labels. A recent comprehensive survey systematically organizes the fundamental principles of GCL, including data augmentation, contrastive modes, and optimization objectives, signifying the field's maturation into a structured discipline with established principles [29]. Existing GCL methods can be broadly categorized into four major aspects [11]: augmentation strategy-based methods [8], negative sampling-based techniques [30], contrastive loss-based methods [31], and conflict addressing methods [10].

Augmentation strategy-based methods aim to improve contrastive learning by generating diverse

graph views that facilitate the learning of robust representations. GraphCL [8] employs a set of handcrafted augmentations including node dropping, edge perturbation, attribute masking, and subgraph sampling to generate diverse graph views and enforce representation invariance. GCA [18] designs an adaptive augmentation strategy that selectively perturbs less informative edges and features based on node centrality, thereby preserving the most critical structural and semantic patterns during augmentation. To move beyond handcrafted or adaptive strategies, recent work has focused on making the augmentation process itself learnable; for instance, the graph pooling contrast (GPS) method [32] proposes an innovative framework that utilizes learnable graph pooling to automatically generate multi-scale augmented views, employing an adversarial training scheme to produce challenging yet semantically consistent positive samples without relying on predefined heuristics.

Negative sampling-based methods focus on designing sampling techniques that effectively distinguish positive and negative pairs. The deep graph information maximization (DGI) model [30] employs a corruption-based negative sampling strategy, where node features are shuffled to construct negative samples while preserving the original graph structure. ProGCL [33] introduces a probability-guided hard negative sampling strategy by modeling similarity distributions with a beta mixture model, effectively reducing the inclusion of false negatives.

Contrastive loss-based methods are primarily concerned with developing loss functions that encourage semantic consistency across augmentation views while increasing the models' ability to distinguish positive samples from negative samples. The deep graph contrastive representation learning (GRACE) model [31] adopts a full-graph loss that encourages alignment of node embeddings across augmentation views. The general graph augmentation (GAME) model [34] retains the information noise contrastive estimation (InfoNCE) loss but reveals that it inherently captures low-frequency invariance, and proposes a spectral augmentation strategy that maximizes high-frequency differences to enhance contrastive effectiveness.

Conflict-addressing methods aim to identify and mitigate conflicting samples that may hinder model performance. Both partial ignored graph contrastive learning (PiGCL) model [10] and ReGCL [11] identify conflict samples by analyzing gradients. PiGCL approximates gradient confusion by monitoring the similarity dynamics of negative pairs during training, while ReGCL directly analyzes the contribution of samples to conflicting gradient directions and mitigates them through structure learning and loss reweighing.

Beyond these foundational pillars, the principles of GCL are increasingly being adapted to solve more complex, data-efficient learning paradigms where label information is scarce or ambiguous. For example, DualGraph [35] integrates GCL into a dual learning framework for semi-supervised graph classification, where a contrastive loss is employed to enforce intra-module consistency between original and augmented graph views. To tackle the challenge of partial label learning, the distribution divergence-based graph contrast (DEER) [36] model introduces a novel method for identifying positive pairs by measuring the distribution divergence between multiple sets of augmented views, thus adapting GCL to handle ambiguous label sets. Extending GCL to the zero-shot setting, the graph contrastive mbedding network (GraphCEN) [37] introduces a two-level contrastive mechanism that jointly learns both node and class embeddings, enabling knowledge transfer from seen to unseen classes.

These methods fully expand the possibilities of GCL, offering diverse strategies for improving training effectiveness and representation quality. In particular, the conflict mitigation strategies proposed by PiGCL and ReGCL offer valuable perspectives for our work.

3. CR-GCL

3.1. Preliminaries

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ represent a graph, where $\mathcal{V} = \{v_1, v_2, \ldots, v_N\}$ denotes the set of N nodes and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ signifies the set of edges. x_i is the feature vector of node v_i, $\boldsymbol{X} = [x_1, x_2, \ldots, x_N] \in \mathbb{R}^{N \times d}$ is the feature matrix, and d is the dimension of the feature. We define the adjacency matrix as $\boldsymbol{A} \in \{0, 1\}^{N \times N}$, where $\boldsymbol{A}_{ij} = 1$ if $(v_i, v_j) \in \mathcal{E}$, and $\boldsymbol{A}_{ij} = 0$ otherwise. $\boldsymbol{D}$ represents the degree matrix. $\tilde{\boldsymbol{A}} = \boldsymbol{A} + \boldsymbol{I}$ is the adjacency matrix for a graph with self-connection, and $\tilde{\boldsymbol{D}} = \boldsymbol{D} + \boldsymbol{I}$ is the degree matrix of $\tilde{\boldsymbol{A}}$, where $\boldsymbol{I}$ is the unit matrix. $\hat{\boldsymbol{A}} = \tilde{\boldsymbol{D}}^{-1/2}\tilde{\boldsymbol{A}}\tilde{\boldsymbol{D}}^{-1/2}$ denotes the symmetrically normalized adjacency matrix with a self-connection. C is the number of classes, $\mathcal{V}_U$ represents the set of unlabeled nodes, and $\mathcal{V}_L$ denotes the set of labeled nodes. $\boldsymbol{Y} \in \mathbb{R}^{|\mathcal{V}_L| \times C}$ is the label indicator matrix. Our goal is to learn a GCN encoder that generates node embeddings $\boldsymbol{H}$ in low dimensionality. In this paper, a two-layer perceptron (MLP) serves as the projector $f(\cdot)$. $Aug_1(\cdot) \sim \mathcal{T}$ and $Aug_2(\cdot) \sim \mathcal{T}$ are two augmentation functions, where $\mathcal{T}$ is the set of all perturbation augmentation functions.

3.2. Framework overview

The overall structure of CR-GCL is illustrated in Figure 3. Our framework consists of the following key modules:

1) Perturbation augmentation: This module adopts random feature masking and edge dropping simultaneously to generate augmentation graphs which are subsequently fed into a GCN encoder to capture node representations. Local and global perspectives are exploited jointly to fuse the representations.

2) Conflict quantification: This module estimates the influence of negative pairs by tracking representation similarity changes across training epochs, with both intra-view and inter-view conflicts taken into account.

3) Conflict refinement: This module uses pseudo-labels to enhance the identification of conflicting negative pairs and subsequently ignores them from training.

3.3. Perturbation augmentation

In this module, we randomly mask features and drop edges of the input graph simultaneously to create two augmentation graphs, denoted as $\mathcal{G}_1$ and $\mathcal{G}_2$.

The typical design of GCN encoders is centered on a message-passing mechanism [38]. CR-GCL uses a two-layer GCN encoder to generate node representations. The simple formula of the GCN encoder can be expressed as:

$$\boldsymbol{H} = \sigma(\hat{\boldsymbol{A}} ReLU(\hat{\boldsymbol{A}}\boldsymbol{X}\boldsymbol{W}^{(0)})\boldsymbol{W}^{(1)}) \tag{3.1}$$

where $\boldsymbol{W}^{(0)}$ and $\boldsymbol{W}^{(1)}$ denote the trainable weight matrix of the first and second layer of the GCN, respectively. σ and $ReLU$ are nonlinear activation functions. The output of the two-layer GCN encoder is represented by $\boldsymbol{H}$.

The input graph and two augmentation graphs are fed into the encoder and then the encoder generates embeddings $\boldsymbol{H}^{\text{raw}}$, $\boldsymbol{H}^{\text{aug}_1}$, and $\boldsymbol{H}^{\text{aug}_2}$, where $\boldsymbol{H}^{\text{raw}}$ is the embedding of the input graph and $\boldsymbol{H}^{\text{aug}_1}$ and $\boldsymbol{H}^{\text{aug}_2}$ are embeddings of augmentation graphs.

