RETHINKING TEMPERATURE IN GRAPH CONTRASTIVE LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

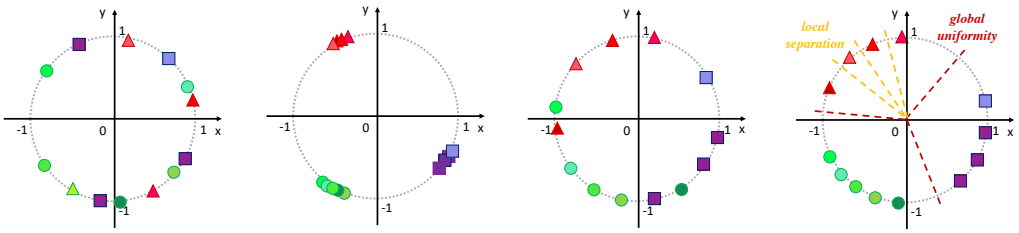
Due to not relying on the rare human-labeled information, self-supervised learning, especially contrastive learning, attracted much attention from researchers. It has begun to show its strong advantages on both IID data (independent and identically distributed data, such as images and texts) and Non-IID data (such as nodes in graphs). Recently, researchers have begun to explore the quality of contrastive representations and proposed some metrics for measuring it, such as alignment, uniformity, and semantic closeness. However, current studies only consider IID data and ignore the evaluation of representations quality in graph contrastive learning. In this paper, we investigate and discuss how to generate high-quality representations for a general loss (InfoNCE) in graph contrastive learning. We argue that the properties of **global uniformity** and **local separation** are both necessary to the representation quality. By theoretical analysis, we find that the two properties can be naturally regulated by temperature τ in InfoNCE loss. Based on this point, we develop a simple but effective algorithm GLATE to dynamically adjust the temperature value in the training phase. On node and graph classification tasks, GLATE is validated to be competitive with the state-of-the-art graph contrastive learning algorithms.

1 INTRODUCTION

Self-supervised learning provides a good learning paradigm without high-cost label information for computer vision (Chen et al. (2020); Chen & He (2021); Grill et al. (2020)), natural language processing (Wu et al. (2019); Gao et al. (2021)), and speech recognition (Ravanelli et al. (2020); Kharitonov et al. (2021)). Contrastive-based methods have a prominent place among numerous self-supervised learning methods (Jaiswal et al. (2021); Le-Khac et al. (2020)). Recently, some researchers have explored graph contrastive frameworks (consisting of data augmentation, network encoding, and contrastive learning) for self-supervised learning on graphs (Velickovic et al. (2019); Peng et al. (2020); Zhu et al. (2020; 2021c); Thakoor et al. (2021)). On benchmark datasets, the state-of-the-art graph contrastive learning (GCL) algorithms have been verified to be competitive with or even superior to the supervised learning algorithms in downstream tasks, such as node classification and graph classification.

Investigating the quality of contrastive representations is important for contrastive learning, but it is absent in the domain of graphs. Wang & Isola (2020) propose two novel loss functions, alignment loss $\mathcal{L}_{\text{align}}$ and uniformity loss $\mathcal{L}_{\text{uniform}}$, for measuring the image representation quality. By optimizing the integration of $\mathcal{L}_{\text{align}}$, $\mathcal{L}_{\text{uniform}}$, and contrastive loss, the performance on the downstream task (such as image classification on IMAGENET (Tian et al. (2019))) becomes better. However, $\mathcal{L}_{\text{align}}$ and $\mathcal{L}_{\text{uniform}}$ designed for independent and identically distributed (IID) data are not suitable for Non-IID data such as nodes in a graph.

The InfoNCE (here "NCE" denotes "Noise-Contrastive Estimation") contrastive loss (Oord et al. (2018)), as a universal loss function of contrastive learning, generally samples negative examples uniformly from the whole training data. To optimize the InfoNCE loss, the strategy of hard negative sampling (Zhuang et al. (2019); Robinson et al. (2021)) collects highly similar but true negative examples in advance to construct contrastive pairs and is widely used in image processing and metric learning (Duan et al. (2018); Robinson et al. (2021); Wang & Liu (2021)). This strategy makes the model be able to correct its mistakes quickly and improve the semantic closeness between represen-



(a) Initial random state. (b) High τ (fixed). (c) Low τ (fixed). (d) Dynamic τ.

Figure 1: The normalized embedding distribution on the unit hypersphere (here we simplify it as the two-dimensional case for a better demonstration). For each node, we use the shape and color to represent its label and embedding, respectively. (a) is the initial random state; (b), (c), and (d) correspond to the embedding distribution when temperature τ is high (fixed), low (fixed), and dynamically adjusted, respectively. When τ is relatively high, the same penalties on all negative examples make the embedding too compact in the local scope. When τ is relatively low, the high penalty on the hard negative examples (highly similar to the anchor example) will lead to the misplacement of embeddings. By dynamically changing τ's value, the final result accords with the global uniformity and local separation.

tations (Wang & Liu (2021)). Despite hard negative sampling's practical importance in the image domain, Yang et al. (2020) have proved that it is sub-optimal in the graph domain.

In this paper, we investigate how to generate high-quality node representations under the InfoNCE contrastive objective in graphs. We highlight the importance of embeddings' global uniformity and local separation for GCL. The temperature coefficient τ, as a key component of InfoNCE loss, decides how the current learning state focuses on global uniformity and local separation. We illustrate this conclusion in Figure 1 and prove it by a gradient analysis in Section 3. The dynamic setting of τ can generate different learning states for the same task and help the algorithm smoothly transit from one state to another. Therefore, inspired by the optimizer called Momentum (Qian (1999)), we develop a **G**raph contrastive **L**earning algorithm with dyn**A**mic **T**emperature **E**stimation (GLATE). Compared with the fixed setting of τ, GLATE develops to its full potential on contrastive learning by further maximizing the self-supervised Information Bottleneck objective.

In a nutshell, the main contributions of this work are as follows:

- To evaluate the node representations in GCL, we propose two new metrics: global uniformity (Eq. (9)) and local separation (Eq. (10)). We prove the importance of temperature τ to them by theoretical analysis (Section 3.1).
- We develop a simple but very effective GCL algorithm GLATE (Section 3.2). In the training phase, GLATE dynamically adjusts τ's value with its momentum to learn high-quality representations. With the help of the information bottleneck principle, we analyze the connections between GLATE and entropy information (Section 3.3).
- The experimental results on node and graph classification tasks indicate that GLATE is superior to the state-of-the-art GCL algorithms (Section 4). For example, for the task of transductive node classification, GLATE outperforms baselines by 2.8 percent on average.

2 Background and Related Work

Preliminaries of Graph Contrastive Learning. Let us review an overall graph contrastive learning process with contrastive pairs. We consider a graph $\mathcal{G} = (\boldsymbol{A}, \boldsymbol{X})$ where $\boldsymbol{A} \in \mathbb{R}^{N \times N}$ denotes the adjacency matrix, $\boldsymbol{X} \in \mathbb{R}^{N \times M}$ denotes the attribute distribution matrix, N is the number of nodes, and M is the dimension of attributes. The raw graph $\mathcal{G}$ is distorted via random data augmentation strategies (such as edge removing and attribute masking) to generate two new graphs $\tilde{\mathcal{G}}_1 = (\boldsymbol{A}', \boldsymbol{X}')$ as well as $\tilde{\mathcal{G}}_2 = (\boldsymbol{A}'', \boldsymbol{X}'')$. Then, the message-passing-based graph neural network (e.g., GCN proposed in (Kipf & Welling (2016))) is used as a shared encoder of $\tilde{\mathcal{G}}_1$ and $\tilde{\mathcal{G}}_2$. It aims to learn node

embeddings from $\tilde{\mathcal{G}}_1$ and $\tilde{\mathcal{G}}_2$. The message passing form of a two-layer GCN is:

$$\boldsymbol{Z}' = \text{softmax}(\hat{\boldsymbol{A}}'\text{ReLU}(\hat{\boldsymbol{A}}'\boldsymbol{X}'\boldsymbol{W}^0)\boldsymbol{W}^1), \tag{1}$$

where $\boldsymbol{Z}'$, $\hat{\boldsymbol{A}}'$, and $\boldsymbol{W}^0$ ($\boldsymbol{W}^1$) are node embedding matrix, renormalized $\boldsymbol{A}'$, and first-layer (second-layer) neural network parameters, respectively. Next, in each training epoch, the pre-defined contrastive objective encourages GCN to minimize the distance between $\boldsymbol{Z}'$ and $\boldsymbol{Z}''$ (another node embedding matrix learned from $\tilde{\mathcal{G}}_2$), and meanwhile to maximize the distance between different nodes (Zhu et al. (2020); Zbontar et al. (2021b); Bardes et al. (2021)). After n rounds of iteration, we use trained GCN to infer each node's embedding on $\mathcal{G}$, which is used as the initial value of new node embedding in different downstream tasks, such as node classification.

