A New Mechanism for Eliminating Implicit Conflict in Graph Contrastive Learning

Dongxiao He[1], **Jitao Zhao**[1], **Cuiying Huo**[1*], **Yongqi Huang**[1], **Yuxiao Huang**[2], **Zhiyong Feng**[1*]
[1]College of Intelligence and Computing, Tianjin University, Tianjin, China.
[2]Department of Data Science, George Washington University, NW Washington DC, America.
{hedongxiao, zjtao, huocuiying, yqhuang, zyfeng}@tju.edu.cn, yuxiaohuang@email.gwu.edu

Abstract

Graph contrastive learning (GCL) has attracted considerable attention because it can self-supervisedly extract low-dimensional representation of graph data. InfoNCE-based loss function is widely used in graph contrastive learning, which pulls the representations of positive pairs close to each other and pulls the representations of negative pairs away from each other. Recent works mainly focus on designing new augmentation methods or sampling strategies. However, we argue that the widely used InfoNCE-based methods may contain an implicit conflict which seriously confuses models when learning from negative pairs. This conflict is engendered by the encoder's message-passing mechanism and the InfoNCE loss function. As a result, the learned representations between negative samples cannot be far away from each other, compromising the model performance. To our best knowledge, this is the first time to report and analysis this conflict of GCL. To address this problem, we propose a simple but effective method called **P**artial **i**gnored **G**raph **C**ontrastive **L**earning (PiGCL). Specifically, PiGCL first dynamically captures the conflicts during training by detecting the gradient of representation similarities. It then enables the loss function to ignore the conflict, allowing the encoder to adaptively learn the ignored information without self-supervised samples. Extensive experiments demonstrate the effectiveness of our method.

Introduction

In recent years, Graph Neural Networks (GNNs) have gained significant attention due to their powerful capabilities in graph data analysis (Wu et al. 2020) and have been applied in a wide range of scenarios, such as electronic recommendation (Wu et al. 2019), community detection (Zhang et al. 2020), and protein molecule prediction (Vlaic et al. 2018). However, training high-quality GNN encoders requires a large amount of manually annotated labels, thereby requiring substantial human and material resources. To solve the dependency on labels, numerous studies have concentrated on Self-Supervised Learning (SSL). SSL finds labels within the data itself, and designs proxy tasks for training without manual label, achieving significant success in the fields of CV and NLP (Jaiswal et al. 2020). At present, there are three mainstream paradigms in Graph Self-Supervised Learning (GSSL). BGRL-like frameworks (Thakoor et al. 2021) design two encoders with non-shared parameters and train one encoder to learn the output of the other encoder. DGI-like frameworks (Veličković et al. 2019) generate a new augmented view, and train the model by making it distinguish whether a node belongs to the original view or the augmented view. However, the training objectives of these two methods do not consider the intrinsic inconsistent relationships among the nodes of graph, which prevent downstream tasks from making effective distinctions.

The third paradigm, InfoNCE-based frameworks (Zhu et al. 2020; You et al. 2020), leverage node inconsistent relationships in datasets by negative sampling, enhancing the distinguishability of the learned representations. Specifically, They generate augmented views through random perturbation. In the augmented views, two nodes generated from the same source node are viewed as positive pairs, while others are perceived as negative pairs. Then, the objective functions make encoder embed consistences between positive pairs and differences between negative pairs (Liu et al. 2022; Shi et al. 2023) by maximizing the similarity between positive pairs and minimizing the similarity between negative pairs. Recently, a substantial volume of research focus on refining the data augmentation techniques and sampling strategies of InfoNCE-based paradigm to improve model performance (Suresh et al. 2021; Zhu et al. 2021; Xia et al. 2022, 2021; Zhang et al. 2022). **However, these methods have failed to recognize a potential conflict inherent in InfoNCE-based methods.**

This kind of conflicts is ubiquitous in InfoNCE-based methods, which is jointly induced by the InfoNCE objective function and the unique topological message-passing mechanism (Zhou et al. 2020) of GNN encoders. Here, we give a intuitive illustration, where the encoder only has a single layer of Graph Convolutional Network (GCN) (Kipf and Welling 2017), meaning its topological receptive field includes the node's one-hop neighbors. As shown in Fig. 1 (a), during the forward propagation of encoder, the anchor node v_1 aggregates information from nodes v_1, v_2, and v_3. Node v_2 aggregates information from v_1 and v_2, while node v_3 aggregates information from v_2 and v_3. During loss calculation, nodes v_2 and v_3 both form negative pairs with v_1. Considering the negative pair formed by v_1 and v_2, as shown in Fig. 1 (b), the objective function requires the encoder to

[*]Corresponding authors

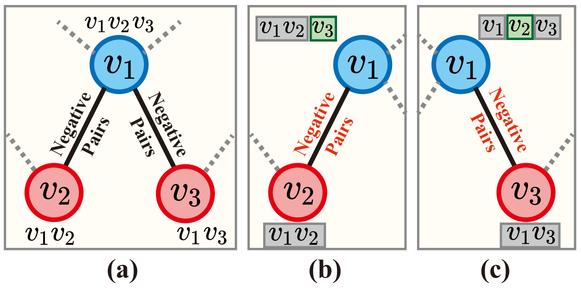


Figure 1: Conflict diagram. (a) shows origin node relationship. (b) shows node relationship considering negative pair of v_1 and v_2. (c) shows node relationship considering negative pair of v_1 and v_3.

embed the differences between the two nodes, i.e., the information from node v_3, while disregarding the similar parts, i.e., the information from node v_1 and v_2, to minimize the similarity between the negative pair. However, when considering the negative pair formed by v_1 and v_3, as shown in Fig. 1 (c), the situation is entirely reversed: the information from node v_2 is required to be embedded, while the information from node v_3 is requested to be ignored. At this point, we find that when the encoder generates the representation of v_1, the InfoNCE loss demands it to simultaneously embed and ignore the aggregated information from nodes v_2 and v_3. In the paper, we will refer to this conflict as the Embedding and Ignoring Conflict (EIC). In addition, EIC issue also occurs in other GNNs encoders. A detailed theoretical proof for the wide existence of EIC is provided in the later section.