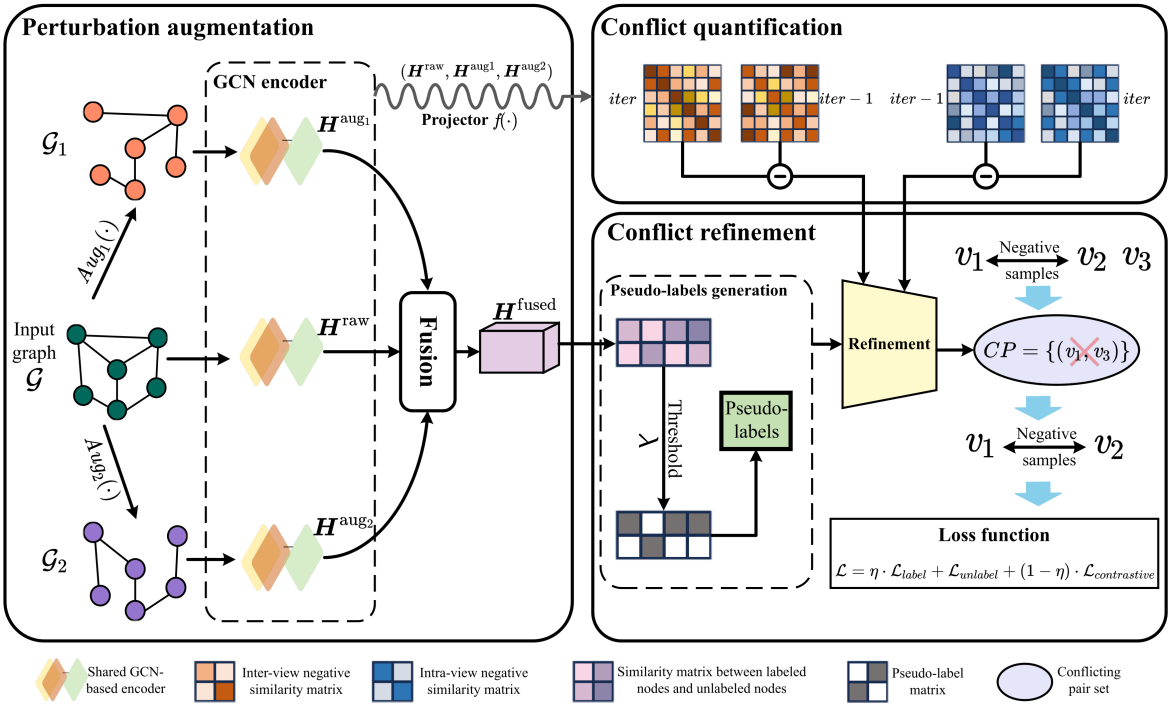


Figure 3. Overview of the CR-GCL framework. The CR-GCL framework starts with augmenting an input graph $\mathcal{G}$ into two views, $\mathcal{G}_1$ and $\mathcal{G}_2$, using random feature masking and edge dropping simultaneously. A shared GCN encoder is applied to generate embeddings: $\boldsymbol{H}^{\text{raw}}$ from the input graph, $\boldsymbol{H}^{\text{aug}_1}$ and $\boldsymbol{H}^{\text{aug}_2}$ from the augmentation views, which are then fused into $\boldsymbol{H}^{\text{fused}}$. The conflict quantification module computes similarity matrices for inter-view and intra-view negative pairs, comparing them with the previous iteration to analyze their gradient information. Subsequently, the conflict refinement module leverages $\boldsymbol{H}^{\text{fused}}$ to generate pseudo-labels, which are then used to identify and refine the conflicting pairs. The pairs identified as conflicting constitute the conflicting pair set CP, and are ignored from training in the current iteration. The model is trained and optimized using a global loss function that integrates losses from labeled nodes, unlabeled nodes, and a contrastive loss term.

Considering the presence of potential noise from unreliable data sources and limited graph augmentation techniques, we adopt a local-to-global fusion strategy, shown in Figure 4. From a global perspective, the embeddings $\boldsymbol{H}^{\text{raw}}$, $\boldsymbol{H}^{\text{aug}_1}$, and $\boldsymbol{H}^{\text{aug}_2}$ are fused by taking the average, denoted as $\boldsymbol{F}_1 = \frac{1}{3}(\boldsymbol{H}^{\text{raw}} + \boldsymbol{H}^{\text{aug}_1} + \boldsymbol{H}^{\text{aug}_2})$. From a local perspective, embeddings are fused pairwise, resulting in three embeddings: $\boldsymbol{F}_2 = \frac{1}{2}(\boldsymbol{H}^{\text{raw}} + \boldsymbol{H}^{\text{aug}_1})$, $\boldsymbol{F}_3 = \frac{1}{2}(\boldsymbol{H}^{\text{raw}} + \boldsymbol{H}^{\text{aug}_2})$, and $\boldsymbol{F}_4 = \frac{1}{2}(\boldsymbol{H}^{\text{aug}_1} + \boldsymbol{H}^{\text{aug}_2})$.

To improve the model's ability to identify important features, we integrate an attention mechanism [22], allowing for a richer and more detailed semantic understanding.

$$(\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2, \boldsymbol{\alpha}_3, \boldsymbol{\alpha}_4) = att(\boldsymbol{F}_1, \boldsymbol{F}_2, \boldsymbol{F}_3, \boldsymbol{F}_4) \tag{3.2}$$

where $\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2, \boldsymbol{\alpha}_3, \boldsymbol{\alpha}_4$ indicate the attention values of N nodes with embeddings $\boldsymbol{F}_1, \boldsymbol{F}_2, \boldsymbol{F}_3, \boldsymbol{F}_4$, respectively. The attention value for the j-th node in the i-th embedding is calculated as:

$$\boldsymbol{\alpha}_{ij} = softmax(\omega_{ij}) \tag{3.3}$$

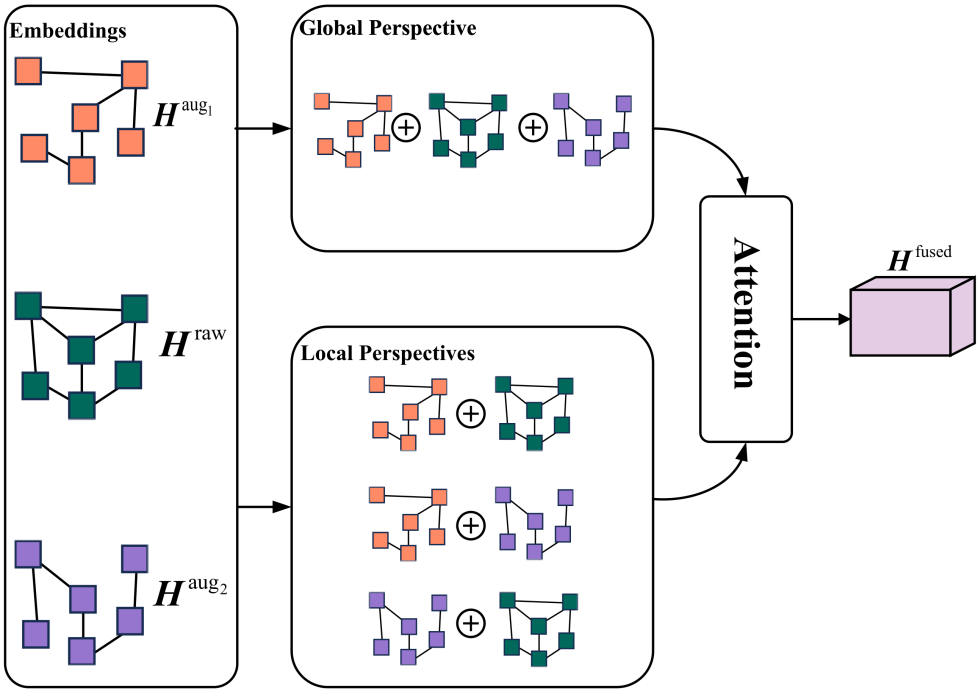


Figure 4. Illustration of the local-to-global fusion strategy.

where the unnormalized attention value ω_{ij} is computed as:

$$\omega_{ij} = \boldsymbol{q}_i^T \cdot tanh(\boldsymbol{\theta}_{att} \cdot (\boldsymbol{f}_{ij})^T + \boldsymbol{b}_i) \tag{3.4}$$

where, $\boldsymbol{q}_i$ is a shared attention vector for the i-th embedding, $\boldsymbol{\theta}_{att}$ is the weight matrix, and $\boldsymbol{b}_i$ is the bias vector. $\boldsymbol{f}_{ij} \in \mathbb{R}^{F_h}$ represents the feature vector of the j-th node in the i-th embedding matrix $\boldsymbol{F}_i$.

Finally, the fused embedding $\boldsymbol{H}^{\text{fused}}$ is aggregated as follows:

$$\boldsymbol{H}^{\text{fused}} = \boldsymbol{\alpha}_1 \cdot \boldsymbol{F}_1 + \boldsymbol{\alpha}_2 \cdot \boldsymbol{F}_2 + \boldsymbol{\alpha}_3 \cdot \boldsymbol{F}_3 + \boldsymbol{\alpha}_4 \cdot \boldsymbol{F}_4$$

3.4. Conflict quantification

In order to address the conflict, the first step is to identify the involved negative pairs and adopt an effective method to quantify their influence on model training. In this study, we develop a conflict quantification module capable of indirectly assessing the impact of negative pairs and identifying those responsible for the conflict. Specifically, the module aims to measure the effect of introducing a particular negative sample on the gradient of the encoder, formulated as follows:

$$\nabla \boldsymbol{W} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{W}} \tag{3.5}$$

where $\mathcal{L}$ represents the contrastive loss and $\boldsymbol{W}$ are the encoder parameters. Negative pairs are not independent: their combined effect must be calculated alongside other negative pairs at the same time. However, directly calculating $\nabla \boldsymbol{W}$ for specific negative pairs using the above formula poses significant mathematical challenges. To this end, we use an iterative solution that approximates the impact of

negative pairs upon the gradient by quantifying the change of representation similarities they produce while training. It is depicted by:

$$GI_{NP(i,j)} = S_{ij}^{iter} - S_{ij}^{iter-1}, \quad iter \geq 1, \tag{3.6}$$

where $NP(i, j)$ represents the negative pair formed by v_i and v_j, and $GI_{NP(i,j)}$ is the gradient impact of $NP(i, j)$. S_{ij}^{iter} denotes the cosine similarity between the representations of v_i and v_j at the current iteration. We track the similarity S_{ij}^{iter} every few epochs. When a negative pair is not involved in the conflict, its similarity tends to drop quickly, resulting in a smaller $GI_{NP(i,j)}$. On the other hand, if the similarity does not decrease as expected or even increases, $GI_{NP(i,j)}$ grows larger, suggesting a higher chance of contributing to the conflict.

We apply this strategy to both negative pairs involved in the intra-view conflict and those contributing to the inter-view conflict. For the intra-view conflict, we compute $GI_{NP(i,j)}^{intra}$ as follows:

$$GI_{NP(i,j)}^{intra} = S_{ij}^{iter} - S_{ij}^{iter-1}, \quad iter \geq 1, \tag{3.7}$$

where representations of node v_i and node v_j are both from $\boldsymbol{H}^{\text{raw}}$.

For the inter-view conflict, we compute $GI_{NP(i,j)}^{inter}$, which is depicted by:

$$GI_{NP(i,j)}^{inter} = S_{ij}^{iter} - S_{ij}^{iter-1}, \quad iter \geq 1, \tag{3.8}$$

where the representation of node v_i is from $\boldsymbol{H}^{\text{aug}_1}$ while the representation of node v_j is from $\boldsymbol{H}^{\text{aug}_2}$.