Graph Contrastive Objectives. To make positive examples closer and negative examples farther, different graph contrastive loss functions are defined, such as: 1) Jason-Shannon Divergence (JSD) loss (Velickovic et al. (2019); Hassani & Khasahmadi (2020)), 2) InfoNCE loss (Zhu et al. (2020); You et al. (2020); Zhu et al. (2021c); Pan & Kang (2021); Xu et al. (2021)), and 3) Triplet Margin (TM) loss (Zhang et al. (2019)). According to a recent empirical study (Zhu et al. (2021b)), the models under the InfoNCE loss generally achieve better performance compared to those under the other graph contrastive objectives with the participation of negative examples. The important role of InfoNCE loss in graph contrastive learning motivates our work to have deep insights into a series of GCL algorithms (e.g. GRACE (Zhu et al. (2020))) derived by it.

Understanding Contrastive Representation Learning. Another main area of concern is understanding contrastive representation learning and exploring what high-quality representations are. In the image domain, alignment, uniformity, and semantic closeness (Wang & Isola (2020); Wang & Liu (2021)) are thought to be three important metrics for measuring the quality of learned representations. Hard negative sampling (Robinson et al. (2021); Wang & Liu (2021)) has been proved to be a successful strategy to meet the needs of the above three metrics. However, recent studies (Xia et al. (2021); Yang et al. (2020)) have confirmed the existence of *sampling bias* when using hard negative sampling for a graph, which limits its application to the graph domain.

3 METHOD

3.1 GRADIENT ANALYSIS

We start with a general contrastive loss function, i.e., InfoNCE loss (Oord et al. (2018); Zhu et al. (2020)). In the graph domain, it can be formulated as:

$$\mathcal{L}_i^{1,2} = -\log \frac{\exp(\frac{1}{\tau}\cdot \mathrm{S}(i', i''))}{\exp(\frac{1}{\tau}\cdot \mathrm{S}(i', i'')) + \underbrace{\sum_{k\neq i}\exp(\frac{1}{\tau}\cdot \mathrm{S}(i', k''))}_{\text{inter-view term}} + \underbrace{\sum_{k\neq i}\exp(\frac{1}{\tau}\cdot \mathrm{S}(i', k'))}_{\text{intra-view term}}}. \tag{2}$$

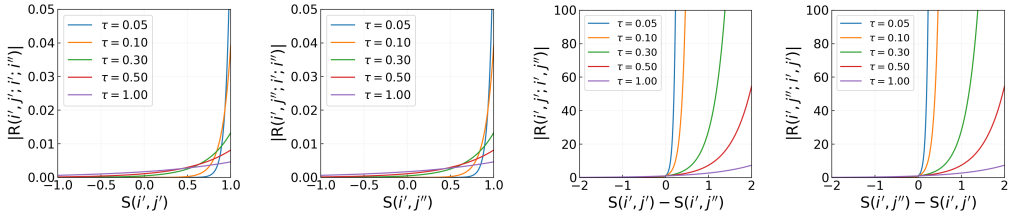
The vector similarity function $\mathrm{S}(i', k'') = f(||h(\boldsymbol{Z}'_{i,:})||_2, ||h(\boldsymbol{Z}''_{k,:})||_2)$ where $f(\cdot)$ and $h(\cdot)$ are respectively cosine similarity and non-linear projection transformation (two layers of multilayer perception). As $h(\cdot)$ is not shared between different channels, the whole network structure is technically asymmetric, which leads to the asymmetric result of learned node representations. To eliminate the bias between different channels, $\tilde{\mathcal{G}}_1$ and $\tilde{\mathcal{G}}_2$ are swapped to use, and thus the total loss function is $\mathcal{L} = \frac{1}{2N}\sum_{i=1}^{N}(\mathcal{L}_i^{1,2} + \mathcal{L}_i^{2,1})$. To simplify the symbols in Eq. (2), we replace $\exp(\frac{1}{\tau}\cdot \mathrm{S}(i', k''))$ by $\mathrm{E}_{i',k''}$, and then derive the following gradient results:

$$\nabla_{\mathrm{S}(i',i'')}\mathcal{L}_i^{1,2} = -\frac{\sum_{k\neq i}\mathrm{E}_{i',k''} + \sum_{k\neq i}\mathrm{E}_{i',k'}}{\tau\cdot(\mathrm{E}_{i',i''} + \sum_{k\neq i}\mathrm{E}_{i',k''} + \sum_{k\neq i}\mathrm{E}_{i',k'})}, \tag{3}$$

$$\nabla_{\mathrm{S}(i',j')}\mathcal{L}_i^{1,2} = -\frac{\mathrm{E}_{i',j'}}{\tau\cdot(\mathrm{E}_{i',i''} + \sum_{k\neq i}\mathrm{E}_{i',k''} + \sum_{k\neq i}\mathrm{E}_{i',k'})}, \tag{4}$$

$$\nabla_{\mathrm{S}(i',j'')}\mathcal{L}_i^{1,2} = -\frac{\mathrm{E}_{i',j''}}{\tau\cdot(\mathrm{E}_{i',i''} + \sum_{k\neq i}\mathrm{E}_{i',k''} + \sum_{k\neq i}\mathrm{E}_{i',k'})}. \tag{5}$$

These gradient results have the same denominator, which can be removed when we calculate the gradient ratio between any two of them.



(a) $\mathrm{R}(i', j'; i', i'')$ curve. (b) $\mathrm{R}(i', j''; i', i'')$ curve. (c) $\mathrm{R}(i', j'; i', j'')$ curve. (d) $\mathrm{R}(i', j''; i', j')$ curve.

Figure 2: The rate curves under different τ. In (a) and (b), lower τ means more penalties on hard negative examples. In (c) and (d), low τ deepens the imbalance update between different views.

Lemma 1. *Following Eq. (2), the ratio of intra-view negative gradient to positive gradient is*

$$\mathrm{R}(i', j'; i', i'') = \frac{\nabla_{\mathrm{S}(i',j')}\mathcal{L}_i^{1,2}}{\nabla_{\mathrm{S}(i',i'')}\mathcal{L}_i^{1,2}} = \frac{\mathrm{E}_{i',j'}}{\sum_{k\neq i}\mathrm{E}_{i',k''} + \sum_{k\neq i}\mathrm{E}_{i',k'}}. \tag{6}$$

The result in Eq. (6) obeys the Boltzmann distribution, as shown in Figure 2 (a,b) (note that we omit the specific form of $\mathrm{R}(i', j''; i', i'')$ because it has the same property as $\mathrm{R}(i', j'; i', i'')$). It means that if τ is low ($\tau \in (0, 0.5)$), the optimizer will punish more on the hard negative examples than the easy ones; otherwise, it will nearly make no difference of them. So high τ is good at distinguishing similar and dissimilar embeddings and low τ is good at partitioning highly similar embeddings to ensure local separation.

It is also worth noting the rate between the negative gradients of different views, which is defined as:

$$\mathrm{R}(i', j'; i', j'') = \frac{\nabla_{\mathrm{S}(i',j')}\mathcal{L}_i^{1,2}}{\nabla_{\mathrm{S}(i',j'')}\mathcal{L}_i^{1,2}} = \frac{\mathrm{E}_{i',j'}}{\mathrm{E}_{i',j''}} = \exp(\frac{1}{\tau}\cdot(S(i', j') - S(i', j''))). \tag{7}$$

In Eq. (7), τ still plays an important role (note that we omit the specific form of $\mathrm{R}(i', j''; i', j')$ because it has the same property as $\mathrm{R}(i', j'; i', j'')$). We have the following proposition:

Proposition 1. *(Different views' imbalanced update.). The temperature τ is crucial to balance the rate between the negative gradients of different views. If $\tau \to \infty$, then $\nabla_{\mathrm{S}(i',j')}\mathcal{L}_i^{1,2} \to \nabla_{\mathrm{S}(i',j'')}\mathcal{L}_i^{1,2}$, i.e., the negative examples of inter-view and intra-view keep the gradient update at the same speed. Otherwise, there exists an imbalance between $\nabla_{\mathrm{S}(i',j')}\mathcal{L}_i^{1,2}$ and $\nabla_{\mathrm{S}(i',j'')}\mathcal{L}_i^{1,2}$. The imbalance will deepen as τ decreases.*

We refer the reader to Appendix A.1 for detailed proof. We illustrate the phenomenon of the imbalanced gradient update in Figure 2 (c) and (d). Based on the above analysis, we conclude that 1) high τ is helpful to the embedding's global uniformity and low τ is helpful to the embedding's local separation; 2) high τ can alleviate the negative effect of imbalanced update. Therefore, dynamic estimation of τ with a relatively high initial value is recommended.