EIC issue has a detrimental impact on node representations because it greatly confuses gradient during the training process. The confusion comes from two sources. (1) For negative pairs involved in EIC, the gradients receipted by the encoder for these nodes are confused. Therefore, the encoder cannot reduce the similarity between the representations of the negative pairs, which directly leads to learned representations over-similar. (2) For negative pairs not involved in EIC, we have discovered through theoretical derivation that the existence of a large number of highly similar EIC negative nodes unfairly reduces the gradient of negative pairs not involved in EIC. As a result, the similarity of non-EIC negative pairs cannot decrease at the normal rate. These two kinds of impacts directly result in the encoder generating overly similar representations, which contradicts the design philosophy of the InfoNCE loss function, making it more difficult for the downstream task predictors to distinguish among such similar representations. Unfortunately, methods with improved sampling strategy (Zhang et al. 2022; He et al. 2023; Wang et al. 2022) cannot solve the EIC issue, as its occurrence is unrelated to sampling accuracy.

To solve above problem, we propose a simple yet effective method called **P**artially **i**gnored **G**raph **C**ontrastive **L**earning (PiGCL), which includes an EIC capturer and controller. The capturer identifies EIC negative pairs by detecting confused gradient information. The controller then dynamically ignores some EIC pairs, allowing the GNN encoder to adaptively learn from ignored negative samples. In addition, the ignored nodes are updated in each iteration, which means all negative samples are used in training. This approach is completely different from existing improvements on positive and negative sampling which focus on how to select samples, whereas we focus on how to better utilize sampling information in training. A wealth of experimental results validate the effectiveness of PiGCL.

We summarize our contributions as follows:

- **Problem**: We introduce the concept of an implicit conflict, termed as EIC, within GCL. To the best of our knowledge, this is the first work to identify and articulate this issue. We provide a theoretical exploration of EIC and prove its detrimental effects on learning.
- **Method**: We propose PiGCL to address the EIC issue. It uses gradient confusion to capture and dynamically ignore EIC negative pairs to eliminate EICs. And PiGCL tones up the gradient on the loss function to mitigate the impact on the gradient caused by the ignoring. In this design, PiGCL greatly improves the encoder's ability to learn from negative examples.
- **Validation**: We conducte experiments on six datasets and two downstream tasks. An abundance of experimental evidence confirms the efficacy of PiGCL

Preliminary

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ represent a graph, where $\mathcal{V} = \{v_1, v_2, ..., v_n\}$ denotes the set of nodes and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ signifies the set of edges. $X \in \mathbb{R}^{N \times F}$ is the feature matrix of the nodes. $A \in \mathbb{R}^{N \times N}$ is the adjacency matrix, where $A_{ij} = 1$ if $(v_i, v_j) \in \mathcal{E}$ and $A_{ij} = 0$ otherwise.

In graph contrastive learning, given the input of a graph's X and A, our goal is to train an encoder $f(\cdot)$, without the need for manual labels. The encoder can generate low-dimensional, dense representations $Z = f(X, A) \in R^{N \times D}$. These representations can then be utilized for a variety of downstream tasks.

GNN encoder. Almost all recent Graph Self-Supervised Learning (GSSL) works employ two-layer GNN encoders. The GNNs generally consist of propagation and aggregation mechanism (Huo et al. 2023; Kipf and Welling 2017; Veličković et al. 2018). The normal form of GNNs is as follows:

$$X^{(l)} = \sigma(\hat{A}^{(l)} W^{(l)} X^{(l-1)}), \tag{1}$$

where $\hat{A}^{(l)}$ is the propagation matrix, extracted from graph through neighbor sampling, transition matrix computation or deep learning (**?**). l is the layer number, $X^{(0)}$ means the original feature. W denotes a learnable parameter matrix, σ symbolizes an activate function. In this design, GNN can efficiently encode the feature and structure information inherit in the graph.

Discovering and Analysing of the EIC in GCL

In this section, we provide a detailed theoretical proof of the wide existence of EIC in InfoNCE-based methods and discuss its impacts on model performance.

The Proof of EIC

In this section, we aim to demonstrate that *when the topological receptive fields of more than two negative samples intersect with that of the anchor node, there is a high likelihood for the nodes to fall into the EIC* in four steps. To simplify the proof, we assume that the encoder leverages the most widely used GCN encoder in GSSL, and that the encoder contains only a single layer of GCN.

Firstly, we select any three nodes v_a, v_b and v_c that satisfy the following condition. (1) Nodes v_b and v_c are negative samples of v_a. (2) These nodes satisfy:

$$TR(v_a) \cap TR(v_b) \neq \emptyset, TR(v_a) \cap TR(v_c) \neq \emptyset, \tag{2}$$

where $TR(v_a)$ denotes the set consisting of nodes within the topological receptive field of v_a.

