Graph Contrastive Learning with Adaptive Augmentation

Yanqiao Zhu\textsuperscript{1,2,*}, Yichen Xu\textsuperscript{3,*}, Feng Yu\textsuperscript{4}, Qiang Liu\textsuperscript{1,2}, Shu Wu\textsuperscript{1,2,†}, and Liang Wang\textsuperscript{1,2}
\textsuperscript{1}Center for Research on Intelligent Perception and Computing, Institute of Automation, Chinese Academy of Sciences
\textsuperscript{2}School of Artificial Intelligence, University of Chinese Academy of Sciences
\textsuperscript{3}School of Computer Science, Beijing University of Posts and Telecommunications \quad \textsuperscript{4}Alibaba Group
yanqiao.zhu@cripac.ia.ac.cn, linyinxs@bupt.edu.cn
yf271406@alibaba-inc.com, {qiang.liu, shu.wu, wangliang}@nlpr.ia.ac.cn

ABSTRACT

Recently, contrastive learning (CL) has emerged as a successful method for unsupervised graph representation learning. Most graph CL methods first perform stochastic augmentation on the input graph to obtain two graph views and maximize the agreement of representations in the two views. Despite the prosperous development of graph CL methods, the design of graph augmentation schemes—a crucial component in CL—remains rarely explored. We argue that the data augmentation schemes should preserve intrinsic structures and attributes of graphs, which will force the model to learn representations that are insensitive to perturbation on unimportant nodes and edges. However, most existing methods adopt uniform data augmentation schemes, like uniformly dropping edges and uniformly shuffling features, leading to suboptimal performance. In this paper, we propose a novel graph contrastive representation learning method with adaptive augmentation that incorporates various priors for topological and semantic aspects of the graph. Specifically, on the topology level, we design augmentation schemes based on node centrality measures to highlight important connective structures. On the node attribute level, we corrupt node features by adding more noise to unimportant node features, to enforce the model to recognize underlying semantic information. We perform extensive experiments of node classification on a variety of real-world datasets. Experimental results demonstrate that our proposed method consistently outperforms existing state-of-the-art baselines and even surpasses some supervised counterparts, which validates the effectiveness of the proposed contrastive framework with adaptive augmentation.

CCS CONCEPTS
\begin{itemize}
\end{itemize}

KEYWORDS
Contrastive learning, graph representation learning, unsupervised learning, self-supervised learning

\textsuperscript{*}The first two authors made equal contribution to this work.
\textsuperscript{†}To whom correspondence should be addressed.

1 INTRODUCTION

Over the past few years, graph representation learning has emerged as a powerful strategy for analyzing graph-structured data. Graph representation learning using Graph Neural Networks (GNN) has received considerable attention, which aims to transform nodes to low-dimensional dense embeddings that preserve graph attribute and structural features. However, existing GNN models are mostly established in a supervised manner [20, 23, 44], which require abundant labeled nodes for training. Recently, Contrastive Learning (CL), as revitalization of the classical Information Maximization (InfoMax) principle [26], achieves great success in many fields, e.g., visual representation learning [1, 17, 41] and natural language processing [4, 28]. These CL methods seek to maximize the Mutual Information (MI) between the input (i.e. images) and its representations (i.e. image embeddings) by contrasting positive pairs with negative-sampled counterparts.

Inspired by previous CL methods, Deep Graph InfoMax (DGI) [45] marries the power of GNN into InfoMax-based methods. DGI firstly augments the original graph by simply shuffling node features. Then, a contrastive objective is proposed to maximize the MI between node embeddings and a global summary embedding. Following DGI, GMI [32] proposes two contrastive objectives to directly measure MI between input and representations of nodes and edges respectively, without explicit data augmentation. Moreover, to supplement the input graph with more global information, MVGRL [16] proposes to augment the input graph via graph diffusion kernels [24]. Then, it constructs graph views by uniformly sampling subgraphs and learns to contrast node representations to global embeddings across the two views.