3.2 Dynamic Temperature Estimation with Momentum

Some studies (Robinson et al. (2021); Wang & Liu (2021)) have confirmed that the properties of uniformity and semantic closeness are both necessary for image contrastive learning. In the graph domain, the closeness of nodes embodies the proximity of the topology information and the similarity of the attribute information. Both of them are incorporated into $\boldsymbol{Z}'$, $\boldsymbol{Z}''$ by the encoder of graph neural networks with the message passing mechanism.

The strategy of hard negative sampling is successful to discriminate the images of similar semantics. It only updates the hard or informative examples by truncating the uninformative ones. Following Eq. (2), the loss under hard negative sampling can be formulated as: $\mathcal{L}_{i,hard}^{1,2} = -\log(\mathrm{E}_{i',i''}/(\mathrm{E}_{i',i''} + \sum_{S_{i',k''}\geqslant\delta}\mathrm{E}_{i',k''} + \sum_{S_{i',k'}\geqslant\delta}\mathrm{E}_{i',k'}))$. Although this strategy is rational in the image domain (Robinson et al. (2021); Wang & Liu (2021); Zhu et al. (2021a)), it is not suitable for the graph (we also compare the proposed algorithm with its variant using hard negative sampling, refer to Table 1). Current negative sampling methods in the graph have a common shortcoming of

sampling bias, i.e., most of the negative examples being highly similar to the anchor example are actually positive examples (Xia et al. (2021)). Yang et al. (2020) have proved that a nice negative sampling distribution should be sub-linearly correlated to the positive sampling distribution in the graph domain: $p_{neg}(i|j) \propto p_{pos}(i|j)^{\alpha},\ \alpha \in (0,1)$. As the real p_{pos} is unknown and its approximation is hard to define, we retain all negative examples and employ dynamic temperature estimation to control global uniformity and local separation.

Based on the gradient analysis in Section 3.1, we propose a new algorithm called GLATE whose main idea is dynamic temperature estimation. More precisely, at the early stage, the temperature τ is set high to ensure that all embeddings are split uniformly by imposing nearly equal punishments on negative examples. As the training goes on, the dissimilar examples are separated gradually and the distance between similar examples becomes close. At the later stage, τ is set relatively lower to distinguish the similar and hard negative examples. Therefore, the change of temperature should be influenced by the uniformity of the node representations.

Inspired by Momentum (Qian (1999)), a classical stochastic gradient descent optimization method, we calculate the temperature's current value τ_t by integrating its last state's momentum $\Delta\tau_{t-1}$ and the current embedding's global uniformity degree D^t_{global} together:

$$\tau_t = \tau_{t-1} \underbrace{-\varepsilon(p\Delta\tau_{t-1} + \frac{1}{D^t_{global}})}_{\Delta\tau_t}. \tag{8}$$

Also, the global uniformity degree at epoch t D^t_{global} is defined as:

$$D^t_{global} = -\log\frac{2}{N(N-1)}\sum_i\sum_{j<i}\exp^{-||\mathbf{z}^t_i - \mathbf{z}^t_j||^2_2}, \tag{9}$$

where ε is the learning rate and p is the momentum parameter. In addition, we define the local separation degree at epoch t and use it as an important metric to measure the model's uniformity from a local perspective (Figure 4):

$$D^t_{local} = -\log\frac{2}{N(N-1)}\sum_i\sum_{j<i}\mathrm{S}(\mathbf{Z}^t_i, \mathbf{Z}^t_j)\cdot\exp^{-||\mathbf{z}^t_i - \mathbf{z}^t_j||^2_2}, \tag{10}$$

where $\mathrm{S}(\mathbf{Z}^t_i, \mathbf{Z}^t_j)$ is the cosine similarity between node embeddings $\mathbf{Z}_i$ and $\mathbf{Z}_j$ at epoch t.

Compared with the steepest descent $\Delta\tau_t = -\varepsilon/D^t_{global}$, the update rule in Eq. (8) makes τ decrease at an adjustable pace. At the early stage, τ decreases slowly due to little information accumulation, while it decreases faster later to distinguish harder negative examples.

3.3 Analysis from the Information Bottleneck Principle

Here we analyze GLATE's intrinsic mechanism and reveal the connection between GLATE and the information bottleneck principle (Tishby & Zaslavsky (2015); Amjad & Geiger (2019)). Assuming the node representation matrix obeys the D-dimensional Gaussian distribution: $\boldsymbol{Z}' \sim \mathcal{N}(\boldsymbol{\mu}_{\boldsymbol{Z}'}, \boldsymbol{\Sigma}_{\boldsymbol{Z}'})$ ($\boldsymbol{\mu}_{\boldsymbol{Z}'} \in \mathbb{R}^{1\times D}$, $\boldsymbol{\Sigma}_{\boldsymbol{Z}'} \in \mathbb{R}^{D\times D}$), the self-supervised learning's information bottleneck (IB) principle (Zbontar et al. (2021a)) is to maximize the following Lagrange's function (we omit $\boldsymbol{Z}''$ in this section because it has the same properties as $\boldsymbol{Z}'$):

$$\mathcal{IB}_{ssl} = I(\boldsymbol{Z}', \boldsymbol{X}) - \beta I(\boldsymbol{Z}', \boldsymbol{X}') \quad (\beta > 0), \tag{11}$$

where $I(\mathbf{a}, \mathbf{b})$ is the mutual information between $\mathbf{a}$ and $\mathbf{b}$. The motivation behind $\mathcal{IB}_{ssl}$ is to make the learned embedding preserve as much feature information of the raw input data $\boldsymbol{X}$ as possible and evade the noise in the augmented data $\boldsymbol{X}'$. In addition, a basic property between mutual information and entropy information is as follow:

Property 1. *The relation between mutual information and conditional entropy is* $I(\boldsymbol{X}, \boldsymbol{Y}) = H(\boldsymbol{X}) - H(\boldsymbol{X}|\boldsymbol{Y}) = H(\boldsymbol{Y}) - H(\boldsymbol{Y}|\boldsymbol{X})$.

According to Property 1, Eq. (11) can be rewritten as (note that $H(\boldsymbol{Z}'|\boldsymbol{X}')$ is determined):

$$\mathcal{IB}_{ssl} = [H(\boldsymbol{Z}') - H(\boldsymbol{Z}'|\boldsymbol{X})] - \beta[H(\boldsymbol{Z}') - H(\boldsymbol{Z}'|\boldsymbol{X}')] = (1-\beta)H(\boldsymbol{Z}') - H(\boldsymbol{Z}'|\boldsymbol{X}). \tag{12}$$

If $1-\beta \leqslant 0$, then the optimal solution in Eq. 12 will be trivial (it is also known as *complete collapse* or *dimensional collapse* (Hua et al. (2021))). To avoid this issue, we define $\lambda = 1-\beta > 0$ and have $\mathcal{IB}_{ssl} = \lambda H(\boldsymbol{Z}') - H(\boldsymbol{Z}'|\boldsymbol{X})$. So the connection between information bottleneck principle and

entropy information is

$$\max \mathcal{IB}_{ssl} = \max \lambda H(\boldsymbol{Z}') - H(\boldsymbol{Z}'|\boldsymbol{X}) \quad (\lambda \in (0,1)). \tag{13}$$

Zhu et al. (2020) have proved that the loss in Eq. (2) is the upper bound of InfoNCE loss: $\mathcal{L}_i^{1,2} \geq \mathcal{L}_{\text{InfoNCE}}$, so minimizing $\mathcal{L}_i^{1,2}$ is equivalent to minimizing $\mathcal{L}_{\text{InfoNCE}}$'s upper bound. Because InfoNCE's principle is to pull positive pairs to be closer (i.e., to minimize $H(\boldsymbol{Z}'|\boldsymbol{X})$) and push negative pairs to be farther (i.e., to maximize $H(\boldsymbol{Z}')$), it can be viewed as a specific form of $\mathcal{IB}_{ssl}$. Therefore, we can conclude that minimizing $\mathcal{L}_i^{1,2}$ is equivalent to maximizing the lower bound of $\mathcal{IB}_{ssl}$.