Step 1. The EIC is jointly induced by the InfoNCE loss function and the GNN encoder. First, in order to find the restriction of the loss function on the representation, we analyze the following loss function:

$$\begin{aligned}&\mathcal{L}_{\text{InfoNCE}}(z_i) = -\log(\\ &\frac{e^{\theta(z_i,z_i')/\tau}}{e^{\theta(z_i,z_i')/\tau} + \sum_{k=1,k\neq i}^{n}(e^{\theta(z_i,z_k)/\tau} + e^{\theta(z_i,z_k')/\tau})}),\end{aligned} \tag{3}$$

where z_i represents the learned representation of node v_i, z_i' is the representation learned from node v_i' on the other augmented view, corresponding to v_i. τ is a hyper-parameter. $\theta(z_i, z_i') = Sim(g(z_i), g(z_i'))$, where $Sim(\cdot)$ means cosine similarity calculation, $g(\cdot)$ is a non-linear projector. From Eq. 3, to minimize the loss function, encoder need to maximize the cosine similarity between positive pairs and minimize that between negative ones.

Step 2. We analyze what kind of representations the GNN encoder has learned from the perspective of spatial domain, which is another contributing factor to the EIC. The representation matrix Z is generated as Eq. 4:

$$Z = \sigma(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}XW), \tag{4}$$

where, σ is the activation function, $\tilde{A} = A + I$ is the self-looped adjacency matrix, and $\tilde{D}$ is the degree matrix of $\tilde{A}$ used for symmetric Laplacian regularization. W is a learnable parameter matrix. Substituting $H = XW$ into Eq. 4, the representation of v_a can be derived as follows:

$$z_a = \sigma(\sum_{i\in TR(v_a)} \frac{1}{\sqrt{\tilde{d}_a}\sqrt{\tilde{d}_i}} h_i) = \sigma(\frac{1}{\sqrt{\tilde{d}_a}}(\sum_{i\in TR(v_a)} \frac{1}{\sqrt{\tilde{d}_i}} h_i)), \tag{5}$$

here, $\tilde{d}_i$ represents the degree of node v_i after the addition of a self-loop. Derived from Eq. 5, the encoder can be conceptualized as first embedding the node features, then performing a regularized aggregation of these embeddings within the topological receptive field.

Step 3. In conjunction with the previous two steps, we analyze how the loss function will influence the encoder. We select a pair of negative nodes, v_a and v_b to find out what kind of representation is the encoder required to generate considering only one negative pair. According to Eq. 5, the representations z_a and z_b can be decomposed as follows:

$$\begin{aligned}z_a &= \sigma(\frac{1}{\sqrt{\tilde{d}_a}}(\sum_{i\in Co(v_a,v_b)} \frac{1}{\sqrt{\tilde{d}_i}} h_i + \sum_{i\in Di_a(v_a,v_b)} \frac{1}{\sqrt{\tilde{d}_i}} h_i)), \\ z_b &= \sigma(\frac{1}{\sqrt{\tilde{d}_b}}(\sum_{i\in Co(v_a,v_b)} \frac{1}{\sqrt{\tilde{d}_i}} h_i + \sum_{i\in Di_b(v_a,v_b)} \frac{1}{\sqrt{\tilde{d}_i}} h_i)),\end{aligned} \tag{6}$$

here, $Co(v_a, v_b)$ denotes the set formed by the nodes shared within the receptive fields of nodes v_a and v_b. Meanwhile, $Di_a(v_a, v_b)$ refers to the set comprising nodes that are within the receptive field of node v_a but not within that of node v_b, $Co(v_a, v_b) \cup Di_a(v_a, v_b) = TR(v_a)$, $Co(v_a, v_b) \cap Di_a(v_a, v_b) = \emptyset$. At this point, it can be observed that in the above equation, the representation can be regarded as the sum of two vectors as follows:

$$\begin{aligned}\overrightarrow{z_a} &= \sigma(\mu_a(\overrightarrow{Vec_{Co(v_a,v_b)}} + \overrightarrow{Vec_{Di_a(v_a,v_b)}})), \\ \overrightarrow{z_b} &= \sigma(\mu_b(\overrightarrow{Vec_{Co(v_a,v_b)}} + \overrightarrow{Vec_{Di_b(v_a,v_b)}})),\end{aligned} \tag{7}$$

where $\mu_a = 1/\sqrt{\tilde{d}_i}$. Cosine similarity measures the angle between vectors, and the direction of the first sub-vector in z_a and z_b is precisely the same. Therefore, $\overrightarrow{Vec_{Co(v_a,v_b)}}$ cannot decisively contribute to the reduction of similarity during the training process; instead, when its magnitude becomes excessively large, it reduces the angle between z_a and z_b (particularly when the activation function is ReLU-liked, which is most commonly used in GSSL (Glorot, Bordes, and Bengio 2011)). Consequently, under the InfoNCE-based loss, the model tends to encode nodes that are in the $Di_a(v_a, v_b)$ set and ignore those in the $Co(v_a, v_b)$ set.

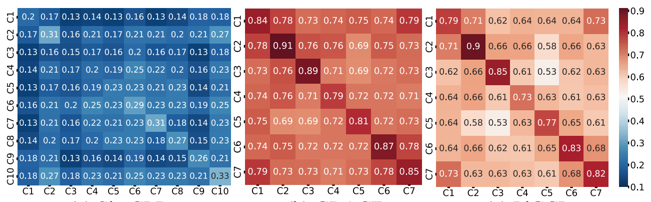
Step 4. On the basis of step 3, we further consider more negative pairs to analyze whether the encoder can meet the embedding requirements of multiple negative pairs at the same time to discover the EIC. When considering the negative pair consisted of v_a and v_c, as in Eq. 6, $Co(v_a, v_c)$ and $Di_a(v_a, v_c)$ will also be produced for v_a. In the case of having only one-hop neighbors, the pair of negative samples certainly satisfies:

$$Co(v_a, v_b) \neq Co(v_a, v_c) \Leftrightarrow Di_a(v_a, v_b) \neq Di_a(v_a, v_c). \tag{8}$$

Moreover, as the receptive field expands, there is a high probability that the pair of negative samples also satisfy Eq. 8. In this scenario, we observe that the nodes inconsistent between sets in Eq. 8 are required to be encoded, yet simultaneously ignored.