3.5. Conflict refinement

Previous work [10] suggested that if the similarity of two nodes did not decrease significantly during training, they should be identified as a conflicting pair. However, it is worth noting that among the identified conflicting negative samples, some nodes may exhibit high similarity with the target node. In such cases, a significant decrease in similarity is not expected, as these nodes are likely to have important homogeneous relationships with the target node. This suggests that they may not be true negatives, and in some cases, could even represent potential positive samples. Therefore, such nodes should not be ignored during training.

To address this issue, we introduce pseudo-labels. If two nodes' representation similarities are high, their pseudo-labels tend to be consistent, thus they should not be regarded as a conflicting pair. Inspired by [39], the element m_{ij} in cosine similarity matrix $\boldsymbol{M} \in \mathbb{R}^{N\times N}$ can be calculated by fused embedding $\boldsymbol{H}^{\text{fused}}$, which can be formulated as follows:

$$m_{ij} = \frac{\boldsymbol{H}_i^{\text{fused}} \cdot \boldsymbol{H}_j^{\text{fused}}}{\|\boldsymbol{H}_i^{\text{fused}}\| \cdot \|\boldsymbol{H}_j^{\text{fused}}\|} \tag{3.9}$$

For an unlabeled data sample v_i, its pseudo-label assignment is determined by comparing the similarity between all labeled samples. Specifically, the cosine similarity m_{ij} is computed between v_i and each labeled sample v_l. A higher m_{il} indicates closer directional alignment in the vector space, suggesting a greater likelihood that v_i and v_l belong to the same class. After iterating through all labeled samples, the model identifies the highest m_{il} value, and the class label of the corresponding labeled

sample v_l is assigned as the pseudo-label for v_l. To ensure the reliability of pseudo-labels, a threshold λ is applied to select the node set $\mathcal{V}_{PL}$ for pseudo-label assignment:

$$\mathcal{V}_{PL} = \{v_i \in \mathcal{V}_U \mid \max_{l:v_l \in \mathcal{V}_L} m_{il} \geq \lambda\} \tag{3.10}$$

For each unlabeled node $v_i \in \mathcal{V}_{PL}$, its pseudo-label $\hat{y}_i$ is assigned based on the most similar labeled node:

$$\hat{y}_i = y_{l^*},\ l^* = \arg\max_{l:v_l \in \mathcal{V}_L} m_{il} \tag{3.11}$$

Thus, we construct the conflicting pair set CP based on the ignore ratio r, formulated as:

$$\begin{aligned} CP_{inter} &= \{NP(i,j) \mid GI^{inter}_{NP(i,j)} > GI^{inter}_{r}\}, \\ CP_{intra} &= \{NP(i,j) \mid GI^{intra}_{NP(i,j)} > GI^{intra}_{r}\}, \\ PL &= \{NP(i,j) \mid \hat{y}_i \neq \hat{y}_j\}, \\ CP &= (CP_{inter} \cup CP_{intra}) \cap PL \end{aligned} \tag{3.12}$$

where, GI^{inter}_r refers to the minimum gradient impact among the top r inter-view negative pairs when sorted in descending order. Similarly, GI^{intra}_r is defined as the minimum gradient impact among the top r intra-view negative pairs when sorted in descending order. The negative pairs in CP are ignored during training.

The conflicting pair set CP is also updated dynamically throughout training. In the early stages, the node pairs within CP exhibit strong conflict and are therefore temporarily ignored to avoid negative effects on representation learning. As training proceeds, some negative pairs that have already contributed sufficient information may no longer exhibit significant similarity changes, and are then added into CP. Meanwhile, previously ignored pairs can be reintroduced into the training process. This dynamic scheduling ensures that all negative samples are eventually processed during training. The time complexity analysis is shown in Appendix B.

3.6. Loss function

CR-GCL defines a global loss function denoted as $\mathcal{L}$ to facilitate the end-to-end training. It is made up of labeled data loss $\mathcal{L}_{label}$, unlabeled data loss $\mathcal{L}_{unlabel}$, and contrastive loss $\mathcal{L}_{contrastive}$.

3.6.1. Labeled data loss

For semi-supervised classification tasks, we adopt the cross-entropy error as the labeled data loss function:

$$\mathcal{L}_{label} = -\sum_{v_i \in \mathcal{V}_L} \sum_{j=1}^{C} \boldsymbol{Y}_{ij} \log(\tilde{\boldsymbol{Y}}_{ij}) \tag{3.13}$$

where, $\mathcal{V}_L$ represents the labeled nodes and $\tilde{\boldsymbol{Y}}$ is the predicted label matrix.

3.6.2. Unlabeled data loss

Similar to labeled data loss, unlabeled data loss also employs the cross-entropy error, formulated as:

$$\mathcal{L}_{unlabel} = -\sum_{v_i \in \mathcal{V}_{PL}} \sum_{j=1}^{C} \hat{\boldsymbol{Y}}_{ij} \log(\tilde{\boldsymbol{Y}}_{ij}) \tag{3.14}$$

$\hat{\boldsymbol{Y}}$ is the assigned pseudo-label matrix.

3.6.3. Contrastive loss

We choose InfoNCE as our contrastive loss, formulated as:

$$\mathcal{L}_{InfoNCE} = -\frac{1}{N}\sum_{i=1}^{N}\log\left(\frac{e^{\psi(\boldsymbol{H}_i^{\text{aug}_1},\boldsymbol{H}_i^{\text{aug}2})/\tau}}{\sum_{j=1}^{N}e^{\psi(\boldsymbol{H}_i^{\text{aug}_1},\boldsymbol{H}_j^{\text{aug}_2})/\tau}}\right) \tag{3.15}$$

where τ is a hyper-parameter, $\psi(\boldsymbol{H}_i^{\text{aug}_1},\boldsymbol{H}_i^{\text{aug}_2}) = Sim(f(\boldsymbol{H}_i^{\text{aug}_1}), f(\boldsymbol{H}_i^{\text{aug}_2}))$, where $Sim(\cdot,\cdot)$ is the cosine similarity calculation, and $f(\cdot)$ is a non-linear projector.

Notice that when partial nodes are ignored, the corresponding gradient will also decrease. Moreover, this numerical attenuation effect becomes significantly amplified through the compounding nature of exponential computations and logarithmic operations involved in the process. To systematically quantify this effect, we introduced the dilation coefficient γ as a critical parameter:

$$\gamma = \frac{\sum_{i=1}^{N}\sum_{j=1}^{N}\left(e^{\psi(\boldsymbol{H}_i^{\text{raw}},\boldsymbol{H}_j^{\text{raw}})} + e^{\psi(\boldsymbol{H}_i^{\text{aug}_1},\boldsymbol{H}j^{\text{aug}2})}\right)}{\sum_{i=1}^{N}\sum_{j=1,NP(i,j)\notin CP}^{N}\left(e^{\psi(\boldsymbol{H}_i^{\text{raw}},\boldsymbol{H}_j^{\text{raw}})} + e^{\psi(\boldsymbol{H}_i^{\text{aug}_1},\boldsymbol{H}_j^{\text{aug}_2})}\right)} \tag{3.16}$$

To simplify the formula, we define pos_i and neg_i as follows:

$$pos_i = e^{\psi(\boldsymbol{H}_i^{\text{aug}_1},\boldsymbol{H}_i^{\text{aug}_2})/\tau} \tag{3.17}$$

$$neg_i = \sum_{j=1,NP(i,j)\notin CP}^{N}\left(e^{\psi(\boldsymbol{H}_i^{\text{aug}_1},\boldsymbol{H}_j^{\text{aug}_1})/\tau} + e^{\psi(\boldsymbol{H}_i^{\text{aug}_1},\boldsymbol{H}_j^{\text{aug}_2})/\tau}\right) \tag{3.18}$$

To handle the negative samples, we multiply the signs of their gradients by γ. For positive pairs, we introduce a compensatory parameter ϕ to balance the learning dynamics, implemented through the following adjustment:

$$\phi = \left(1 - \frac{1}{\gamma}\right)\cdot\frac{1}{N}\sum_{i=1}^{N}\psi(\boldsymbol{H}_i^{\text{aug}_1},\boldsymbol{H}_i^{\text{aug}_2}) \tag{3.19}$$

By introducing gradient compensation mechanisms, the proposed algorithm maintains steady intensities under significant negative sample sparsity. This adaptive compensation reduces the risks of premature convergence resulting from gradient vanishing when using large r, ultimately leading to our optimal contrastive loss formulation:

$$\mathcal{L}_{contrastive} = -\frac{1}{N}\sum_{i=1}^{N}\log\left(\frac{pos_i}{pos_i + \gamma\cdot neg_i}\right) - \phi \tag{3.20}$$

To control the correlation between the label loss and the contrastive loss, we employ a hyper-parameter η. Therefore, the final loss function of CR-GCL is:

$$\mathcal{L} = \eta\cdot\mathcal{L}_{label} + \mathcal{L}_{unlabel} + (1-\eta)\cdot\mathcal{L}_{contrastive} \tag{3.21}$$

4. Experiments

This section evaluates our proposed model against several state-of-the-art graph-based semi-supervised node classification approaches, followed by ablation studies to examine the contributions of key components in CR-GCL.

4.1. Configuration

4.1.1. Datasets

We conduct comprehensive tests on six benchmark datasets: three citation networks (Cora, CiteSeer, PubMed) [40], the ACM academic network [41], and two Wikipedia-derived graphs (Squirrel, Chameleon) [42]. Among them, Cora, CiteSeer, PubMed, and ACM are homogeneous graph datasets, and Chameleon, and Squirrel are heterogeneous graph datasets. Their statistical details are provided in Table 1.