Despite the prosperous development of graph CL methods, data augmentation schemes, proved to be a critical component for visual representation learning [47], remain rarely explored in existing literature. Unlike abundant data transformation techniques available for images and texts, graph augmentation schemes are non-trivial to define in CL methods, since graphs are far more complex due to the non-Euclidean property. We argue that the augmentation schemes used in the aforementioned methods suffer from two drawbacks. At first, simple data augmentation in either the structural domain or the attribute domain, such as feature shifting in DGI [45], is not sufficient for generating diverse neighborhoods (i.e. local context) and their impact on the global graph embedding. Moreover, the above methods use simple augmentation schemes, like uniformly dropping edges and shuffling features, which are not enough to capture the inductive bias of the model. In this paper, we propose a novel graph contrastive learning method with adaptive augmentation that incorporates various priors for topological and semantic aspects of the graph. Specifically, on the topology level, we design augmentation schemes based on node centrality measures to highlight important connective structures. On the node attribute level, we corrupt node features by adding more noise to unimportant node features, to enforce the model to recognize underlying semantic information.
contexts) for nodes, especially when node features are sparse, leading to difficulty in optimizing the contrastive objective. Secondly, previous work ignores the discrepancy in the impact of nodes and edges when performing data augmentation. For example, if we construct graph views by uniformly dropping edges, removing some influential edges will deteriorate the embedding quality. As the representations learned by the contrastive objective tend to be invariant to corruption induced by the data augmentation scheme [49], the data augmentation strategies should be adaptive to the input graph to reflect its intrinsic patterns. Again, taking the edge removing scheme as an example, we can give larger probabilities to unimportant edges and lower probabilities to important ones, when randomly removing the edges. Then, this scheme is able to guide the model to ignore the introduced noise on unimportant edges and thus learn important patterns underneath the input graph.

To this end, we propose a novel contrastive framework for unsupervised graph representation learning, as shown in Figure 1, which we refer to as Graph Contrastive learning with Adaptive augmentation, GCA for brevity. In GCA, we first generate two correlated graph views by performing stochastic corruption on the input. Then, we train the model using a contrastive loss to maximize the agreement between node embeddings in these two views. Specifically, we propose a joint, adaptive data augmentation scheme at both topology and node attribute levels, namely removing edges and masking features, to provide diverse contexts for nodes in different views, so as to boost optimization of the contrastive objective. Moreover, we identify important edges and feature dimensions via centrality measures. Then, on the topology level, we adaptively drop edges by giving large removal probabilities to unimportant edges to highlight important connective structures. On the node attribute level, we corrupt attributes by adding more noise to unimportant feature dimensions, to enforce the model to recognize underlying semantic information.

The core contribution of this paper is two-fold:

- Firstly, we propose a general contrastive framework for unsupervised graph representation learning with strong, adaptive data augmentation. The proposed GCA framework jointly performs data augmentation on both topology and attribute levels that are adaptive to the graph structure and attributes, which encourages the model to learn important features from both aspects.

- Secondly, we conduct comprehensive empirical studies using five public benchmark datasets on node classification under the commonly-used linear evaluation protocol. GCA consistently outperforms existing methods and our unsupervised method even surpasses its supervised counterparts on several transductive tasks.

To make the results of this work reproducible, we make all the code publicly available at https://github.com/CRIPAC-DIG/GCA.

The remaining of the paper includes the following sections. We briefly review related work in Section 2. In Section 3, we present the proposed GCA model in detail. The results of the experiments are analyzed in Section 4. Finally, we conclude the paper in Section 5. For readers of interest, additional configurations of experiments and details of proofs are provided in Appendix A and B, respectively.

2 RELATED WORK

In this section, we briefly review prior work on contrastive representation learning. Then, we review graph representation learning methods. At last, we provide a summary of comparisons between the proposed method and its related work.

2.1 Contrastive Representation Learning

Being popular in self-supervised representation learning, contrastive methods aim to learn discriminative representations by contrasting positive and negative samples. For visual data, negative samples can be generated using a multiple-stage augmentation pipeline [1, 3, 6], consisting of color jitter, random flip, cropping, resizing, rotation [8], color distortion [25], etc. Existing work [17, 41, 48] employs a memory bank for storing negative samples. Other work [1, 3, 50] explores in-batch negative samples. For an image patch as the anchor, these methods usually find a global summary vector [1, 19] or patches in neighboring views [18, 43] as the positive sample, and
contrast them with negative-sampled counterparts, such as patches of other images within the same batch [19].

Theoretical analysis sheds light on the reasons behind their success [35]. Objectives used in these methods can be seen as maximizing a lower bound of MI between input features and their representations [26]. However, recent work [42] reveals that downstream performance in evaluating the quality of representations may strongly depend on the bias that is encoded not only in the convolutional architectures but also in the specific estimator of the InfoMax objective.

2.2 Graph Representation Learning

Many traditional methods on unsupervised graph representation learning inherently follow the contrastive paradigm [11, 14, 22, 34]. Prior work on unsupervised graph representation learning focuses on local contrastive patterns, which forces neighboring nodes to have similar embeddings. For example, in the pioneering work DeepWalk [34] and node2vec [11], nodes appearing in the same random walk are considered as positive samples. Moreover, to model probabilities of node co-occurrence pairs, many studies resort to Noise-Contrastive Estimation (NCE) [12]. However, these random-walk-based methods are proved to be equivalent to factorizing some forms of graph proximity (e.g., multiplication of the adjacent matrix to model high-order connection) [37] and thus tend to overly emphasize on the encoded structural information. Also, these methods are known to be error-prone with inappropriate hyperparameter tuning [11, 34].