Therefore, when there are more than two negative samples' topology receptive fields have intersection with that of anchor node, there is a high probability that nodes will fall into the EIC.

The generalization of the proof. When the encoder is extended to a two-layer GCN, due to the expansion of the encoder's receptive field, more nodes will fall into the EIC. The above derivation generally holds when GNNs that cannot learn edge weights, such as GraphSAGE (Hamilton, Ying, and Leskovec 2017), GIN (Xu et al. 2018). When GNNs that can learn edge weights, like GAT (Veličković et al. 2018),



(a) SimCLR (b) GRACE (c) PiGCL

Figure 2: Similarity Heatmap. Figure (a) presents the result of training with SimCLR (Chen et al. 2020) on the Cifar10 (?) dataset, Figure (b) exhibits the outcome of training with GRACE on the Cora dataset, and Figure (c) denotes the result of **our PiGCL** on the Cora dataset. The values in the figures represent the average similarity between classes.

the model has to reduce edge weights to mitigate the conflict due to the widespread EIC, preventing the model from acquiring sufficient topology information. Moreover, from the above derivation, it can be seen that the occurrence of the EIC is independent of whether the positive and negative sampling is correct.

The Impact of EIC

As inferred above, the encoder faces the paradox of encoding and ignoring nodes in EIC. This hampers its ability to learn from abundant negative EIC samples, concurrently affecting non-EIC negative samples.

Deriving the negative pairs similarity $k = j$ in Eq. 3, we obtain:

$$\frac{\partial \mathcal{L}_{\text{InfoNCE}}}{\partial S_{ij}} = \frac{1}{\tau} \frac{e^{S_{ij}/\tau}}{e^{S_{ii'}/\tau} + \sum_{k=1,k\neq i}^{n} (e^{S_{ik}/\tau} + e^{S_{ik'}/\tau})}, \quad S_{ij} = \theta(z_i, z_j). \tag{9}$$

From the above equation, suppose nodes v_i and v_j form a non-EIC negative pair. In the beginning of training, the gradient of this pair is normal, and its similarity can decrease appropriately. However, the similarity of many EIC negative pairs has not decreased appropriately, resulting in a larger denominator of its derivative than that under normal conditions. Consequently, this causes a reduction in its gradient magnitude, rendering even the non-EIC negative pairs incapable of conveying accurate gradient information to the encoder.

The aforementioned effects result in InfoNCE-based GCLs being unable to decrease similarity among representations. Figure 2 experimentally illustrates this issue by presenting heatmaps of representation similarities learned using the InfoNCE loss function on visual (SimCLR (Chen et al. 2020)) and graph (GRACE (Zhu et al. 2020)) datasets. Notably, on graph data the representation learned by GRACE exhibits excessive similarity between classes. This hampers the downstream tasks' ability to differentiate such closely related data, thereby undermining their accuracy. In stark contrast, our PiGCL significantly reduces the similarity between inter-class nodes.

The Potential of EIC

While the EIC issue obstructs model learning from negative pairs, it does not mean that the EIC negative pairs are useless. Quite the contrary, if isolated from the EIC issue, these negative pairs can provide invaluable information for model training.

EIC negative pairs pervade across the entire graph. As postulated by the "Small-World" theory (Watts and Strogatz 1998), the diameter of the graph is remarkably small, indicating a high degree of interconnectivity among nodes. When we align this with the EIC occurrence principles outlined in the preceding section, it becomes evident that a substantial ratio of negative pairs are susceptible to EICs. The abundance of EIC negative pairs should not be deemed entirely futile for model training. On the contrary, they significantly contribute to the training. From the perspective of gradient, the derivative of the similarity of positive cases in Eq. 3 can be obtained as follows:

$$\frac{\partial \mathcal{L}_{\text{InfoNCE}}}{\partial S_{ii'}} = -\frac{1}{\tau} \frac{\sum_{k=1,k\neq i}^{n} (e^{S_{ik}/\tau} + e^{S_{ik'}/\tau})}{e^{S_{ii'}/\tau} + \sum_{k=1,k\neq i}^{n} (e^{S_{ik}/\tau} + e^{S_{ik'}/\tau})}, \tag{10}$$

The above equation reveals that $\sum_{k=1,k\neq i}^{n} (e^{S_{ik})/\tau} + e^{S_{ik'}/\tau})$, the sum of the similarity of all negative pairs, impacts the gradient of positive samples. The larger it is, the more the model tends to encode the consistency between positive pairs. Moreover, a greater similarity difference among connected nodes is beneficial for downstream tasks. For classification tasks, the greater similarity between class-margin nodes and other class connected nodes eases the downstream pre-head's distinction. For regression tasks, the uniqueness of nodes assists in capturing the distinctive inherent properties of different nodes.

Therefore, the solution to the EIC lies not in improving the sampling method to eliminate some negative samples, but in designing a sensible training mechanism that enables the model to fully utilize EIC negative pairs for training purposes.

Methodology

In this section, we introduce the proposed method, PiGCL. As shown in Fig. 3, PiGCL utilizes the InfoNCE-based GCL framework, with the addition of two novel components: the EIC capturer and EIC controller.