Table 1. Dataset description.

Datasets	Nodes	Edges	Features	Classes	Training
Cora	2708	5429	1433	7	14/21/28/140
CiteSeer	3327	4732	3703	6	12/18/36/120
PubMed	19,717	44,338	500	3	9/15/30/60
ACM	3025	13,128	1870	3	9/15/30/60
Chameleon	2277	36,101	2325	5	10/20/35/100
Squirrel	5201	217,073	2089	5	10/15/50/100

4.1.2. Baselines

Our comparative analysis spans five categories of graph neural networks: (i) foundational architectures including GCN [40] and GFNN [43]; (ii) attention-enhanced models like GAT [26], DGCN [44], and MHMOGAT [45]; (iii) multi-scale/view information fusion frameworks such as MixHop [15], N-GCN [14], MOGCN [23], and LA-GCN [46]; (iv) contrastive learning approaches represented by PA-GCN [12] and LoGo-GNN [27]; (v) conflict-addressing-based methods such as PiGCL [10] and ReGCL [11]. To ensure comparability with established evaluation protocols [40, 47], we ran the proposed model 20 times and recorded the average accuracy. For baseline methods, we reproduced results using their official implementations under identical experimental conditions.

4.1.3. Experiment setting

We evaluate the effectiveness of CR-GCL in the semi-supervised node classification task. The compared baselines, as detailed above, include a mixture of architectures ranging from conventional GNNs to recent conflict-aware methods. Among them, ReGCL and PiGCL are self-supervised methods, while the remaining baselines and CR-GCL itself operate under semi-supervised conditions. Key hyperparameters specific to each dataset are detailed in Table 2. The architecture employs a two-layer multilayer perceptron (MLP) as the projection function. Full-batch gradient descent is utilized throughout the training epochs, with implementation carried out in PyTorch and optimization managed

through the Adam optimizer. For dataset splitting, varying quantities of labeled nodes per class are selectively retained for training, while the remaining nodes constitute the test set.

Table 2. Hyperparameter specifications.

Datasets	Learning rate	Weight decay	Training epochs	Hidden sizes	Dropout
Cora	0.0005	1.00E-05	500	128	0.5
CiteSeer	0.0005	1.00E-05	150	128	0.5
PubMed	0.001	1.00E-04	1500	128	0.5
ACM	0.0005	1.00E-05	500	128	0.5
Chameleon	0.0005	1.00E-05	500	128	0.5
Squirrel	0.0005	1.00E-05	500	128	0.5

In addition, λ is set at the range of $[0.9, 1)$ in order to ensure the quality of generated pseudo-labels. In our experiment, we perform an analysis on the balance parameter η and ignore ratio r.

4.2. Experiment results

Table 3 presents the node classification accuracy (ACC) of CR-GCL and various baseline methods across six benchmark datasets, where **OURS** indicates the performance of CR-GCL. ReGCL and PiGCL are evaluated under a self-supervised setting, while all other methods are under a semi-supervised setting. The 'Input' refers to data we can obtain for training, where $\boldsymbol{X}$, $\boldsymbol{A}$, and $\boldsymbol{Y}$ denote the feature matrix, adjacency matrix, and label matrix, respectively. CR-GCL achieves superior performance, obtaining the best results on three datasets and the second-best on the remaining ones. This validates the effectiveness of CR-GCL's conflict refinement strategy in improving training quality.

To further evaluate CR-GCL's performance, we conduct experiments under varying label rates, as shown in Table 4. The 'Average' metric denotes the average classification accuracy across varying numbers of training samples for each dataset.

We observe the following results:

(1) Compared to the baseline described above, CR-GCL achieves the best 'Average' results on all six datasets. Notably, CR-GCL shows a significant advantage on the Cora, PubMed, Chameleon, and Squirrel datasets.

(2) The proposed method exhibits remarkable advantages in low-label-rate learning scenarios. For example, when trained with only 15 labeled nodes per class on the ACM dataset, our approach achieves 82.8% classification accuracy—a significant 4.2% improvement over MHMOGAT's second-best performance of 78.6%, establishing new state-of-the-art results in low-labeled learning settings. This superiority can be attributed to CR-GCL's contrastive learning foundation, which inherently requires fewer labels due to its self-supervised nature. In addition, the pseudo-label-based conflict refinement strategy helps prevent non-conflicting negative pairs from being ignored during training, thereby improving learning quality. Furthermore, the local-to-global fusion strategy enhances node representations under limited labels, contributing to overall performance gains.

(3) In some cases CR-GCL does not outperform other models, as shown in Table 4. Taking the LoGo-GNN and PA-GCN as examples, our results are weaker than theirs in several datasets when the label rate is low. This can be attributed to the fact that the LoGo-GNN and PA-GCN both have designed unique augmentation strategies to enhance node representations while CR-GCL only randomly masks

Table 3. ACC (%) of node classification tasks. (Bold denotes the best result; underline indicates the second-best.)

Method	Input	Cora	CiteSeer	PubMed	ACM	Chameleon	Squirrel
ReGCL	$\boldsymbol{X}, \boldsymbol{A}$	<u>84.8</u>	71.2	**84.5**	91.4	44.2	29.3
PiGCL	$\boldsymbol{X}, \boldsymbol{A}$	84.6	71.7	83.2	90.9	<u>53.6</u>	**36.5**
GCN	$\boldsymbol{X}, \boldsymbol{A}, \boldsymbol{Y}$	81.7	70.4	79.0	87.8	47.6	25.2
GFNN	$\boldsymbol{X}, \boldsymbol{A}, \boldsymbol{Y}$	81.9	69.6	80.7	86.8	47.4	30.5
GAT	$\boldsymbol{X}, \boldsymbol{A}, \boldsymbol{Y}$	83.2	72.6	79.0	87.4	27.9	31.6
PA-GCN	$\boldsymbol{X}, \boldsymbol{A}, \boldsymbol{Y}$	83.6	70.4	79.3	90.0	49.0	29.1
MOGCN	$\boldsymbol{X}, \boldsymbol{A}, \boldsymbol{Y}$	82.4	72.4	79.2	90.1	46.9	31.4
N-GCN	$\boldsymbol{X}, \boldsymbol{A}, \boldsymbol{Y}$	83.0	72.2	79.2	88.0	48.9	30.6
MixHop	$\boldsymbol{X}, \boldsymbol{A}, \boldsymbol{Y}$	81.9	71.4	80.8	87.4	40.6	30.6
DGCN	$\boldsymbol{X}, \boldsymbol{A}, \boldsymbol{Y}$	84.1	73.3	80.6	90.2	49.6	27.9
LA-GCN	$\boldsymbol{X}, \boldsymbol{A}, \boldsymbol{Y}$	84.6	**74.7**	81.7	89.9	48.3	31.2
MHMOGAT	$\boldsymbol{X}, \boldsymbol{A}, \boldsymbol{Y}$	80.3	68.0	77.2	82.4	45.7	23.4
LoGo-GNN	$\boldsymbol{X}, \boldsymbol{A}, \boldsymbol{Y}$	84.6	73.4	81.6	<u>91.8</u>	52.7	29.4
OURS	$\boldsymbol{X}, \boldsymbol{A}, \boldsymbol{Y}$	**85.1**	<u>73.7</u>	<u>83.7</u>	**92.0**	**54.1**	<u>32.5</u>

features and edges. What is more, with a standard label rate, CR-GCL cannot outperform the LA-GCN. The LA-GCN uses learnable local feature augmentation based on the raw graph relationships and focuses more on enhancing information from feature perspectives. Overall, CR-GCL's performances are still superior to other methods because CR-GCL not only utilized supervised and self-supervised information, but also tackled the inherent conflict in GCL.

4.3. Ablation study

We conducted an ablation study to figure out whether each component of our CR-GCL framework really works. Basically, we made three variants of the model, each excluding one key component, and tested them against the full thing. We tested on six datasets and tracked the average classification accuracy (ACC%) results in Table 5.

1) CR-GCL-L2G: No Local-to-Global Fusion. This variant skips the local-to-global fusion strategy and just uses the attention mechanism to mix the original and augmentation graphs. Accuracy took a hit everywhere. On Cora, it only drops a bit, from 72.4% to 71.2%, but on Chameleon, it decreases from 40.9% to 38.7%. That sharp decline on Chameleon tells us this fusion step is a big deal, especially for complex graph structures.

2) CR-GCL-InterVC: Dropping Inter-View Conflict Handling. In this variant, we exclude the identification of inter-view conflicting negative pairs, limiting the conflict quantification module to only identify intra-view negative pairs within a single view. As shown in Table 5, performance decreases across all datasets. For instance, PubMed's ACC drops from 73.3% to 69.7%, and Cora declines from 72.4% to 69.7%. This degradation validates the presence of inter-view conflict, where negative pairs across augmentation views can interfere with the optimization of positive pairs. By capturing inter-view conflict, CR-GCL expands the scope of conflict detection, effectively mitigating a broader range of conflicting negative pairs that would otherwise degrade model performance.

Table 4. Node classification accuracy (%) under different label rates. (Bold denotes the best result; underline indicates the second-best.)