Next, we discuss the effect of temperature τ for $H(\boldsymbol{Z}')$. We have the following proposition:

Proposition 2. *(Temperature τ is crucial to $H(\boldsymbol{Z}')$.) When each column of $\boldsymbol{Z}'$ is normalized, the node embedding's entropy $H(\boldsymbol{Z}')$ increases with the decrease of τ.*

We refer the reader to Appendix A.2 for detailed proof. In Proposition 2, the condition of the normalized column in $\boldsymbol{Z}'$ is actually hard to meet. Also, if always keeping a low τ and only focusing on the local separation (as shown in Figure 1 (c)), it is easy to generate the problem of embedding misplacement. So the local separation (low τ) should be arranged after the global uniformity (high τ).

4 Experiments

In this section, we verify GLATE's performance on two most important graph-related tasks: node classification and graph classification. The node classification task includes two types of learning: transductive learning and inductive learning[1]. With the help of the loss curve, global uniformity metric and local separation metric, we design some ablation studies to understand the contributions of different components in GLATE to the overall system. For the detailed dataset descriptions and experimental setups, please refer to Appendix A.4 and A.5. More experimental designs and results can be seen in Appendix A.6.

4.1 Node classification Task

4.1.1 Results of Transductive Learning

We compare GLATE with several baseline models including the supervised GCN (Kipf & Welling (2016)) model and self-supervised GCL models, as well as several variants of GLATE.

We compare six different GCL models which implement a variety of techniques. DGI (Velickovic et al. (2019)) and GMI (Peng et al. (2020)) use graph contrastive objectives based on the mutual information maximization. MVGRL (Hassani & Khasahmadi (2020)) learns node's and graph's representations from two structural views: first-order neighbors and a graph diffusion. Following SimCLR (Chen et al. (2020)), GRACE (Zhu et al. (2020)) and GCA (Zhu et al. (2021c)) employ InfoNCE loss to learn node representations for graph data. Inspired by BYOL (Grill et al. (2020)), BGRL (Thakoor et al. (2021)) imitatively uses an asymmetric encoder structure to get rid of GCL's dependence on negative pairs. The variants of GLATE include four GLATE models with fixed temperature values and a GLATE model with the hard negative sampling. The hard negative sampling strategy only collects hard negative examples (the similarity threshold $\delta = 0.8$) to construct contrastive pairs.

Table 1 shows the accuracy scores on the task of transductive node classification. We can find that GLATE outperforms all the baseline models on the three datasets with an average improvement of 2.8 percent. Unsurprisingly, an extremely high (or low) temperature value cannot make the model converge. By setting dynamic temperature values, GLATE performs better than the models with a fixed temperature value, especially on Citeseer. Other experiments about the temperature values can be seen in Section 4.3.

4.1.2 Results of Inductive Learning

[1]Our code is available at: https://github.com/anonymousICLR22/GLATE.

Table 1: Comparison with the state-of-the-art GCL algorithms on transductive node classification.

Methods	Cora	Citeseer	Pubmed
Supervised GCN (Kipf & Welling (2016))	81.50	70.30	84.90
DGI (Velickovic et al. (2019))	82.60±0.40	68.80±0.70	86.00±0.10
GMI (Peng et al. (2020))	83.00±0.30	72.40±0.10	79.90±0.20
MVGRL (Hassani & Khasahmadi (2020))	83.50±0.40	73.30±0.5	80.10±0.70
GRACE (Zhu et al. (2020))	83.30±0.40	72.10±0.50	86.70±0.10
GCA (Zhu et al. (2021c))	80.90±0.41	72.14±0.06	86.01±0.05
BGRL (Thakoor et al. (2021))	82.77±0.75	64.45±0.15	84.34±0.17
Average value	82.67	70.53	83.84
Hard negative sampling (Threshold $\delta = 0.8$)	83.14±0.43	67.11±0.56	85.28±0.14
Fixed temperature ($\tau = 10000$)[§]	31.62±0.00	41.50±0.15	62.71±0.10
Fixed temperature ($\tau = \tau_{initial} = 0.8$)	84.32±0.15	67.83±0.22	85.23±0.08
Fixed temperature ($\tau = \tau_{lower} = 0.2$)	83.10±0.28	64.02±0.54	82.13±0.06
Fixed temperature ($\tau = 0.01$)[†]	15.07±0.00	10.48±0.00	21.90±0.00
GLATE	**84.80±0.33**	**73.40±0.19**	**87.29±0.12**

[§] The value of training loss is almost not changed.

[†] The value of training loss eventually becomes "NaN".

For the task of inductive learning, we use GraphSAGE-GCN (Hamilton et al. (2017)) as the encoder of GLATE and other GCL baseline models including DGI, GMI, GRACE, and BGRL. We also compare our model with unsupervised GraphSAGE models with different aggregation functions, which are denoted as GraphSAGE-*. The method of "Raw features" only uses node features as inputs, so its accuracy score can be regarded as the lower-bound of all methods. "Supervised GraphSAGE-GCN" uses supervised learning with the encoder of GraphSAGE-GCN and its accuracy score can be seen as the upper-bound of all methods. From Table 2, we find that $\text{GLATE}_{+\text{GraphSAGE}-\text{GCN}}$ outperforms all the self-supervised baselines and is 0.89 percent ahead of the nearest competitor BGRL. Moreover, all the GCL models perform better than unsupervised GraphSAGE models, which shows the effectiveness of GCL on inductive node classification.

Methods	PPI
Raw features	42.20
Supervised GraphSAGE-GCN	96.90±0.20
GraphSAGE-GCN	46.50
GraphSAGE-mean	48.60
GraphSAGE-LSTM	48.20
GraphSAGE-pool	50.20
$\text{DGI}_{+\text{GraphSAGE}-\text{GCN}}$	63.80±0.20
$\text{GMI}_{+\text{GraphSAGE}-\text{GCN}}$	65.00±0.02
$\text{GRACE}_{+\text{GraphSAGE}-\text{GCN}}$	66.20±0.10
$\text{BGRL}_{+\text{GraphSAGE}-\text{GCN}}$	67.52±0.05
$\text{GLATE}_{+\text{GraphSAGE}-\text{GCN}}$	**68.41±0.01**

Table 2: Comparison with the state-of-the-art GCL methods on inductive node classification.

4.1.3 Results on larger graph dataset

To verify GLATE's scalability, we select a larger graph dataset ogbn-arxiv[2] to test its performance. The ogbn-arxiv dataset is a directed graph, representing the citation network between all Computer Science (CS) arXiv papers indexed by Microsoft Academic Graph (MAG). There are 169,343 nodes and 1,166,243 edges in ogbn-arxiv. The results are shown in Table (3). Apart from GLATE, all of the reported experimental results are from Thakoor et al. (2021). For GRACE and GLATE, we subsample 2048 nodes from the full graph as each anchor node's negative examples. From Table 3, we can see that GLATE outperforms the three GCL methods and is competitive with the supervised GCN model on validation and test sets.

Methods	Validation set	Test set
Supervised GCN	73.00±0.17	71.74±0.29
DGI	71.26±0.11	70.34±0.16
GRACE	72.61±0.15	71.51±0.11
BGRL	72.53±0.09	71.64±0.12
GLATE	**72.80±0.02**	**72.13±0.05**

Table 3: Comparison with the state-of-the-art GCL methods on ogbn-arxiv dataset.

[2]https://ogb.stanford.edu/docs/nodeprop/

4.2 GRAPH CLASSIFICATION TASK

Next, we evaluate GLATE's performance on the graph classification task using four graph benchmark datasets. Here we choose the following methods as baseline models: node2vec (Grover & Leskovec (2016)), sub2vec (Adhikari et al. (2018)), graph2vec (Narayanan et al. (2017)), InfoGraph (Sun et al. (2020)), and GraphCL (You et al. (2020)). All the baselines and GLATE are used to generate graph embeddings by unsupervised learning. These embeddings are as the inputs of the downstream SVM classifier. In GLATE, we use graph isomorphism network (Xu et al. (2018)) as its encoder networks, which is the same as GraphCL's. The results are shown in Table 4. Apart from GLATE, all of the reported results are from (You et al. (2020)). From Table 4, we can see that GLATE outperforms all the baselines on most datasets except on IMDB-B with small graph size (the number of average nodes is less than 20). Besides, compared with another GCL method GraphCL, the average accuracy result of GLATE is higher than it 1.2 percent.