Gradient-based EIC capturer

In order to address the EIC issue, the first is to capture negative pairs in the EIC state and have a sensible method of measuring their impact on model training. We designed an EIC capturer capable of indirectly gauging the influence of negative pairs on the encoder's gradient, and identifying those pairs causing gradient confusion. Specifically, the capturer aims to obtain the impact on the gradient of the encoder due to the introduction of a particular negative sample, as follows:

$$\nabla W = \frac{\partial \mathcal{L}_{\text{InfoNCE}}}{\partial W}, \tag{11}$$

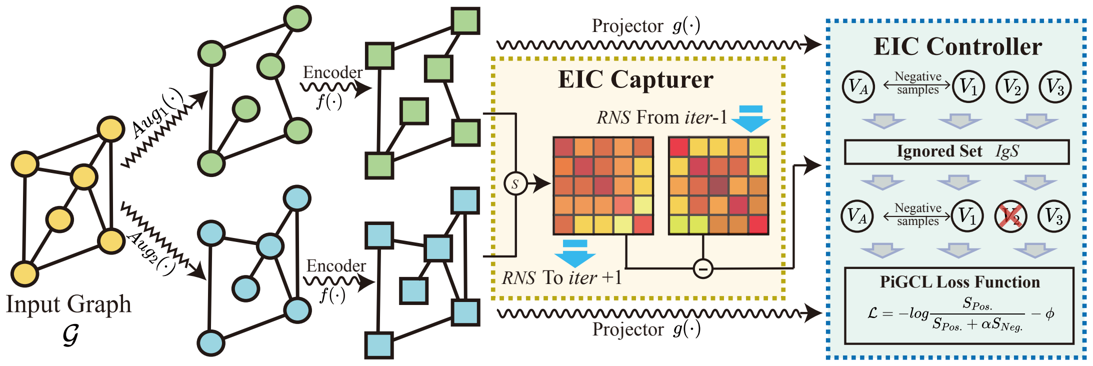


Figure 3: The overview of PiGCL. Given an input graph $\mathcal{G}$, two augmented graphs are generated through $Aug_1(\cdot)$ and $Aug_2(\cdot)$, which involve random feature masking and edge dropping. A encoder $f(\cdot)$ is then used to generate node representations, which are fed into the EIC capturer to compute a negative similarity matrix RNS for negative pairs. This RNS is compared with the one from the previous iteration to approximate the impact of negative pairs on the model gradients, thus capturing the EIC negative samples. The EIC controller uses the capturer results to generate an Ignored Set IgS, which ignore some negative samples in the current iteration. PiGCL also tone up the gradient of the objective function, which maximize consistency between positive samples and reduce it between negative samples.

where, W denotes the encoder parameters. The EIC negative pairs do not exist in isolation, necessitating the combined calculation of their influence with other negative pairs. However, directly calculating the ∇W caused from some negative pairs by the above formula is mathematically challenging. Therefore, we devised an iterative strategy that uses the amount of similarity change produced by negative pairs during the training as an approximation to measure their impact on the gradient, as shown in below:

$$GI_{NP(i,j)} = S_{ij}^{iter} - S_{ij}^{iter-1}, iter \geq 1, \qquad (12)$$

where $GI_{NP(i,j)}$ represents the gradient impact of the negative pair consisted of v_i and v_j. S_{ij}^{iter} represents the similarity between node v_i and v_j at the current iteration. We set every epo epochs as one iteration and store the current S_{ij}^{iter}. If a negative pair is not implicated in EIC or if the EIC minimally impacts the nodes involved in the negative pair, then the similarity of the negative pair can decrease swiftly, resulting in a smaller value for $GI_{NP(i,j)}$. Conversely, if the similarity does not decrease appropriately or even escalates, the value for $GI_{NP(i,j)}$ is larger. During the training process, we dynamically calculate the $GI_{NP(i,j)}$ value of negative pairs to continuously assess their impact on the model. Furthermore, negative pairs that are associated with larger $GI_{NP(i,j)}$ values can also be identified as having completed training. These pairs can be eliminated to introduce new negative pairs without inducing EIC.

Additionally, we considered capturing EIC through its cause, i.e., using N-hop neighbors in the structure. However, this approach captures too many negative pairs and makes it challenging to appropriately measure how each hop's negative samples impact the training. Calculating the receptive field conflict of negative samples for every two pairs of negative pairs would also result in excessive computational overhead. Therefore, we opted for a gradient-based capturing mechanism.

Ignore-based EIC Controller

We have designed an EIC controller to dynamically ignore partial negative pairs during the training. We first set an ignore ratio $r\%$, and select negative pairs with larger values in Eq. 12 to form an ignore set IgS as follows:

$$IgS = \{NP(i,j) | GI_{NP(i,j)} > GI_r\}, \qquad (13)$$

where GI_r represents the minimum value among the top r% nodes when arranged in descending order. The IgS is also dynamically updated. With the formula, we can, in real-time, select negative pairs that are in EIC and those that have completed training which are no longer needed. For the negative pairs in the IgS, the controller will temporarily ignore them and will not use them as self-supervised sampling signals for training. We cleverly used the semi-supervised nature of GCN to allow the model to adaptively learn the information.

Moreover, in the initial stage of training, most of the negative pairs in IgS are significantly affected by EIC. However, after a period of training, some negative pairs that have participated in the training have already provided enough information, and their similarity will not decrease further, thus they will enter the IgS set. At this time, the negative pairs that were ignored at the beginning will re-participate in the training without producing EIC. This is the essential difference between PiGCL and the improved sampling algorithm: we focus on improving the training strategy.