Datasets	Training	GCN	GFNN	GAT	PA-GCN	MOGCN	N-GCN	MixHop	DGCN	LA-GCN	MHMOGAT	LoGo-GNN	**OURS**
	14	43.8	37.8	44.5	47.6	41.9	40.8	31.9	50.6	53.3	61.4	60.3	**62.7**
	21	50.9	48.9	55.6	56.7	62.8	52.4	51.2	54.8	62.5	62.8	58.8	**65.3**
Cora	28	62.3	60.3	66.8	72.1	71.5	61.8	62.9	68.2	73.4	68.5	62.6	**76.1**
	140	81.7	81.9	83.2	83.6	82.4	83.0	81.9	84.1	84.6	80.3	84.6	**85.1**
	Average	59.7	57.2	62.5	65.0	64.7	59.5	57.0	64.4	68.5	68.3	66.6	**72.3**
	12	24.7	36.3	33.0	28.9	28.1	20.5	20.2	35.4	30.0	**49.1**	43.5	48.6
	18	43.6	40.8	58.4	**60.4**	58.9	47.2	44.2	54.9	53.4	53.2	57.9	58.7
CiteSeer	36	55.3	54.1	61.1	64.7	62.8	58.4	56.7	57.4	62.6	59.7	65.6	**68.7**
	120	70.4	69.6	72.6	70.4	72.4	72.2	71.4	73.3	**74.7**	68.0	73.4	73.7
	Average	48.5	50.2	56.3	56.1	55.6	49.6	48.1	55.3	55.2	57.5	60.1	**62.4**
	9	43.9	35.5	42.0	51.0	40.2	39.9	39.9	51.7	53.6	**68.2**	63.5	64.1
	15	60.5	62.5	56.6	63.5	63.2	62.4	61.6	68.6	65.0	68.0	**73.5**	73.3
PubMed	30	57.5	54.3	70.7	73.5	68.5	58.5	59.1	73.8	67.6	74.6	69.9	**77.5**
	60	79.0	80.7	79.0	79.3	79.2	79.2	80.8	80.6	81.7	77.2	81.6	**83.7**
	Average	60.2	58.3	62.1	66.8	62.8	62.8	60.4	68.7	67.0	72.0	72.1	**74.7**
	9	51.8	50.2	56.3	56.7	56.6	54.4	34.5	62.3	65.3	**72.1**	68.0	65.1
	15	65.6	62.3	64.6	69.7	68.1	64.3	41.0	73.3	68.9	78.6	72.0	**82.8**
ACM	30	79.6	74.3	80.0	87.2	87.5	76.5	61.9	85.8	86.6	81.1	87.9	**89.7**
	60	87.8	86.8	87.4	90.0	90.1	88.0	87.4	90.2	89.9	82.4	91.8	**92.0**
	Average	71.2	68.4	72.1	76.1	75.6	70.8	56.2	77.9	77.7	78.6	79.9	**82.4**
	10	23.9	25.0	27.7	29.0	28.1	21.5	21.9	27.5	25.2	25.3	24.8	**31.7**
	20	23.6	23.8	27.1	27.7	25.5	25.3	26.6	29.3	26.2	29.1	**35.1**	34.3
Chameleon	35	31.4	28.6	28.2	31.9	30.2	29.2	30.8	35.1	38.4	28.0	41.4	**43.3**
	100	47.6	47.4	27.9	49.0	46.9	48.9	40.6	49.6	48.3	45.7	52.7	**54.1**
	Average	31.6	31.2	27.7	34.4	32.7	31.2	30.0	35.4	34.5	32.0	38.5	**40.9**
	10	19.8	20.6	20.2	23.2	24.0	19.8	20.9	21.1	20.3	20.4	22.7	**25.7**
	15	20.6	21.3	19.4	**26.3**	24.2	21.3	25.5	21.9	22.2	21.0	23.4	25.8
Squirrel	50	23.9	27.3	25.5	23.9	26.9	23.7	**27.5**	24.1	25.3	21.0	26.0	**27.5**
	100	25.2	30.5	31.6	29.1	31.4	30.6	30.6	27.9	31.2	23.4	29.4	**32.5**
	Average	22.4	24.9	24.2	25.6	26.6	23.9	26.1	23.8	24.8	22.0	25.4	**27.9**

3) CR-GCL-PL: Without Pseudo-Label. This variant omits the pseudo-label-based refinement strategy, which enhances the identification of conflicting pairs. Accuracy slips in all cases like Cora going from 72.4% to 70.6%, or PubMed dropping from 73.3% to 68.3%. The results indicate that the previous identification strategy may capture negative pairs that do not actually lead to the conflict. This observation validates both the rationality and necessity of our conflict refinement strategy.

4) CR-GCL-Feature & CR-GCL-Edge: Only Feature Masking or Edge Dropping. CR-GCL-Feature exclusively applies feature masking, perturbing the node features while leaving the graph topology intact. The variant CR-GCL-Edge only utilizes edge dropping, perturbing the original graph's topological structure while keeping the original node features. Our experimental results show that both of these variants underperform compared to the complete model that employs both perturbation strategies simultaneously. This finding confirms that jointly perturbing both the node features and the graph topology is crucial. The combination of these two augmentation types creates more challenging and comprehensive views for contrastive learning, compelling the model to learn more robust and generalizable representations, thus achieving superior performance.

4.4. Analysis of the fusion mechanism

To figure out the effectiveness of the fusion mechanism, we conducted two sets of experiments. These experiments were designed to isolate the contribution of the fusion mechanism and quantify its impact on both pseudo-label quality and final node classification accuracy.

Table 5. Average ACC (%) of the ablation study (bold denotes the best result).

Datasets	**OURS**	CR-GCL-L2G	CR-GCL-InterVC	CR-GCL-PL	CR-GCL-Feature	CR-GCL-Edge
Cora	**72.3**	71.2	69.7	70.6	71.8	66.2
CiteSeer	**62.4**	61.4	61.1	62.3	58.2	58.1
PubMed	**74.7**	70.1	69.7	68.3	72.6	63.1
ACM	**82.4**	79.5	80.9	80.9	75.5	75.8
Chameleon	**40.9**	38.7	39.0	37.6	40.1	40.5
Squirrel	**27.9**	24.6	25.0	24.8	26.4	26.2

We designed two variants of our model to serve as direct comparisons against our full CR-GCL framework:

1) CR-GCL (Hraw-only): In this variant, we bypass the fusion module entirely. The generation of pseudo-labels and the calculation of similarity for conflict quantification are performed using only the embeddings from the original, un-augmented graph (Hraw).

2) CR-GCL (Haug1-only): Similarly, this variant relies only on the embeddings from the first augmented view (Haug1) for all pseudo-label and conflict quantification tasks.

3) CR-GCL (Full): This is our full CR-GCL model which fuses Hraw, Haug1 and Haug2 with the local-to-global fusion mechanism.

Table 6. Average ACC (%) of pseudo-labels and classification (bold denotes the best result).

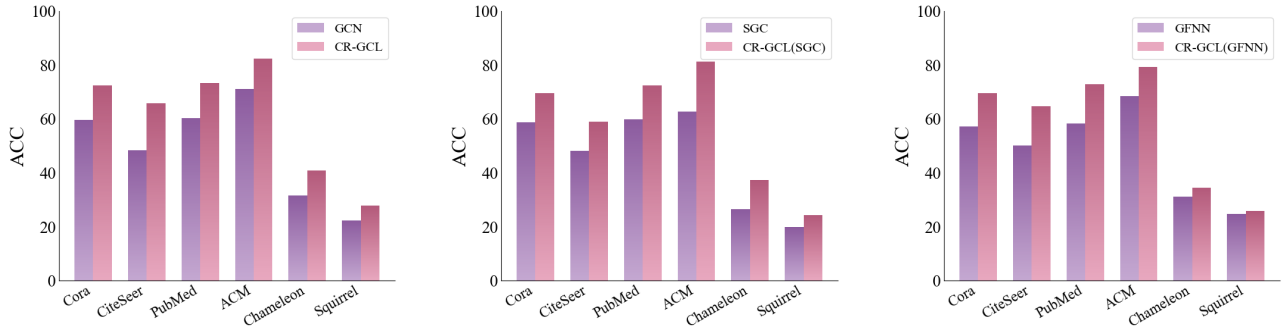
Datasets	Variants	Pseudo-label accuracy	Node classification accuracy
Cora	CR-GCL (Full)	**68.7**	**72.3**
	CR-GCL (Hraw-only)	66.9	69.6
	CR-GCL (Haug1-only)	67.5	69.8
CiteSeer	CR-GCL (Full)	**60.4**	**62.4**
	CR-GCL (Hraw-only)	58.2	58.0
	CR-GCL (Haug1-only)	58.3	59.0
PubMed	CR-GCL (Full)	**73.2**	**74.7**
	CR-GCL (Hraw-only)	71.6	73.9
	CR-GCL (Haug1-only)	69.1	71.7
ACM	CR-GCL (Full)	**84.1**	**82.4**
	CR-GCL (Hraw-only)	80.8	77.7
	CR-GCL (Haug1-only)	83.1	80.8
Chameleon	CR-GCL (Full)	**42.9**	**40.9**
	CR-GCL (Hraw-only)	41.3	38.0
	CR-GCL (Haug1-only)	41.8	38.1
Squirrel	CR-GCL (Full)	**27.1**	**27.9**
	CR-GCL (Hraw-only)	27.0	27.3
	CR-GCL (Haug1-only)	27.1	27.6

From Table 6, we can observe that the full CR-GCL model, which utilizes the fusion strategy, consistently and significantly outperforms both the CR-GCL (Hraw-only) and CR-GCL (Haug1-only) variants across all six benchmark datasets. This holds true for both the final node classification accuracy

and the pseudo-label accuracy. This experiment also verifies the effectiveness of the local-to-global fusion mechanism.

4.5. Framework study

As previously described, CR-GCL can easily incorporate different GCN encoders without imposing restrictions. Two various base encoders SGC and GFNN are integrated in the CR-GCL architecture to result in the CR-GCL(SGC) and CR-GCL(GFNN) models, respectively. We compare their classification ability using an evaluation on various label rates, and extensive results are shown in Figure 5.