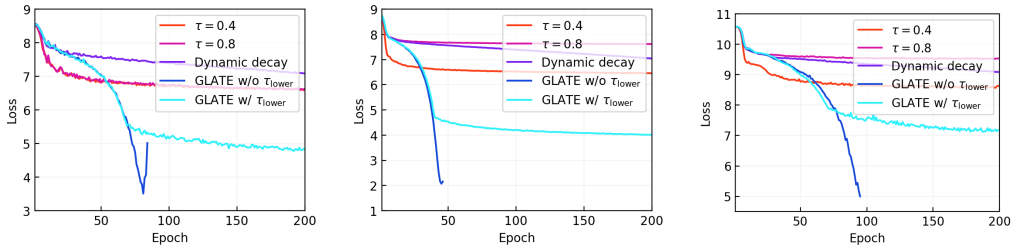
Table 4: Comparison with graph representation learning methods on graph classification task.

Methods	NCI1	PROTEINS	MUTAG	IMDB-B
node2vec	54.89±1.61	57.49±3.57	72.63±10.20	-
sub2vec	52.84±1.47	53.03±5.55	61.05±15.80	55.26±1.54
graph2vec	73.22±1.81	73.30±2.05	83.15±9.25	71.10±0.54
InfoGraph	76.20±1.06	74.44±0.31	89.01±1.13	**73.03±0.87**
GraphCL	77.87±0.41	74.39±0.45	86.80±1.34	71.14±0.44
GLATE	**78.03±0.61**	**74.68±0.21**	**90.88±0.33**	71.60±0.85

4.3 ABLATION STUDIES

4.3.1 PERFORMANCE ON THE TRAINING LOSS

In this section, we use different temperature mechanisms to visualize the effects of temperature and momentum on the training loss. Specifically, we design different training strategies to analyze the components of temperature and momentum, including 1) removing dynamic temperature estimation by fixing τ as 0.4 or 0.8; 2) replacing the momentum with an alternative dynamic solution, i.e, multiplying τ by a fixed decay factor at each epoch; 3) removing the constraint of the temperature's lower-bound (denoted as "GLATE w/o τ_{lower}" in Figure 3); 4) our proposed model GLATE (denoted as "GLATE w/ τ_{lower}" in Figure 3). We visualize the change of training loss under the above strategies for transductive learning in Figure 3.



(a) Cora (b) Citeseer (c) Pubmed

Figure 3: The curves of loss under different training strategies on Cora, Citeseer, and Pubmed. The loss value of GLATE w/o τ_{lower} eventually becomes "NaN".

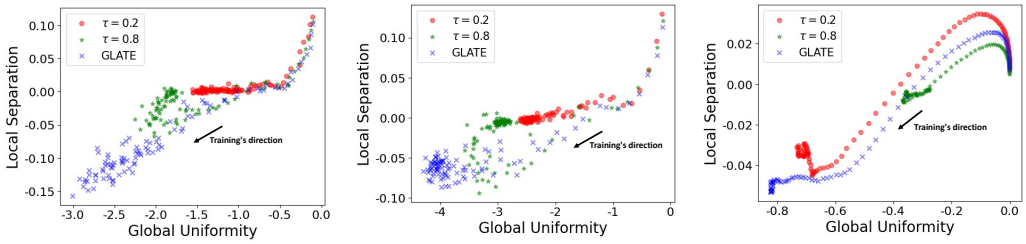
To evaluate the static and dynamic settings, we mainly focus on the performances of "$\tau = 0.4$", "$\tau = 0.8$", and "GLATE w/ τ_{lower}" in Figure 3. Although GLATE converges slower than the settings of $\tau = 0.4$ and $\tau = 0.8$, it eventually obtains a lower loss value. Compared with the static setting, the dynamic temperature estimation can explore a larger learning space, which improves the model's learning ability.

To evaluate the effect of the momentum mechanism, we mainly focus on "dynamic decay", "GLATE w/o τ_{lower}", and "GLATE w/ τ_{lower}" in Figure 3. Both GLATE w/o τ_{lower} and GLATE w/ τ_{lower}

have faster convergence speeds than the setting of dynamic decay, which indicates the effectiveness of momentum. In addition, when without the constraint of τ_{lower}, the model's convergence speed becomes very fast during training, which eventually leads to the overflow of loss ("NaN"). So we can conclude that both momentum and τ_{lower} are necessary to GLATE.

4.3.2 PERFORMANCE ON GLOBAL UNIFORMITY AND LOCAL SEPARATION

As we discussed in previous sections, global uniformity and local separation are two important metrics for measuring the quality of graph contrastive representations. We use $-D^t_{global}$ and $-D^t_{local}$ (global uniformity degree D^t_{global} and local separation degree D^t_{local} are defined in Eq. (9) and Eq. (10), respectively) as the coordinates to evaluate the quality of learned representations in GLATE. We plot the changing trend of results under these two metrics in Figure 4. The datum of each epoch is a single point. Because the coordinates mean the negative uniformity degree and negative separation degree, the ideal result lies in the left lower corner. From Figure 4, we find all of the three experimental settings (GLATE, GLATE with a fixed temperature $\tau_{initial} = 0.8$, as well as GLATE with a fixed temperature $\tau_{lower} = 0.2$) can make the results of uniformity and separation better in the training phase. Moreover, the performance of GLATE is significantly better than those of the other two methods. Note that the changing trend on Pubmed is different from those on Cora and Citeseer because we implement mini-batch training for the dataset of Pubmed and full-batch training for the other two datasets, respectively.



(a) Cora (b) Citeseer (c) Pubmed

Figure 4: The results of global uniformity and local separation on Cora, Citeseer, and Pubmed. Here the horizontal and vertical axis represent $-D^t_{global}$ and $-D^t_{local}$, respectively. The black arrow represents the training directions of all methods. The point in the left-lower corner corresponds to the best result.

5 CONCLUSION AND DISCUSSION

In this paper, we discuss the limitations of the classical InfoNCE objective for graph contrastive learning: when using a fixed temperature, the InfoNCE objective can only capture a certain state of embedding distribution and ignore the others. By theoretical analysis, we find that the temperature τ is crucial to control the representation distribution's global uniformity and local separation and thus introduce GLATE, a simple but effective algorithm for graph contrastive learning. GLATE uses dynamic temperature estimation with momentum to adjust the gradient rates so that it can cater to different distribution states. On both tasks of node classification and graph classification, we show GLATE's advantages over the state-of-the-art GCL algorithms. By some ablation studies, we illustrate the positive effects brought by the dynamic temperature setting and visualize the results under the two new metrics in the training phase.

Broader impacts. Studying the temperature's property is an interesting topic in contrastive learning, even in deep learning. Considering the dominant role of InfoNCE objectives in graph contrastive learning, we believe that deep insights into the temperature in InfoNCE objectives have an important impact on the development of graph contrastive learning. In our work, the gradient analysis and dynamic temperature estimation are not only suitable to GRACE but also any graph contrastive learning algorithm based on InfoNCE objectives. More experimental details about GLATE's generality can be seen in Appendix A.6.1.

REFERENCES

Bijaya Adhikari, Yao Zhang, Naren Ramakrishnan, and B Aditya Prakash. Sub2vec: Feature learning for subgraphs. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pp. 170–182. Springer, 2018.

Rana Ali Amjad and Bernhard C Geiger. Learning representations for neural network-based classification using the information bottleneck principle. *IEEE transactions on pattern analysis and machine intelligence*, 42(9):2225–2239, 2019.

Adrien Bardes, Jean Ponce, and Yann LeCun. Vicreg: Variance-invariance-covariance regularization for self-supervised learning. *arXiv preprint arXiv:2105.04906*, 2021.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pp. 1597–1607. PMLR, 2020.

Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15750–15758, 2021.

Yueqi Duan, Wenzhao Zheng, Xudong Lin, Jiwen Lu, and Jie Zhou. Deep adversarial metric learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2780–2789, 2018.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. Simcse: Simple contrastive learning of sentence embeddings. *arXiv preprint arXiv:2104.08821*, 2021.

Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent: A new approach to self-supervised learning. *arXiv preprint arXiv:2006.07733*, 2020.

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

William L Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pp. 1025–1035, 2017.

Kaveh Hassani and Amir Hosein Khasahmadi. Contrastive multi-view representation learning on graphs. In *International Conference on Machine Learning*, pp. 4116–4126. PMLR, 2020.

Tianyu Hua, Wenxiao Wang, Zihui Xue, Sucheng Ren, Yue Wang, and Hang Zhao. On feature decorrelation in self-supervised learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9598–9608, 2021.