Recent work on Graph Neural Networks (GNNs) employs more powerful graph convolutional encoders over conventional methods. Among them, considerable literature has grown up around the theme of supervised GNN [20, 23, 44, 46], which requires labeled datasets that may not be accessible in real-world applications. Along the other line of development, unsupervised GNNs receive little attention. Representative methods include GraphSAGE [15], which incorporates DeepWalk-like objectives. Recent work DGI [45] marries the power of GNN and CL, which focuses on maximizing MI between global graph-level and local node-level embeddings. Specifically, to implement the InfoMax objective, DGI requires an injective readout function to produce the global graph-level embedding. However, it is too restrictive to fulfill the injective property of the graph readout function, such that the graph embedding may be deteriorated. In contrast to DGI, our preliminary work [52] proposes to not rely on an explicit graph embedding, but rather focuses on maximizing the agreement of node embeddings across two corrupted views of the graph.

Following DGI, GMI [32] employs two discriminators to directly measure MI between input and representations of both nodes and edges without data augmentation; MVGRL [16] proposes to learn both node- and graph-level representations by performing node diffusion and contrasting node representations to augmented graph summary representations. Moreover, GCC [36] proposes a pretraining framework based on CL. It proposes to construct multiple graph views by sampling subgraphs based on random walks and then learn model weights with several feature engineering schemes. However, these methods do not explicitly consider adaptive graph augmentation at both structural and attribute levels, leading to suboptimal performance. Unlike these work, the adaptive data augmentation at both topology and attribute levels used in our GCA is able to preserve important patterns underneath the graph through stochastic perturbation.

Comparisons with related graph CL methods. In summary, we provide a brief comparison between the proposed GCA and other state-of-the-art graph contrastive representation learning methods, including DGI [45], GMI [32], and MVGRL [16] in Table 1, where the last two columns denote data augmentation strategies at topology and attribute levels respectively. It is seen that the proposed GCA method simplifies previous node–global contrastive scheme by defining contrastive objective at the node level. Most importantly, GCA is the only one that proposes adaptive data augmentation on both topology and attribute levels.

<table>
<thead>
<tr>
<th>Method</th>
<th>Contrastive objective</th>
<th>Topology</th>
<th>Attribute</th>
</tr>
</thead>
<tbody>
<tr>
<td>DGI</td>
<td>Node–global</td>
<td>Uniform</td>
<td>—</td>
</tr>
<tr>
<td>GMI</td>
<td>Node–node</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>MVGRL</td>
<td>Node–global</td>
<td>Uniform</td>
<td>—</td>
</tr>
<tr>
<td><strong>GCA</strong></td>
<td>Node–node</td>
<td>Adaptive</td>
<td>Adaptive</td>
</tr>
</tbody>
</table>

3 THE PROPOSED METHOD

In the following section, we present GCA in detail, starting with the overall contrastive learning framework, followed by the proposed adaptive graph augmentation schemes. Finally, we provide theoretical justification behind our method.

3.1 Preliminaries

Let $G = (V, E)$ denote a graph, where $V = \{v_1, v_2, \ldots, v_N\}$, $E \subseteq V \times V$ represent the node set and the edge set respectively. We denote the feature matrix and the adjacency matrix as $X \in \mathbb{R}^{N \times F}$ and $A \in \{0, 1\}^{N \times N}$, where $x_i \in \mathbb{R}^{F}$ is the feature of $v_i$, and $A_{ij} = 1$ iff $(v_i, v_j) \in E$. There is no given class information of nodes in $G$ during training in the unsupervised setting. Our objective is to learn a GNN encoder $f(X, A) \in \mathbb{R}^{N \times F}$ receiving the graph features and structure as input, that produces node embeddings in low dimensionality, i.e. $F' \ll F$. We denote $h_i = f(X, A)$ as the learned representations of nodes, where $h_i$ is the embedding of node $v_i$. These representations can be used in downstream tasks, such as node classification and community detection.

3.2 The Contrastive Learning Framework

The proposed GCA framework follows the common graph CL paradigm where the model seeks to maximize the agreement of representations between different views [16, 52]. To be specific, we first generate two graph views by performing stochastic graph augmentation on the input. Then, we employ a contrastive objective that enforces the encoded embeddings of each node in the two different views to agree with each other and can be discriminated from embeddings of other nodes.
In our GCA model, at each iteration, we sample two stochastic augmentation functions \( t \sim \mathcal{T} \) and \( t' \sim \mathcal{T} \), where \( \mathcal{T} \) is the set of all possible augmentation functions. Then, we generate two graph views, denoted as \( \tilde{G}_1 = t(G) \) and \( \tilde{G}_2 = t'(G) \), and denote node embeddings in the two generated views as \( U = f(\tilde{X}_1, A_1) \) and \( V = f(\tilde{X}_2, A_2) \), where \( \tilde{X}_x \) and \( A_x \) are the feature matrices and adjacent matrices of the views.