(a) GCN encoder **(b)** SGC encoder **(c)** GFNN encoder

Figure 5. Average ACC(%) of base encoders and CR-GCL on six datasets.

The results indicate notable performance gains on all benchmark datasets. Figure 5 shows how CR-GCL-boosted models have significant improvement in accuracy compared with their base encoder in various label rates. Two key advantages are evident from these findings: 1) CR-GCL effectively enhances the generalizability of standard GCN architectures; 2) its design is both scalable and adaptable to different types of encoders while maintaining strong performance. These results underscore the efficacy of CR-GCL as a generalizable framework for graph convolutional networks.

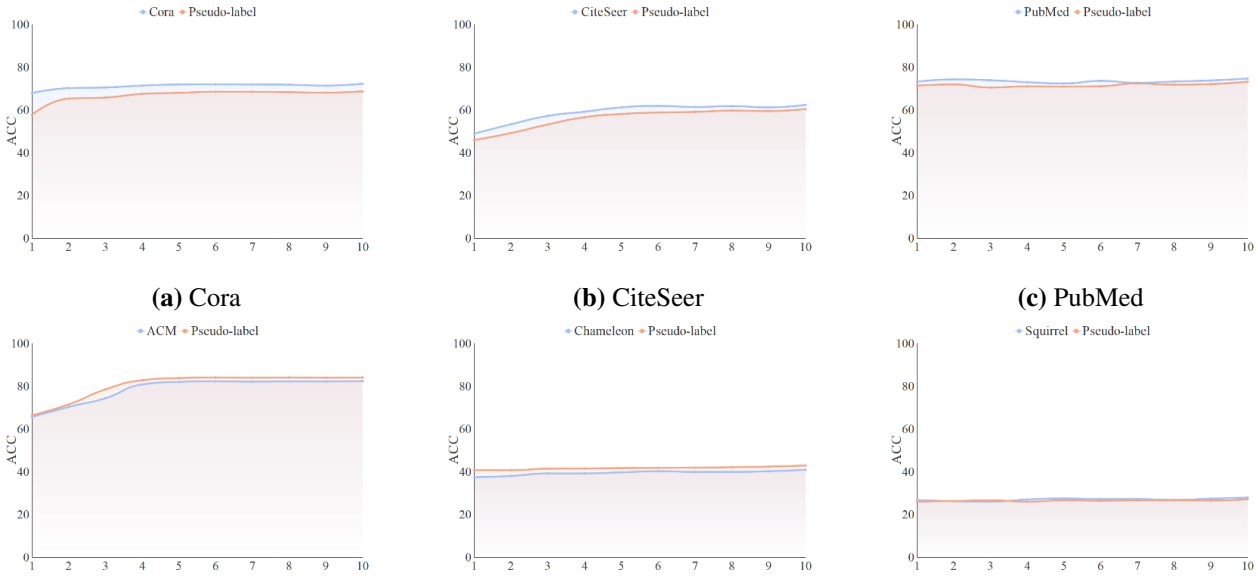
4.6. Pseudo-label accuracy analysis

We conducted a set of experiments to empirically analyze the accuracy of the pseudo-labels throughout the training process and to explore their impact on the final classification performance. We divided the training process into 10 stages, and at the end of each stage, we measured two key metrics:

Pseudo-label accuracy (ACC): The percentage of pseudo-labels assigned to unlabeled nodes that match their ground-truth labels.

Node classification accuracy (ACC): The model's classification accuracy on the test set.

As is shown in Figure 6, at the beginning of the training, the quality of the pseudo-labels is relatively low, and correspondingly, the classification performance is modest. As the training continues, the model learns more discriminative representations, which enables it to generate progressively higher-quality pseudo-labels, which in turn brings higher classification accuracy.



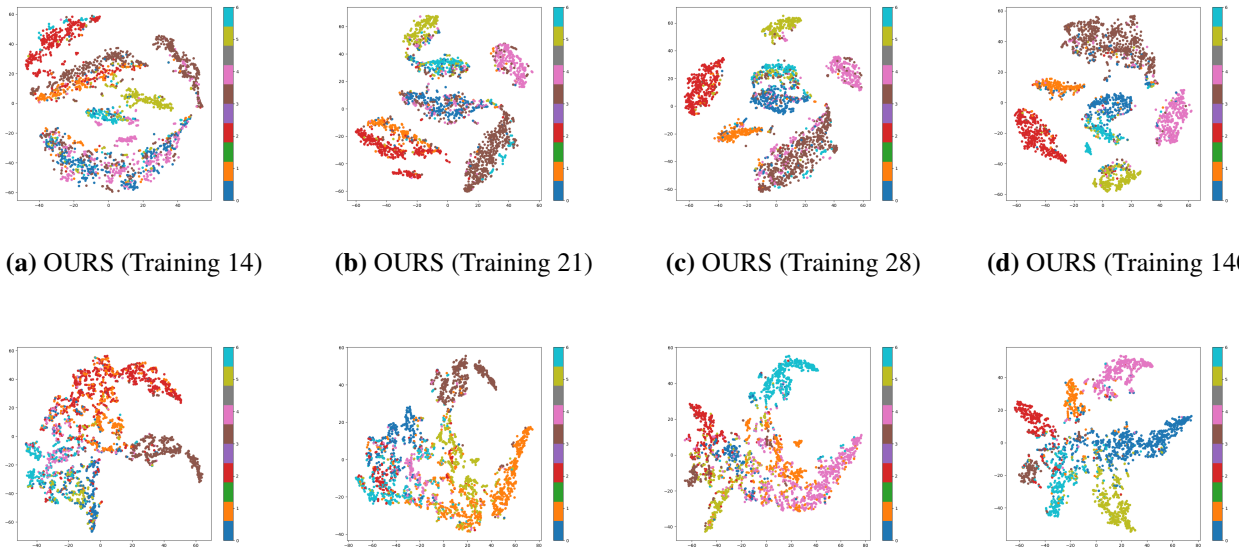
(a) Cora **(b)** CiteSeer **(c)** PubMed

(d) ACM **(e)** Chameleon **(f)** Squirrel

Figure 6. Average ACC (%) of the pseudo-label and classification at different stages.

4.7. Visualization

To further evaluate the effectiveness of CR-GCL, we conducted a node classification visualization study on the Cora dataset under varying quantities of labeled training nodes. Figure 7 presents t-SNE visualizations [22] of learned test set embeddings, with color-coding reflecting ground-truth class labels.



(a) OURS (Training 14) **(b)** OURS (Training 21) **(c)** OURS (Training 28) **(d)** OURS (Training 140)

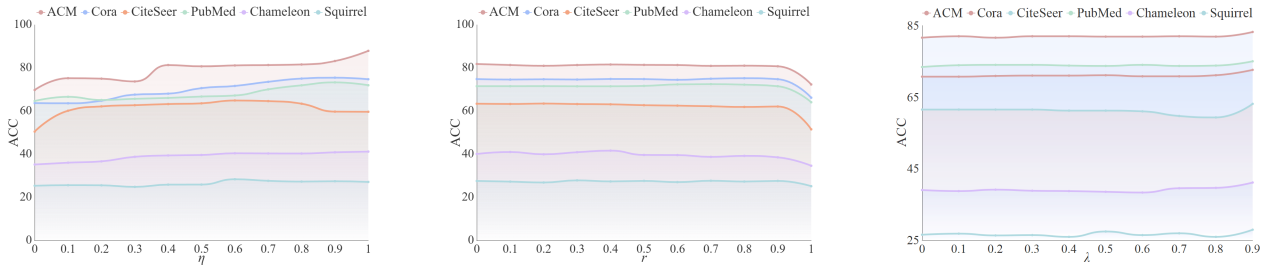
(e) GCN (Training 14) **(f)** GCN (Training 21) **(g)** GCN (Training 28) **(h)** GCN (Training 140)

Figure 7. Visualization of the learned node embeddings on the Cora dataset.

The visualization results reveal distinct performance differences. Graph convolutional networks (GCNs) exhibit inadequate class separation, as evidenced by substantial overlapping of differently colored clusters in the embedding space. In comparison, CR-GCL produces notably superior visual patterns, generating well-defined clusters with three key characteristics: 1) tighter intra-class cohesion, 2) enhanced similarity among same-class instances, and 3) markedly improved inter-class separation boundaries. Furthermore, CR-GCL demonstrates robust performance consistency across different training set sizes, maintaining clear cluster distinctions even when the number of labeled nodes varies significantly.

4.8. Parameter sensitivity

In this part, we conduct sensitivity analysis on three hyper-parameters: balance parameter η, ignore ratio r, and pseudo-label threshold λ over six datasets. We train CR-GCL with η, r ranging from 0.0 to 1.0 in increments of 0.1, respectively. As for λ, we choose to train the model from 0 to 0.9 with an interval of 0.1. Then, we evaluate the average classification accuracy over different label rates, with results shown in Figure 8.



(a) Parameter η **(b)** Parameter r **(c)** Parameter λ

Figure 8. The performance of CR-GCL with parameters η, r, and λ in terms of average ACC (%).

It can be observed that when η grows, the result gets better. Generally, CR-GCL reaches the best result when η ranges from 0.5 to 0.8. As for parameter r, the performance of the model is relatively stable; when $r > 0.9$, the performance of the model decreases significantly. When λ is 0.9, CR-GCL reaches the best classification result.