Ashish Jaiswal, Ashwin Ramesh Babu, Mohammad Zaki Zadeh, Debapriya Banerjee, and Fillia Makedon. A survey on contrastive self-supervised learning. *Technologies*, 9(1):2, 2021.

Eugene Kharitonov, Morgane Rivière, Gabriel Synnaeve, Lior Wolf, Pierre-Emmanuel Mazaré, Matthijs Douze, and Emmanuel Dupoux. Data augmenting contrastive learning of speech representations in the time domain. In *2021 IEEE Spoken Language Technology Workshop (SLT)*, pp. 215–222. IEEE, 2021.

Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

Phuc H. Le-Khac, Graham Healy, and Alan F. Smeaton. Contrastive representation learning: A framework and review. *IEEE Access*, 8:193907–193934, 2020. doi: 10.1109/ACCESS.2020.3031549.

Annamalai Narayanan, Mahinthan Chandramohan, Rajasekar Venkatesan, Lihui Chen, Yang Liu, and Shantanu Jaiswal. graph2vec: Learning distributed representations of graphs. *arXiv preprint arXiv:1707.05005*, 2017.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

Erlin Pan and Zhao Kang. Multi-view contrastive graph clustering. *NeurIPS 2021*, 2021.

Zhen Peng, Wenbing Huang, Minnan Luo, Qinghua Zheng, Yu Rong, Tingyang Xu, and Junzhou Huang. Graph representation learning via graphical mutual information maximization. In *Proceedings of The Web Conference 2020*, pp. 259–270, 2020.

Ning Qian. On the momentum term in gradient descent learning algorithms. *Neural networks*, 12 (1):145–151, 1999.

Mirco Ravanelli, Jianyuan Zhong, Santiago Pascual, Pawel Swietojanski, Joao Monteiro, Jan Trmal, and Yoshua Bengio. Multi-task self-supervised learning for robust speech recognition. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6989–6993. IEEE, 2020.

Joshua David Robinson, Ching-Yao Chuang, Suvrit Sra, and Stefanie Jegelka. Contrastive learning with hard negative samples. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=CR1XOQ0UTh-.

Fan-Yun Sun, Jordan Hoffmann, Vikas Verma, and Jian Tang. Infograph: Unsupervised and semi-supervised graph-level representation learning via mutual information maximization. *ICLR 2020*, 2020.

Shantanu Thakoor, Corentin Tallec, Mohammad Gheshlaghi Azar, Rémi Munos, Petar Veličković, and Michal Valko. Bootstrapped representation learning on graphs. *ICLR 2021 Workshop*, 2021.

Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive multiview coding. *CoRR*, abs/1906.05849, 2019. URL http://arxiv.org/abs/1906.05849.

Naftali Tishby and Noga Zaslavsky. Deep learning and the information bottleneck principle. In *2015 IEEE Information Theory Workshop (ITW)*, pp. 1–5. IEEE, 2015.

Petar Velickovic, William Fedus, William L Hamilton, Pietro Liò, Yoshua Bengio, and R Devon Hjelm. Deep graph infomax. *ICLR (Poster)*, 2(3):4, 2019.

Feng Wang and Huaping Liu. Understanding the behaviour of contrastive loss. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2495–2504, 2021.

Tongzhou Wang and Phillip Isola. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *International Conference on Machine Learning*, pp. 9929–9939. PMLR, 2020.

Jiawei Wu, Xin Wang, and William Yang Wang. Self-supervised dialogue learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 3857–3867, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1375. URL https://aclanthology.org/P19-1375.

Jun Xia, Lirong Wu, Jintao Chen, Ge Wang, and Stan Z Li. Debiased graph contrastive learning. *arXiv preprint arXiv:2110.02027*, 2021.

Dongkuan Xu, Wei Cheng, Dongsheng Luo, Haifeng Chen, and Xiang Zhang. Infogcl: Information-aware graph contrastive learning. *NeurIPS 2021*, 2021.

Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *International Conference on Learning Representations*, 2018.

Zhen Yang, Ming Ding, Chang Zhou, Hongxia Yang, Jingren Zhou, and Jie Tang. Understanding negative sampling in graph representation learning. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1666–1676, 2020.

Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. Graph contrastive learning with augmentations. *Advances in Neural Information Processing Systems*, 33:5812–5823, 2020.

Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny. Barlow twins: Self-supervised learning via redundancy reduction. *arXiv preprint arXiv:2103.03230*, 2021a.

Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny. Barlow twins: Self-supervised learning via redundancy reduction. *arXiv preprint arXiv:2103.03230*, 2021b.

Ji Zhang, Kevin J Shih, Ahmed Elgammal, Andrew Tao, and Bryan Catanzaro. Graphical contrastive losses for scene graph parsing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11535–11543, 2019.

Yanqiao Zhu, Yichen Xu, Feng Yu, Qiang Liu, Shu Wu, and Liang Wang. Deep graph contrastive representation learning. In *GRL+@ICML 2020*, 2020.

Yanqiao Zhu, Yichen Xu, Hejie Cui, Carl Yang, Qiang Liu, and Shu Wu. Structure-Aware Hard Negative Mining for Heterogeneous Graph Contrastive Learning. In *KDD Workshop on Deep Learning on Graphs: Method and Applications*, 2021a.

Yanqiao Zhu, Yichen Xu, Qiang Liu, and Shu Wu. An empirical study of graph contrastive learning. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021b. URL https://openreview.net/forum?id=UuUbIYnHKO.

Yanqiao Zhu, Yichen Xu, Feng Yu, Qiang Liu, Shu Wu, and Liang Wang. Graph contrastive learning with adaptive augmentation. In *Proceedings of the Web Conference 2021*, pp. 2069–2080, 2021c.

Chengxu Zhuang, Alex Lin Zhai, and Daniel Yamins. Local aggregation for unsupervised learning of visual embeddings. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 6002–6012, 2019.

A APPENDIX

A.1 PROOF OF PROPOSITION 1

Proof. Using Eq. (4) and (5), we derive:

$$\frac{\nabla_{\mathrm{S}(i',j')}\mathcal{L}_i^{1,2}}{\nabla_{\mathrm{S}(i',j'')}\mathcal{L}_i^{1,2}} = \frac{\mathrm{E}_{i',j'}}{\mathrm{E}_{i',j''}} = \exp(\frac{1}{\tau}\cdot(\mathrm{S}(i',j') - \mathrm{S}(i',j''))). \tag{14}$$

Here $1/\tau$ is a scaling coefficient of $\mathrm{S}(i',j') - \mathrm{S}(i',j'')$. If $\tau \to \infty$, the whole exponential term tends to 1, so we conclude $\nabla_{\mathrm{S}(i',j')}\mathcal{L}_i^{1,2} \to \nabla_{\mathrm{S}(i',j'')}\mathcal{L}_i^{1,2}$. If τ decreases, the difference between $\nabla_{\mathrm{S}(i',j')}\mathcal{L}_i^{1,2}$ and $\nabla_{\mathrm{S}(i',j'')}\mathcal{L}_i^{1,2}$ will be larger. □

A.2 PROOF OF PROPOSITION 2

Property 2. *If $\boldsymbol{Z}$ obeys a D-dimensional Gaussian distribution: $\boldsymbol{Z} \sim \mathcal{N}(\boldsymbol{\mu_Z}, \boldsymbol{\Sigma_Z})$ ($\boldsymbol{\mu_Z} \in \mathbb{R}^{1\times D}$,, $\boldsymbol{\Sigma_Z} \in \mathbb{R}^{D\times D}$), then $H(\boldsymbol{Z}) = \frac{D}{2}(\ln 2\pi + 1) + \frac{1}{2}\ln(\det\boldsymbol{\Sigma_Z})$.*