Toning Up Gradient with Compensation

We designed a dilation coefficient, α, for gradient toning up. This is because when partial nodes are ignored, the corresponding gradient (i.e., formulas 9 and 10) would also decrease. Moreover, the degree of this decrease is further amplified due to the involved exponential and logarithmic operations. The α coefficient is defined as follows:

$$\alpha = \frac{\sum_{i=1}^{n} \sum_{j=1}^{n} e^{\theta(z_i, z_j)/\tau}}{\sum_{i=1}^{n} \sum_{j=1, NP(i,j) \notin IgS}^{n} e^{\theta(z_i, z_j)/\tau}}. \qquad (14)$$

Method	**Available Data**	**Cora**	**CiteSeer**	**PubMed**	**CS**	**Photo**	**Computers**
Raw Features	X	64.80	64.60	84.80	90.37	78.53	73.81
Node2vec	A	74.80	52.30	80.30	85.08	89.67	84.39
DeepWalk	A	75.70	50.50	80.50	84.61	89.44	85.68
DeepWalk + Features	X, A	73.10	47.60	83.70	87.70	90.05	86.28
BGRL	X, A	81.33 ± 0.31	70.57 ± 0.98	85.86 ± 0.15	92.37 ± 0.22	92.36 ± 0.09	87.28 ± 0.34
MVGRL	X, A	84.32 ± 0.97	72.29 ± 0.75	85.33 ± 0.25	92.28 ± 0.19	92.07 ± 0.26	87.68 ± 0.31
DGI	X, A	83.24 ± 0.73	71.91 ± 0.85	85.67 ± 0.28	92.86 ± 0.15	92.56 ± 0.41	86.93 ± 0.25
GBT	X, A	83.50 ± 1.05	69.12 ± 1.39	85.29 ± 0.34	92.63 ± 0.14	92.52 ± 0.34	87.30 ± 0.29
GRACE	X, A	83.30 ± 0.40	71.65 ± 1.03	85.69 ± 0.20	92.06 ± 0.18	92.13 ± 0.20	87.13 ± 0.37
GCA	X, A	83.42 ± 0.78	70.97 ± 1.32	86.12 ± 0.22	93.01 ± 0.21	92.15 ± 0.26	88.04 ± 0.34
Local-GCL	X, A	83.08 ± 0.84	71.00 ± 0.94	85.67 ± 0.28	92.48 ± 0.12	92.99 ± 0.34	88.64 ± 0.25
ProGCL	X, A	82.80 ± 1.03	72.85 ± 0.92	OOM	OOM	92.85 ± 0.32	OOM
PiGCL(Ours)	X, A	**84.63 ± 0.78**	**73.51 ± 0.64**	**86.75 ± 0.20**	**93.30 ± 0.09**	**93.14 ± 0.30**	**89.25 ± 0.27**
Supervised GCN	X, A, Y	82.80	72.00	84.80	93.03	92.42	86.51

Table 1: Performance on node classification. X, A, Y denote the node attributes, adjacency matrix, and labels in the datasets, OOM signifies out-of-memory on 24GB RTX 3090. Data without deviation are drawn from (Zhu et al. 2021, 2020).

For negative samples, we multiply them by α to magnify their gradient. For positive samples, we added a compensatory term, ϕ, as follows.

$$\phi = (1 - 1/\alpha) \cdot \frac{1}{N} \sum_{i=1}^{n} S_{ii'}. \tag{15}$$

By complementing the gradient, the numerical value of the gradient can remain relatively constant even when plenty of negative samples are ignored. This prevents the model from converging prematurely due to excessively small gradients caused by a large r. Consequently, we derive the loss function for PiGCL as follows:

$$\mathcal{L}_{\text{PiGCL}}(z_i) = -\log(\frac{e^{S_{ii'}/\tau}}{e^{S_{ii'}/\tau} + \alpha(\sum_{k=1, NP(i,k) \notin IgS}^{n} (e^{S_{ik}/\tau} + e^{S_{ik'}/\tau}))}) - \phi. \tag{16}$$

Experiment

Experimental Setup

Datasets. We evaluate our method on six widely-used datasets, Cora, Citeseer, and PubMed from the Plantoid (Kipf and Welling 2017), Photo and Computers from the Amazon (McAuley et al. 2015), and CS (Sinha et al. 2015) from the Coauthor. Detailed descriptions are in Appendix C.

Compared Methods. We primarily compare our PiGCL with four categories of models: (1)Representative methods under the InfoNCE-based framework: GRACE (Zhu et al. 2020), GCA (Zhu et al. 2021) with improved augmentation, Local-GCL (Zhang et al. 2022) and ProGCL (Xia et al. 2021) with improved sampling methods. (2) Other recent GSSL baselines: BGRL (Thakoor et al. 2021), MVGRL (Hassani and Khasahmadi 2020), DGI (Veličković et al. 2019), and GBT (Bielak, Kajdanowicz, and Chawla 2022). (3) The classical GSSL algorithms: Node2vec (Grover and Leskovec 2016) and Deepwalk (Perozzi, Al-Rfou, and Skiena 2014). (4) The semi-supervised training baselines GCN (Kipf and Welling 2017). Further detailed introduction of these baselines can be found in Appendix A.

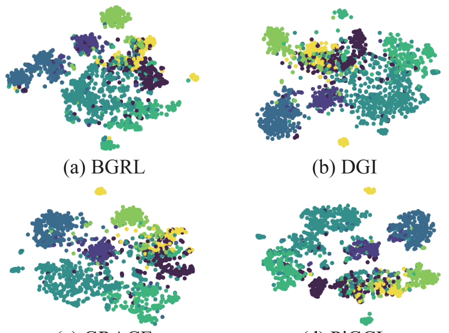
Detailed Setup. We test PiGCL on both classification and clustering tasks. All GSSL models are first trained self-supervisedly with a two-layer GCN encoder. The settings for the hyperparameters can be found in the Appendix C. We utilise the encoder to generate representation for downstream tasks. For the classification tasks, we follow the testing method from the GRACE (Zhu et al. 2020). We train and test a downstream $l2$ regularized logistic regression classifier using the learned representations. Specifically, we use 10% of the data for training the downstream classifier and the remaining 90% for testing. We run this 20 times and report the average and variance of F1-score. For clustering tasks, we directly input the obtained representations into a randomly initialized K-Means (**?**) predictor with a maximum of 500 iterations. We run this 10 times and report the average NMI and ARI. All experiments were conducted on a server equipped an RTX 3090 GPU and an i5-12400 CPU. More details and the source code are available at https://github.com/hedongxiao-tju/PiGCL.