5. Conclusions

In this paper, we tackle the conflict between the message-passing mechanism of a GCN encoder and the goal of contrastive learning in GCL. Our proposed framework, conflict refined graph contrastive learning (CR-GCL), offers a practical solution by integrating a perturbation augmentation module, a conflict quantification module, and a conflict refinement module, enhanced by pseudo-labels. This approach effectively identifies and refines conflicting negative pairs and thus yields improved node classification performance. Experiments on six benchmark datasets validate that CR-GCL outperforms state-of-the-art techniques and suggests its effectiveness in enhancing GCL frameworks. This research not only deepens the understanding of conflict in GCL but also introduces a principled approach to advancing graph-based learning.

Use of AI tools declaration

The authors declare they have not used Artificial Intelligence (AI) tools in the creation of this article.

Acknowledgments

This research was supported by the National Natural Science Foundation of China (No. 62506145), and the Postgraduate Research and Practice Innovation Program of Jiangsu Province (No. KYCX25_4382).

Conflict of interest

The authors declare there are no conflicts of interest.

References

1. S. Wu, Y. Tang, Y. Zhu, L. Wang, X. Xie, T. Tan, Session-based recommendation with graph neural networks, in *Proceedings of the AAAI Conference on Artificial Intelligence*, **33** (2019), 346–353. https://doi.org/10.1609/aaai.v33i01.3301346
2. Y. Zhang, Y. Xiong, Y. Ye, T. Liu, W. Wang, Y. Zhu, et al., SEAL: Learning heuristics for community detection with generative adversarial networks, in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, (2020), 1103–1113. https://doi.org/10.1145/3394486.3403154
3. S. Vlaic, T. Conrad, C. Tokarski-Schnelle, M. Gustafsson, U. Dahmen, R. Guthke, et al., ModuleDiscoverer: Identification of regulatory modules in protein-protein interaction networks, *Sci. Rep.*, **8** (2018), 433. https://doi.org/10.1038/s41598-017-18370-2
4. J. Yu, H. Yin, X. Xia, T. Chen, L. Cui, Q. V. H. Nguyen, Are graph augmentations necessary? simple graph contrastive learning for recommendation, in *Proceedings of the 45th international ACM SIGIR conference on research and development in information retrieval*, (2022), 1294–1303. https://doi.org/10.1145/3477495.3531937
5. Y. Zhang, H. Zhu, Z. Song, P. Koniusz, I. King, Spectral feature augmentation for graph contrastive learning and beyond, in *Proceedings of the AAAI Conference on Artificial Intelligence*, **37** (2023), 11289–11297. https://doi.org/10.1080/14432471.2023.2265618
6. Z. Zhang, S. Sun, G. Ma, C. Zhong, Line graph contrastive learning for link prediction, *Pattern Recognit.*, **140** (2023), 109537. https://doi.org/10.1016/j.patcog.2023.109537
7. Z. Tong, Y. Liang, H. Ding, Y. Dai, X. Li, C. Wang, Directed graph contrastive learning, *Adv. Neural Inf. Process. Syst.*, **34** (2021), 19580–19593.
8. Y. You, T. Chen, Y. Sui, T. Chen, Z. Wang, Y. Shen, Graph contrastive learning with augmentations, *Adv. Neural Inf. Process. Syst.*, **33** (2020), 5812–5823.
9. J. Wang, J. Ren, Graphormer based contrastive learning for recommendation, *Appl. Soft Comput.*, **159** (2024), 111626. https://doi.org/10.1016/j.asoc.2024.111626

10. D. He, J. Zhao, C. Huo, Y. Huang, Y. Huang, Z. Feng, A new mechanism for eliminating implicit conflict in graph contrastive learning, in *Proceedings of the AAAI Conference on Artificial Intelligence*, **38** (2024), 12340–12348. https://doi.org/10.1609/aaai.v38i11.29125
11. C. Ji, Z. Huang, Q. Sun, H. Peng, X. Fu, Q. Li, et al., ReGCL: Rethinking message passing in graph contrastive learning, in *Proceedings of the AAAI Conference on Artificial Intelligence*, **38** (2024), 8544–8552. https://doi.org/10.1609/aaai.v38i8.28698
12. Q. Guo, X. Yang, F. Zhang, T. Xu, Perturbation-augmented graph convolutional networks: A graph contrastive learning architecture for effective node classification tasks, *Eng. Appl. Artif. Intell.*, **129** (2024), 107616. https://doi.org/10.1016/j.engappai.2023.107616
13. Q. Sun, X. Wei, X. Yang, Graphsage with deep reinforcement learning for financial portfolio optimization, *Expert Syst. Appl.*, **238** (2024), 122027. https://doi.org/10.1016/j.eswa.2023.122027
14. S. Abu-El-Haija, A. Kapoor, B. Perozzi, J. Lee, N-GCN: Multi-scale graph convolution for semi-supervised node classification, in *Proceedings of The 35th Uncertainty in Artificial Intelligence Conference*, PMLR, **115** (2020), 841–851.
15. S. Abu-El-Haija, B. Perozzi, A. Kapoor, N. Alipourfard, K. Lerman, H. Harutyunyan, et al., Mixhop: Higher-order graph convolutional architectures via sparsified neighborhood mixing, in *International Conference on Machine Learning*, **97** (2019), 21–29.
16. K. Yao, J. Liang, J. Liang, M. Li, F. Cao, Multi-view graph convolutional networks with attention mechanism, *Artif. Intell.*, **307** (2022), 103708. https://doi.org/10.1016/j.artint.2022.103708
17. L. He, L. Bai, X. Yang, H. Du, J. Liang, High-order graph attention network, *Inf. Sci.*, **630** (2023), 222–234. https://doi.org/10.1016/j.ins.2023.02.054
18. Y. Zhu, Y. Xu, F. Yu, Q. Liu, S. Wu, L. Wang, Graph contrastive learning with adaptive augmentation, in *Proceedings of the Web Conference*, (2021), 2069–2080. https://doi.org/10.1145/3442381.3449802
19. J. Chen, T. Ma, C. Xiao, FastGCN: Fast learning with graph convolutional networks via importance sampling, in *International Conference on Learning Representations*, 2018.
20. Q. Ni, W. Peng, Y. Zhu, R. Ye, Graph dropout self-learning hierarchical graph convolution network for traffic prediction, *Eng. Appl. Artif. Intell.*, **123** (2023), 106460. https://doi.org/10.1016/j.engappai.2023.106460
21. H. Cong, Q. Sun, X. Yang, K. Liu, Y. Qian, Enhancing graph convolutional networks with progressive granular ball sampling fusion: A novel approach to efficient and accurate GCN training, *Inf. Sci.*, **676** (2024), 120831. https://doi.org/10.1016/j.ins.2024.120831
22. X. Wang, M. Zhu, D. Bo, P. Cui, C. Shi, J. Pei, AM-GCN: Adaptive multi-channel graph convolutional networks, in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, (2020), 1243–1253. https://doi.org/10.1145/3394486.3403177
23. J. Wang, J. Liang, J. Cui, J. Liang, Semi-supervised learning with mixed-order graph convolutional networks, *Inf. Sci.*, **573** (2021), 171–181. https://doi.org/10.1016/j.ins.2021.05.057
24. Q. Teng, X. Yang, Q. Sun, P. Wang, X. Wang, T. Xu, Sequential attention layer-wise fusion network for multi-view classification, *Int. J. Mach. Learn. Cybern.*, **15** (2024), 5549–5561. https://doi.org/10.1007/s13042-024-02260-x

25. Q. Guo, X. Yang, W. Ding, Y. Qian, Cross-graph interaction networks, *IEEE Trans. Knowl. Data Eng.*, **37** (2025), 2341–2355. https://doi.org/10.1109/TKDE.2025.3543377

26. P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, Y. Bengio, Graph attention networks, in *International Conference on Learning Representations*, 2018.

27. Q. Guo, X. Yang, M. Li, Y. Qian, Collaborative graph neural networks for augmented graphs: A local-to-global perspective, *Pattern Recognit.*, **158** (2025), 111020. https://doi.org/10.1016/j.patcog.2024.111020

28. J. Qiu, Q. Chen, Y. Dong, J. Zhang, H. Yang, M. Ding, et al., GCC: Graph contrastive coding for graph neural network pre-training, in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, (2020), 1150–1160. https://doi.org/10.1145/3394486.3403168

29. W. Ju, Y. Wang, Y. Qin, Z. Mao, Z. Xiao, J. Luo, et al., Towards graph contrastive learning: A survey and beyond, preprint, arXiv:2405.11868. https://doi.org/10.48550/arXiv.2405.11868

30. P. Veličković, W. Fedus, W. L. Hamilton, P. Liò, Y. Bengio, R. D. Hjelm, Deep graph infomax, preprint, arXiv:1809.10341. https://doi.org/10.48550/arXiv.1809.10341

31. Y. Zhu, Y. Xu, F. Yu, Q. Liu, S. Wu, L. Wang, Deep graph contrastive representation learning, preprint, arXiv:2006.04131. https://doi.org/10.48550/arXiv.2006.04131

32. W. Ju, Y. Gu, Z. Mao, Z. Qiao, Y. Qin, X. Luo, et al., GPS: Graph contrastive learning via multi-scale augmented views from adversarial pooling, *Sci. China Inf. Sci.*, **68** (2025), 112101. https://doi.org/10.1007/s11432-022-3952-3

33. J. Xia, L. Wu, G. Wang, J. Chen, S. Z. Li, Progcl: Rethinking hard negative mining in graph contrastive learning, in *International Conference on Machine Learning*, (2022), 24332–24346. https://doi.org/10.48550/arXiv.2110.02027

34. N. Liu, X. Wang, D. Bo, C. Shi, J. Pei, Revisiting graph contrastive learning from the perspective of graph spectrum, *Adv. Neural Inf. Process. Syst.*, **35** (2022), 2972–2983.