Proof. If τ decreases, then the learning algorithm will punish more on hard examples than easy examples. So it makes the similarity between hard negative examples lower. Besides, the easy negative examples are always dissimilar. So we have $\boldsymbol{Z}'_i\boldsymbol{Z}'^{\mathrm{T}}_j \to 0$ ($i \neq j$) and $\boldsymbol{Z}'_i\boldsymbol{Z}'^{\mathrm{T}}_i \to 1$. Due to $n \gg d$ (dense node representations), the (normalized) columns of $\boldsymbol{Z}'$ tend to be linearly independent and $\boldsymbol{Z}'^{\mathrm{T}}\boldsymbol{Z}' \to \boldsymbol{I}_D$ ($\boldsymbol{I}_D$ is an identity matrix, $\boldsymbol{I}_D \in \mathbb{R}^{D\times D}$). Due to $(\boldsymbol{\Sigma}_{\boldsymbol{Z}'})_{i,j} = \mathrm{Cov}[\boldsymbol{Z}'_i, \boldsymbol{Z}'_j]$, we can rewrite the determinant of $\boldsymbol{\Sigma}_{\boldsymbol{Z}'}$ as:

$$\det(\boldsymbol{\Sigma}_{\boldsymbol{Z}'}) = \det(\boldsymbol{Z}'^{\mathrm{T}}\boldsymbol{Z}') = \prod_{i=1}^{D}\lambda_i = \exp(\sum_{i=1}^{D}\log\lambda_i) \tag{15}$$

where $\lambda_1, \lambda_1, \cdots \lambda_D$ are $\boldsymbol{\Sigma}_{\boldsymbol{Z}'}$'s eigenvalues. According to Jensen inequality, as $\log(\cdot)$ is a concave function, we have $\frac{1}{D}\sum_{i=1}^{D}\log\lambda_i \leqslant \log\frac{\sum_{i=1}^{D}\lambda_i}{D} = 0$ (note that $\sum_{i=1}^{D}\lambda_i = \mathrm{trace}(\boldsymbol{\Sigma}_{\boldsymbol{Z}'}) = D$). So $\det(\boldsymbol{\Sigma}_{\boldsymbol{Z}'}) \leqslant \exp(0) = 1$. Note that if $\boldsymbol{Z}'^{\mathrm{T}}\boldsymbol{Z}' \to \boldsymbol{I}_D$, then $\det(\boldsymbol{\Sigma}_{\boldsymbol{Z}'}) \to 1$ (i.e., its upper bound) and $H(\boldsymbol{Z}')$ tends to be its global maximum (according to Property 2). □

A.3 THEORETICAL DIFFERENCE WITH THE CURRENT WORK

We investigate another work (https://arxiv.org/abs/2106.05819) using the analysis of the graph information bottleneck principle. Here we give some explanation about the difference between our theoretical basis and theirs. The main difference is **the negative terms in the information bottleneck** when ignoring the impact of the coefficients. We analyze the information bottleneck principles of https://arxiv.org/abs/2106.05819 (denoted as $\mathcal{IB}^1_{ssl}$) and our paper (denoted as $\mathcal{IB}^2_{ssl}$) in detail and give the proof below. For the task of learning node representations Z' and Z'', Equation (5) in https://arxiv.org/abs/2106.05819 can be represented as: $\mathcal{IB}^1_{ssl} = I(Z', Z'') = H(Z') - H(Z'|Z'')$. As $H(Z'|Z'') = H(Z', Z'') - H(Z'')$, then

$$\mathcal{IB}^1_{ssl} = H(Z') + H(Z'') - H(Z', Z''). \tag{16}$$

While, in our paper (Equation (11)): $\mathcal{IB}^2_{ssl} = \lambda H(Z') - H(Z'|X) \quad (\lambda \in (0,1))$. Because Z'' has the same property as Z', we have:

$$\mathcal{IB}^2_{ssl} = \frac{\lambda}{2} * (H(Z') + H(Z'')) - \frac{1}{2} * (H(Z'|X) + H(Z''|X)). \tag{17}$$

Regardless of the effect of coefficients, an intuitive understanding to Equation (16) and (17) is: the common goal of $\mathcal{IB}^1_{ssl}$ and $\mathcal{IB}^2_{ssl}$ is to learn the most informative part of raw data X and remove the redundant information from it. Although $\mathcal{IB}^1_{ssl}$ and $\mathcal{IB}^2_{ssl}$ have consistent optimization objectives,

they have a minor difference at the minimization term: one is to minimize $H(Z', Z'')$ and the other is to minimize $H(Z'|X) + H(Z''|X)$. It implies that $\mathcal{IB}^1_{ssl}$ replies more on the choice of predefined graph augmentation strategies, while $\mathcal{IB}^2_{ssl}$ is not limited to graph augmentation strategies. In a summary, if we aim to select the most suitable graph augmentation strategies, $\mathcal{IB}^1_{ssl}$ is more helpful; but if we want to avoid the bad impact of an improper choice of graph augmentation, $\mathcal{IB}^2_{ssl}$ is better than $\mathcal{IB}^1_{ssl}$.

A.4 DATASETS

For the node classification task, we utilize some commonly-used benchmarks in the experiments, including four citation networks (Cora, Citeseer, Pubmed, and ogbn-arxiv) for the transductive single-label node classification task, as well as the Protein-Protein Interaction dataset (PPI) for the inductive multi-label node classification task. In the citation networks, each node represents a paper whose features are topic types (Cora, Citeseer, and Pubmed) or are a vector obtained by averaging the embeddings of words in its title and abstract (ogbn-arxiv), and each edge represents the citation relationship between papers. In the PPI dataset, there are 24 graphs. Each node in the graphs of PPI denotes a protein with biological features, and each edge denotes the interaction between proteins. The details of the datasets are in Table 5.

For the graph classification task, we use four datasets including NCI1 (small molecules), PROTEINS (small molecules), MUTAG (bioinformatics), and IMDB-B (social networks). Their detailed information is shown in Table 6.

Table 5: Details of datasets for node classification task.

Datasets	# Nodes	# Edges	# Features	# Classes
Cora	2,708	10,556	1,433	7
Citeseer	3,327	9,228	3,703	6
Pubmed	19,717	88,651	500	3
ogbn-arxiv	169,343	1,166,243	128	40
PPI (24 graphs)	56,944	818,716	50	121(multi-label)

Table 6: Details of datasets for graph classification task.

Datasets	# Graphs	# Avg. Nodes	# Avg. Edges	# Classes
NCI1	4110	29.87	32.30	2
PROTEINS	1113	39.06	72.82	2
MUTAG	188	17.93	19.79	2
IMDB-B	1000	19.77	96.53	2

A.5 EXPERIMENTAL SETUPS

Under the contrastive setting, we use data augmentation strategies, i.e., edge removing and node feature masking, to generate the new graphs $\tilde{\mathcal{G}}_1 = (\boldsymbol{A}', \boldsymbol{X}')$ and $\tilde{\mathcal{G}}_2 = (\boldsymbol{A}'', \boldsymbol{X}'')$. For the transductive task, we use a classical two-layer GCN model (Kipf & Welling (2016)), whose message passing form is Eq. (1), as the encoder of GLATE. For the inductive task, the encoder model is a two-layer GraphSAGE model (Hamilton et al. (2017)). Its message passing process can be formalized as:

$$\boldsymbol{Z}_i^k = \sigma\left(\boldsymbol{W} \cdot \mathrm{CONCAT}(\boldsymbol{Z}_i^{k-1}, \mathrm{AGGREGATE}(\{\boldsymbol{Z}_j^{k-1}, j \in \mathcal{N}(i)\}))\right) \tag{18}$$

where $\boldsymbol{Z}_i^0 = \boldsymbol{X}_i$, σ denotes the activation function, $\mathcal{N}(i)$ denotes the neighbors of node i, the aggregation function $\mathrm{AGGREGATE}(\cdot)$ can employ arbitrary aggregators such as GCN, Mean, LSTM or Pool. In experiments, we consistently choose GCN as the aggregator for all methods.

To evaluate the performance of the GLATE, we use a logistic regression model with a one-vs-rest classifier framework for the downstream classification tasks. The size of both the hidden layer representations and the node representations are set to 256 for the transductive task and 512 for the inductive task. The temperature's initial value $\tau_{initial}$ is set as 0.8 and τ will decrease every 20 epochs until $\tau_{lower} = 0.2$ in all the experiments of GLATE. The model runs 200 epochs and 100 epochs for the transductive and inductive tasks, respectively.

A.6 MORE EXPERIMENTS

A.6.1 THE GENERALITY OF DYNAMIC TEMPERATURE ESTIMATION

Considering the dominant role of InfoNCE objectives in GCL, we believe that deep insights into the temperature in InfoNCE objectives will have an important impact on the development of GCL. To validate our method's generality, we conduct experiments for another GCL method GraphCL using the InfoNCE objective on four graph datasets. The results of the graph classification task are shown in Table 7 ("Improved GraphCL" denotes a new GraphCL method using dynamic temperature estimation). We can see that the dynamic temperature estimation indeed brings positive effects to GraphCL.

Table 7: Results on the graph classification task. "Improved GraphCL" is a new GraphCL method using dynamic temperature estimation.