Performance Analysis

Classification. Tab. 1 presents the F1-score of all methods on the node classification task. From the table, it is observed that the proposed PiGCL shows its strong performance across six datasets. Compared to the baseline method GRACE, our method improve the accuracy by an average of 1.43%, which validates that our proposed method has addressed the EIC issue during the training. PiGCL also shows an improvement over methods like GCA, Local-GCL, and ProGCL, which only modify the augmentation and sampling strategy. This highlights the significant impact of the EIC issue on model performance. Furthermore, the approach of PiGCL can be applied to these improved methods to further enhance their performance. We also found that GRACE, which takes into account the inherent negative relationships between nodes within the dataset, does not show significant advantages compared to other baselines (such as BGRL, DGI and GBT) that do not consider nodes relationships in dataset. However, PiGCL, which solves the EIC issue, outperforms these other types of baseline methods. This further proves that the EIC issue prevents effective utilization

		BGRL	DGI	GRACE	GBT	GCA	Local-GCL	Pro-GCL	**PiGCL(Ours)**
Cora	NMI	0.4211	0.5370	0.4758	0.4562	0.4510	0.5386	0.5131	**0.5494**
	ARI	0.2905	0.4469	0.3633	0.3683	0.3104	0.4479	0.3434	**0.4670**
CiteSeer	NMI	0.3748	0.4185	0.3960	0.3414	0.3737	0.4508	0.4115	**0.4581**
	ARI	0.3855	0.4140	0.3977	0.3193	0.3675	0.4494	0.4219	**0.4720**
PubMed	NMI	0.3149	0.3188	0.3508	0.2992	0.3307	0.3469	OOM	**0.3784**
	ARI	0.2928	0.3165	0.3286	0.2942	0.2919	0.3304	OOM	**0.3612**

Table 2: Performance on node clustering


(a) BGRL (b) DGI

(c) GRACE (d) PiGCL

Figure 4: t-SNE embeddings of nodes in Cora dataset.

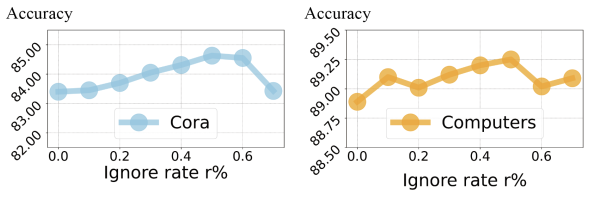


Figure 5: Analysis of parameter ignore radio r

of these negative samples in GRACE.

Clustering. In the clustering task, we evaluated the PiGCL on the Cora, CiteSeer, and PubMed datasets, as shown in Tab. 2. Once again, PiGCL shows a strong performance, improving on the baseline GRACE by an average of 5.44% in NMI and 7.02% in ARI, and outperforming all other compared methods. It can be observed that the clustering accuracy of GRACE is inferior to the BGRL and DGI. GRACE's InfoNCE-based loss function uses negative sampling information to separate the nodes, thereby reducing node similarity and aiding the clustering task. However, due to its struggle with the EIC issue, the negative samples are not effectively utilized. The clustering accuracy of PiGCL significantly surpasses GRACE, demonstrating the success of PiGCL's strategy of dynamically ignoring EIC negative samples. This proves that while the number of negative samples decreases in every epoch for PiGCL, it still can effectively learn from these ignored negative samples due to the semi-supervised nature of GNNs.

Visualization. To provide a more intuitive demonstration of the advantages of the representations learned by PiGCL, we employ t-SNE (**?**) to visualize the representations learned from the Cora dataset, as shown in Fig. 4. It is evident that, compared with the other three baselines, the representations learned by PiGCL exhibit broader and clearer boundaries between classes, which are consistent with the similarity finding in Fig. 2. This firstly validates the effectiveness of PiGCL's strategy in handling EIC negative node pairs, demonstrating a stronger capability of reducing the similarity of negative pairs than GRACE. Furthermore, the cohesiveness of intra-class nodes in PiGCL, as shown in Figure 4, either remains unchanged or elevates. This results from PiGCL's utilization of the semi-supervised nature of GCN to adaptively learn the sampling information that is dynamically ignored. This further corroborates that PiGCL can effectively utilize negative samples.

Parameters Analysis

We examined the impact of the hyperparameter ignore rate r on model performance on datasets with the most and least edges (Cora, Computers), as illustrated in Figure 5. It is observed that the accuracy where $r > 0$ is generally superior to that of $r = 0$, which establishes the benefit of dynamically disregarding some EIC negative pairs to mitigate the generation of EIC and enhance model learning from negative samples. The model overall exhibits a unimodal pattern. As r increases, the model can further reduce EIC by ignoring more negative pairs. While, when r becomes too large, although the ignore set is updated dynamically, the model fails to obtain enough negative pairs for training, thus leading to a performance decline. This once again demonstrates the importance of negative samples for training and consistent with our previous view.

Conclusion

In this paper, we are the first to identify and investigate the implicit Embedding and Ignoring conflict (EIC). The EIC issue is widespread in InfoNCE-based GCL and lead to gradient confusion during training. To address the EIC issue, we propose PiGCL, which dynamically captures and ignores a portion of self-supervised negative sampling during the training based on gradient detection. Leveraging the inherent semi-supervised nature of graph encoders, PiGCL makes the encoder adaptively learn from the ignored negative samples. Thus, PiGCL greatly improves the utilization of negative sampling. A large number of experimental results confirm the effectiveness of the proposed method.