35. X. Luo, W. Ju, M. Qu, C. Chen, M. Deng, X. S. Hua, et al., Dualgraph: Improving semi-supervised graph classification via dual contrastive learning, in *2022 IEEE 38th International Conference on Data Engineering (ICDE)*, IEEE, (2022), 699–712. https://doi.org/10.1109/ICDE53745.2022.00057

36. Y. Gu, Z. Chen, Y. Qin, Z. Mao, Z. Xiao, W. Ju, et al., DEER: Distribution divergence-based graph contrast for partial label learning on graphs, *IEEE Trans. Multimedia*, **2024** (2024). https://doi.org/10.1109/TMM.2024.3408038

37. W. Ju, Y. Qin, S. Yi, Z. Mao, K. Zheng, L. Liu, et al., Zero-shot node classification with graph contrastive embedding network, *Trans. Mach. Learn. Res.*, 2023.

38. C. Huo, D. Jin, Y. Li, D. He, Y. B. Yang, L. Wu, T2-GNN: Graph neural networks for graphs with incomplete features and structure via teacher-student distillation, in *Proceedings of the AAAI Conference on Artificial Intelligence*, **37** (2023), 4339–4346. https://doi.org/10.1080/14432471.2023.2265618

39. Y. Yang, Y. Sun, F. Ju, S. Wang, J. Gao, B. Yin, Multi-graph fusion graph convolutional networks with pseudo-label supervision, *Neural Networks*, **158** (2023), 305–317. https://doi.org/10.1016/j.neunet.2022.11.027

40. T. N. Kipf, M. Welling, Semi-supervised classification with graph convolutional networks, in *International Conference on Learning Representations*, 2017.

41. X. Wang, H. Ji, C. Shi, B. Wang, Y. Ye, P. Cui, et al., Heterogeneous graph attention network, in *The World Wide Web Conference*, (2019), 2022–2032. https://doi.org/10.1145/3308558.3313562

42. H. Pei, B. Wei, K. C. C. Chang, Y. Lei, B. Yang, Geom-GCN: Geometric graph convolutional networks, in *International Conference on Learning Representations*, 2020. https://doi.org/10.48550/arXiv.2002.05287

43. N. Hoang, T. Maehara, T. Murata, Revisiting graph neural networks: Graph filtering perspective, in *International Conference on Pattern Recognition*, (2021), 8376–8383. https://doi.org/10.1109/ICPR48806.2021.9412278

44. T. Jin, H. Dai, L. Cao, B. Zhang, F. Huang, Y. Gao, et al., Deepwalk-aware graph convolutional networks, *Sci. China Inf. Sci.*, **65** (2022), 152104. https://doi.org/10.1007/s11432-020-3318-5

45. J. Ben, Q. Sun, K. Liu, X. Yang, F. Zhang, Multi-head multi-order graph attention networks, *Appl. Intell.*, **54** (2024), 8092–8107. https://doi.org/10.1007/s10489-024-05601-z

46. S. Liu, R. Ying, H. Dong, L. Li, T. Xu, Y. Rong, et al., Local augmentation for graph neural networks, in *Proceedings of the 39th International Conference on Machine Learning*, **162** (2022), 14054–14072.

47. X. Zhang, G. Ding, J. Li, W. Wang, Q. Wu, Deep learning empowered mac protocol identification with squeeze-and-excitation networks, *IEEE Trans. Cognit. Commun. Networking*, **8** (2021), 683–693. https://doi.org/10.1109/TCCN.2021.3126306

Appendix

A. Theoretical analysis of inter-view conflict

To provide a rigorous theoretical foundation for the concept of "inter-view conflict," we present a formal proof that demonstrates how the GCN message-passing mechanism inherently interferes with the optimization objectives of the InfoNCE loss.

A.1. Step 1: Formalizing the conflict scenario

As illustrated in Figure 1(b) of our paper, the inter-view conflict arises from a specific set of relationships. We formalize this scenario with the following elements:

Anchor node (u_i): The representation of node v_i in the first augmented view $\mathcal{G}_1$.

Positive sample (u_i'): The representation of the same node v_i in the second augmented view $\mathcal{G}_2$.

Conflicting neighbor (v_k): A node that is a neighbor of v_i in the first view $\mathcal{G}_1$. Its representation in this view is u_k.

The InfoNCE loss function imposes two simultaneous and, as we will show, conflicting optimization objectives on the representation of the anchor node u_i:

1). Positive pair alignment: Maximize the similarity with its corresponding positive sample, $sim(u_i, u_i')$.

2). Negative pair separation: Minimize the similarity with all negative samples, including its neighbor, $sim(u_i, u_k)$.

A.2. Step 2: The role of GCN message-passing

The conflict is rooted in the GCN's message-passing mechanism. For simplicity, let us consider a single-layer GCN encoder. The representation of the anchor node, H_i, is an aggregation of its own features and its neighbors' features. We can explicitly write out the contribution of the conflicting neighbor v_k:

$$H_i = \sigma\left(\frac{1}{\sqrt{\tilde{d}_i\tilde{d}_k}}X_kW + \sum_{j\in(\mathcal{N}(i)\cup\{i\})\setminus\{k\}}\frac{1}{\sqrt{\tilde{d}_i\tilde{d}_j}}\tilde{A}_{ij}X_jW\right) \tag{A.1}$$

Here, X_k is the input feature vector of node v_k and W is the GCN's weight matrix. Let $h_k = X_kW$ be the transformed feature of the neighbor. The final representation u_i is obtained after passing H_i through a projector $u_i = f(H_i)$. Crucially, this means u_i is a function of h_k. By the chain rule, we establish the fundamental link:

$$\frac{\partial u_i}{\partial h_k} \neq 0 \tag{A.2}$$

This non-zero partial derivative confirms that any change to the neighbor's features h_k will directly affect the anchor's representation u_i.

A.3. Step 3: Decomposing the gradient forces on the anchor

The total InfoNCE loss $\mathcal{L}$ exerts a gradient force on the anchor representation u_i. This force, $\nabla_{u_i}\mathcal{L}$, can be decomposed into two distinct components:

1) Attract force (F_{pos}): This component arises from the positive pair term in the loss function. It pushes u_i in the direction that maximizes its similarity with the positive sample u_i', effectively pulling u_i toward u_i'.

2) Push force (F_{neg}): This component arises from the negative pair term. It pushes u_i in the direction that minimizes its similarity with the negative sample u_k, effectively pushing u_i away from u_k.

The total gradient applied to the anchor is the vector sum of these forces: $\nabla_{u_i}\mathcal{L} = F_{pos} + F_{neg}$. These two forces generally point in different directions in the embedding space.

A.4. Step 4: Propagating conflicting forces to the neighbor via the chain rule

Using the chain rule, we can calculate the gradient of the total loss with respect to the neighbor's feature h_k:

$$\nabla_{h_k}\mathcal{L} = \frac{\partial\mathcal{L}}{\partial u_i}\frac{\partial u_i}{\partial h_k} = (\nabla_{u_i}\mathcal{L})^T\frac{\partial u_i}{\partial h_k} \tag{A.3}$$

By substituting the decomposed force from Step 3, we reveal the conflict:

$$\nabla_{h_k}\mathcal{L} = (F_{pos} + F_{neg})^T \frac{\partial u_i}{\partial h_k} = \underbrace{(F_{pos})^T \frac{\partial u_i}{\partial h_k}}_{\text{Term A: Cooperative Update}} + \underbrace{(F_{neg})^T \frac{\partial u_i}{\partial h_k}}_{\text{Term B: Antagonistic Update}} \tag{A.4}$$

A.5. Step 5: Conclusion of the analysis

The equation above provides the formal proof of the inter-view conflict. It shows that the feature vector h_k of the conflicting neighbor receives two simultaneous and contradictory update signals during backpropagation:

1) Term A (Cooperative update): This term represents the gradient update required of h_k to help move the anchor u_i closer to the positive sample u_i'. It is a "cooperative" signal, as the neighbor's features are adjusted to aid in positive pair alignment.

2) Term B (Antagonistic update): This term represents the gradient update required of h_k to help move the anchor u_i away from the negative sample u_k. It is an "antagonistic" signal, as the neighbor's features are adjusted to aid in negative pair separation.

Since the attractive force F_{pos} and repulsive force F_{neg} are not aligned, Term A and Term B pull the feature vector h_k in different directions. The final gradient applied to h_k is a compromised vector sum of these two conflicting objectives. This directly demonstrates that the process of separating the anchor from its neighbor (a negative sample) mathematically interferes with the primary goal of aligning the anchor with its positive sample. This interference is the essence of the inter-view conflict.

B. Time complexity

Let N be the number of nodes, E the number of edges, F the dimension of the input node features, and d the embedding dimension of the GCN encoder output.

The overall time complexity of the CR-GCL framework per training epoch is $O(E \cdot F + N^2 \cdot d + N^2 \cdot \log N)$. This complexity is primarily composed of the computational costs of the following three core modules:

1) Perturbation augmentation module: The computational cost of this module is dominated by the two-layer GCN encoder, which runs on the original graph and two augmented views. Its complexity is $O(E \cdot F + N \cdot d^2)$.

2) Conflict quantification module: This module tracks representation changes by calculating the cosine similarity between all pairs of nodes to estimate the gradient impact (GI). This step requires constructing an $N \times N$ similarity matrix, leading to a dominant complexity of $O(N^2 \cdot d)$.

3) Conflict refinement module: The computational bottleneck of this module lies in the all-pairs similarity calculation for pseudo-label generation ($O(N^2 \cdot d)$) and the sorting of N^2 GI values to determine the conflict threshold ($O(N^2 \cdot \log N)$).