Methods	NCI1	PROTEINS	MUTAG	IMDB-B
node2vec	54.89±1.61	57.49±3.57	72.63±10.20	-
sub2vec	52.84±1.47	53.03±5.55	61.05±15.80	55.26±1.54
graph2vec	73.22±1.81	73.30±2.05	83.15±9.25	71.10±0.54
InfoGraph	76.20±1.06	74.44±0.31	89.01±1.13	**73.03±0.87**
GraphCL	77.87±0.41	74.39±0.45	86.80±1.34	71.14±0.44
Improved GraphCL	**78.22±0.59**	**74.58±0.66**	**89.27±0.15**	71.20±0.35

A.6.2 TIME COMPLEXITY ANALYSIS

Because τ decreases every 20 epochs, compared with GRACE, the extra time cost for computing a new τ's value is minor. When computing the full negatives, GLATE's time complexity is quadratic in the size of the input graph per epoch (O(N^2)). When computing the sampled negatives (we denote the number of sampled negatives as k), GLATE's time complexity is linear in the size of the input graph per epoch (O(kN)).

We also record the total running time of the two most representative baselines and GLATE in Table 8. We run all experiments on NVIDIA GeForce GTX 1080 Ti. As GLATE relies on **less training epochs** to achieve competitive results than GRACE and BGRL, it spends less training time than the other baselines.

Table 8: Time costs of GRACE, BGRL, and GLATE.

Methods	Cora	Citeseer	Pubmed
GRACE	6.7 seconds	11.2 seconds	207.0 seconds
BGRL	6.3 seconds	8.1 seconds	149.7 seconds
GLATE	**3.9 seconds**	**5.9 seconds**	**28.6 seconds**

A.6.3 EXPERIMENTAL RESULTS ON THE OTHER GRAPH DATASETS.

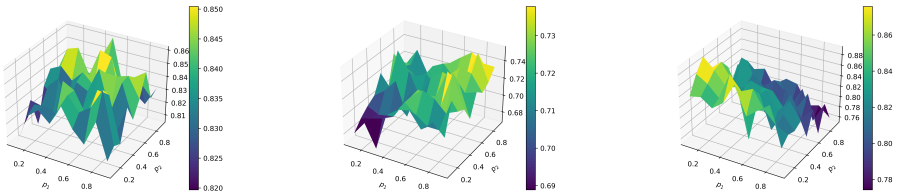
Table 9: Results on the datasets of Amazon and Coauthor.

Methods	Amazon-Computers	Amazon-Photo	Coauthor-CS	Coauthor-Physics
DGI	83.95±0.47	91.61±0.22	92.15±0.63	94.51±0.52
GMI	82.21±0.31	90.68±0.17	OOM	OOM
MVGRL	87.52±0.11	91.74±0.07	92.11±0.12	95.33±0.03
GRACE	87.46±0.22	92.15±0.24	92.93±0.01	95.26±0.02
GCA	88.94±0.15	92.53±0.16	93.10±0.01	95.73±0.03
BGRL	89.68±0.31	92.87±0.27	**93.21±0.18**	95.56±0.12
GLATE	**89.96±0.20**	**93.12±0.14**	93.01±0.27	**95.97±0.08**

We evaluate GLATE's performance on the datasets of Amazon-Computers, Amazon-Photo, Coauthor-CS, and Coauthor-Physics, which are also used in GCA. The graphs in these datasets are larger than those in Cora and Citeseer. For example, there are totally 13,752 nodes and 245,861 edges in the graph of Amazon-Computers; there are totally 34,493 nodes and 247,962 edges in the graph of Coauthor-Physics. The experimental results are shown in Table 9. We can see that GLATE outperforms GRACE and is competitive with BGRL on the four datasets.

A.6.4 SENSITIVITY ANALYSIS

We analyze the effects of different hyperparameters p_1 and p_2 (p_1, p_2 are the probability of removing edges and masking features) for GLATE's performance. We plot the accuracy score's results on the datasets of Cora, Citeseer, and Pubmed in Figure 5. Although GLATE's best results occur at different locations (e.g., $p_1 = p_2 = 0.8$ for Citeseer, $p_1 = 0.1$ and $p_2 = 0.4$ for Pubmed) in different datasets, we note that the average of the difference between peak and valley performance values is less than 9 percent. It indicates that GLATE is not very sensitive to the two hyperparameters. Therefore, GLATE is robust to p_1 and p_2.



(a) Cora (b) Citeseer (c) Pubmed

Figure 5: GLATE's performance with different values of hyperparameters p_1 and p_2.

Table 10: GLATE's performances with different temperatures under different contrastive view.

	$p_1 = 0.9$	$p_1 = 0.8$	$p_1 = 0.6$	$p_1 = 0.4$	$p_1 = 0.2$
$p_2 = 0.9$	82.26 (1.0)	83.35 (1.0)	84.27 (1.0)	83.58 (1.0)	83.19 (1.0)
	83.35 (0.8)	**84.02 (0.8)**	82.75 (0.8)	84.46 (0.8)	83.95 (0.8)
	84.18 (0.6)	82.47 (0.6)	**84.66 (0.6)**	80.35 (0.6)	83.26 (0.6)
	83.51 (0.4)	83.83 (0.4)	83.12 (0.4)	82.40 (0.4)	**84.92 (0.4)**
	82.50 (0.2)	83.88 (0.2)	82.89 (0.2)	**85.33 (0.2)**	84.06 (0.2)
$p_2 = 0.8$	**84.52 (1.0)**	83.93 (1.0)	**84.92 (1.0)**	81.85 (1.0)	83.95 (1.0)
	82.73 (0.8)	**86.07 (0.8)**	82.03 (0.8)	81.90 (0.8)	83.37 (0.8)
	83.46 (0.6)	83.79 (0.6)	81.20 (0.6)	82.68 (0.6)	82.54 (0.6)
	82.20 (0.4)	82.75 (0.4)	82.20 (0.4)	83.69 (0.4)	83.51 (0.4)
	83.90 (0.2)	84.32 (0.2)	82.73 (0.2)	**84.55 (0.2)**	**84.39 (0.2)**
$p_2 = 0.6$	83.39 (1.0)	83.74 (1.0)	84.11 (1.0)	82.52 (1.0)	81.53 (1.0)
	84.36 (0.8)	82.52 (0.8)	82.40 (0.8)	83.00 (0.8)	82.73 (0.8)
	83.97 (0.6)	83.33 (0.6)	84.13 (0.6)	82.84 (0.6)	82.29 (0.6)
	83.69 (0.4)	**85.40 (0.4)**	**84.52 (0.4)**	**84.73 (0.4)**	84.18 (0.4)
	83.19 (0.2)	83.99 (0.2)	83.00 (0.2)	82.66 (0.2)	**84.39 (0.2)**
$p_2 = 0.4$	83.33 (1.0)	**84.89 (1.0)**	**85.06 (1.0)**	84.52 (1.0)	82.66 (1.0)
	82.63 (0.8)	83.74 (0.8)	84.27 (0.8)	84.11 (0.8)	83.72 (0.8)
	83.51 (0.6)	84.52 (0.6)	82.52 (0.6)	84.48 (0.6)	84.78 (0.6)
	84.41 (0.4)	84.29 (0.4)	81.41 (0.4)	**84.70 (0.4)**	**85.18 (0.4)**
	82.84 (0.2)	84.78 (0.2)	81.76 (0.2)	84.09 (0.2)	83.42 (0.2)
$p_2 = 0.2$	83.81 (1.0)	80.51 (1.0)	83.38 (1.0)	82.33 (1.0)	83.16 (1.0)
	82.80 (0.8)	84.20 (0.8)	**84.80 (0.8)**	83.69 (0.8)	82.38 (0.8)
	82.75 (0.6)	82.86 (0.6)	84.52 (0.6)	83.97 (0.6)	82.66 (0.6)
	83.76 (0.4)	**85.40 (0.4)**	82.39 (0.4)	83.19 (0.4)	**84.27 (0.4)**
	84.46 (0.2)	82.31 (0.2)	81.72 (0.2)	**85.08 (0.2)**	83.53 (0.2)

Moreover, we evaluate GLATE's performance using different selections of $\tau_{initial}$ under different initial contrastive views. The results on Cora are shown in Table 10 (p_1 and p_2 denote the rate of removing edges and masking features, respectively; the number in square brackets is the selected value of $\tau_{initial}$). We can see that using high $\tau_{initial}$ (such as 0.6, 0.8, and 1.0) is better when p_1 and p_2 are large (top left corner), while using low $\tau_{initial}$ is more suitable to the opposite situation (bottom right corner). As the lower-bound of τ is fixed as 0.2, we think it is reasonable because a wider adjustable range of temperatures is helpful to GLATE to learn from more noisy data.