Acknowledgements

This work is supported by National Key Research and Development Program of China (2023YFC3304503), National Natural Science Foundation of China (62276187).

References

Bielak, P.; Kajdanowicz, T.; and Chawla, N. V. 2022. Graph barlow twins: A self-supervised representation learning framework for graphs. *Knowledge-Based Systems*, 256: 109631.

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

Glorot, X.; Bordes, A.; and Bengio, Y. 2011. Deep sparse rectifier neural networks. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, 315–323. JMLR Workshop and Conference Proceedings.

Grover, A.; and Leskovec, J. 2016. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, 855–864.

Hamilton, W.; Ying, Z.; and Leskovec, J. 2017. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30: 1024–1034.

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

He, D.; Zhao, J.; Guo, R.; Feng, Z.; Jin, D.; Huang, Y.; Wang, Z.; and Zhang, W. 2023. Contrastive Learning Meets Homophily: Two Birds with One Stone. In *International Conference on Machine Learning*.

Huo, C.; Jin, D.; Li, Y.; He, D.; Yang, Y.-B.; and Wu, L. 2023. 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*, 4339–4346.

Jaiswal, A.; Babu, A. R.; Zadeh, M. Z.; Banerjee, D.; and Makedon, F. 2020. A survey on contrastive self-supervised learning. *Technologies*, 9(1): 2.

Kipf, T. N.; and Welling, M. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *International Conference on Learning Representations*.

Liu, N.; Wang, X.; Bo, D.; Shi, C.; and Pei, J. 2022. Revisiting graph contrastive learning from the perspective of graph spectrum. *Advances in Neural Information Processing Systems*, 35: 2972–2983.

McAuley, J.; Targett, C.; Shi, Q.; and Van Den Hengel, A. 2015. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*, 43–52.

Perozzi, B.; Al-Rfou, R.; and Skiena, S. 2014. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, 701–710.

Shi, Z.; Chen, J.; Li, K.; Raghuram, J.; Wu, X.; Liang, Y.; and Jha, S. 2023. The Trade-off between Universality and Label Efficiency of Representations from Contrastive Learning. In *International Conference on Learning Representations*.

Sinha, A.; Shen, Z.; Song, Y.; Ma, H.; Eide, D.; Hsu, B.-J.; and Wang, K. 2015. An overview of microsoft academic service (mas) and applications. In *Proceedings of the 24th international conference on world wide web*, 243–246.

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

Thakoor, S.; Tallec, C.; Azar, M. G.; Munos, R.; Veličković, P.; and Valko, M. 2021. Bootstrapped representation learning on graphs. In *ICLR 2021 Workshop on Geometrical and Topological Representation Learning*.

Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Lio, P.; and Bengio, Y. 2018. Graph Attention Networks. In *International Conference on Learning Representations*.

Veličković, P.; Fedus, W.; Hamilton, W. L.; Liò, P.; Bengio, Y.; and Hjelm, R. D. 2019. Deep Graph Infomax. In *International Conference on Learning Representations*.

Vlaic, S.; Conrad, T.; Tokarski-Schnelle, C.; Gustafsson, M.; Dahmen, U.; Guthke, R.; and Schuster, S. 2018. ModuleDiscoverer: Identification of regulatory modules in protein-protein interaction networks. *Scientific reports*, 8(1): 433.

Wang, H.; Zhang, J.; Zhu, Q.; and Huang, W. 2022. Augmentation-free graph contrastive learning with performance guarantee. *arXiv preprint arXiv:2204.04874*.

Watts, D. J.; and Strogatz, S. H. 1998. Collective dynamics of 'small-world'networks. *nature*, 393(6684): 440–442.

Wu, S.; Tang, Y.; Zhu, Y.; Wang, L.; Xie, X.; and Tan, T. 2019. Session-based recommendation with graph neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, 346–353.

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

Xia, J.; Wu, L.; Chen, J.; Hu, B.; and Li, S. Z. 2022. Simgrace: A simple framework for graph contrastive learning without data augmentation. In *Proceedings of the ACM Web Conference 2022*, 1070–1079.

Xia, J.; Wu, L.; Wang, G.; Chen, J.; and Li, S. Z. 2021. Progcl: Rethinking hard negative mining in graph contrastive learning. In *International conference on machine learning*, 24332–24346.

Xu, K.; Hu, W.; Leskovec, J.; and Jegelka, S. 2018. How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826*.

You, Y.; Chen, T.; Sui, Y.; Chen, T.; Wang, Z.; and Shen, Y. 2020. Graph contrastive learning with augmentations. *Advances in neural information processing systems*, 33: 5812–5823.

Zhang, H.; Wu, Q.; Wang, Y.; Zhang, S.; Yan, J.; and Yu, P. S. 2022. Localized Contrastive Learning on Graphs. *arXiv preprint arXiv:2212.04604*.

Zhang, Y.; Xiong, Y.; Ye, Y.; Liu, T.; Wang, W.; Zhu, Y.; and Yu, P. S. 2020. SEAL: Learning heuristics for community detection with generative adversarial networks. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 1103–1113.

Zhou, J.; Cui, G.; Hu, S.; Zhang, Z.; Yang, C.; Liu, Z.; Wang, L.; Li, C.; and Sun, M. 2020. Graph neural networks: A review of methods and applications. *AI open*, 1: 57–81.

Zhu, Y.; Xu, Y.; Yu, F.; Liu, Q.; Wu, S.; and Wang, L. 2020. Deep graph contrastive representation learning. *arXiv preprint arXiv:2006.04131*.

Zhu, Y.; Xu, Y.; Yu, F.; Liu, Q.; Wu, S.; and Wang, L. 2021. Graph contrastive learning with adaptive augmentation. In *Proceedings of the Web Conference 2021*, 2069–2080.