After that, we employ a contrastive objective, i.e. a discriminator, that distinguishes the embeddings of the same node in these two different views from other node embeddings. For any node \( v_i \), its embedding generated in one view, \( \tilde{u}_i \), is treated as the anchor, the embedding of it generated in the other view, \( u_i \), forms the positive sample, and the other embeddings in the two views are naturally regarded as negative samples. Mirroring the InfoNCE objective [43] in our multi-view graph CL setting, we define the pairwise objective for each positive pair \((u_i, v_i)\) as

\[
\ell(u_i, v_i) = \log \frac{e^{\theta(u_i, v_i)/\tau}}{\sum_{k \neq i} e^{\theta(u_i, v_k)/\tau} + \sum_{k \neq i} e^{\theta(u_k, v_i)/\tau}},
\]

where \( \tau \) is a temperature parameter. We define the critic \( \theta(u, v) = s(g(u), g(v)) \), where \( s(\cdot, \cdot) \) is the cosine similarity and \( g(\cdot) \) is a non-linear projection to enhance the expression power of the critic function [3, 42]. The projection function \( g \) in our method is implemented with a two-layer perceptron model.

Given a positive pair, we naturally define negative samples as all other nodes in the two views. Therefore, negative samples come from two sources, that are inter-view and intra-view nodes, corresponding to the second and the third term in the denominator in Eq. (1), respectively. Since two views are symmetric, the loss for another view is defined similarly for \((v_i, u_i)\). The overall objective to be maximized is then defined as the average over all positive pairs, formally given by

\[
\mathcal{J} = \frac{1}{2N} \sum_{i=1}^{N} [\ell(u_i, v_i) + \ell(v_i, u_i)].
\]

To sum up, at each training epoch, GCA first draws two data augmentation functions \( t \) and \( t' \), and then generates two graph views \( \tilde{G}_1 = t(G) \) and \( \tilde{G}_2 = t'(G) \) of graph \( G \) accordingly. Then, we obtain node representations \( U \) and \( V \) of \( \tilde{G}_1 \) and \( \tilde{G}_2 \) using a GNN encoder \( f \). Finally, the parameters are updated by maximizing the objective in Eq. (2). The training algorithm is summarized in Algorithm 1.

### 3.3 Adaptive Graph Augmentation

In essence, CL methods that maximize agreement between views seek to learn representations that are invariant to perturbation introduced by the augmentation schemes [49]. In the GCA model, we propose to design augmentation schemes that tend to keep important structures and attributes unchanged, while perturbing possibly unimportant links and features. Specifically, we corrupt the input graph by randomly removing edges and masking node features in the graph, and the removing or masking probabilities are skewed for unimportant edges or features, that is, higher for unimportant edges or features, and lower for important ones. From an amortized perspective, we emphasize important structures and attributes over randomly corrupted views, which will guide the model to preserve fundamental topological and semantic graph patterns.

#### 3.3.1 Topology-level augmentation. For topology-level augmentation, we consider a direct way for corrupting input graphs where we randomly remove edges in the graph [52]. Formally, we sample a modified subset \( \tilde{E} \) from the original \( E \) with probability

\[
P((u, v) \in \tilde{E}) = 1 - p^e_{uv},
\]

where \((u, v) \in E\) and \( p^e_{uv} \) is the probability of removing \((u, v)\). \( \tilde{E} \) is then used as the edge set in the generated view. \( p^e_{uv} \) should reflect the importance of the edge \((u, v)\) such that the augmentation function are more likely to corrupt unimportant edges while keep important connective structures intact in augmented views.

In network science, node centrality is a widely-used measure that quantifies the influence of nodes in the graph [29]. We define edge centrality \( w^e_{uv} \) for edge \((u, v)\) to measure its influence based on centrality of two connected nodes. Given a node centrality measure \( \phi_c(\cdot) : \mathcal{V} \to \mathbb{R}_+ \), we define edge centrality as the average of two adjacent nodes’ centrality scores, i.e. \( w^e_{uv} = (\phi_c(u) + \phi_c(v))/2 \), and on directed graph, we simply use the centrality of the tail node, i.e. \( w^e_{uv} = \phi_c(v) \), since the importance of edges is generally characterized by nodes they are pointing to [29].

Next, we calculate the probability of each edge based on its centrality value. Since node centrality values like degrees may vary across orders of magnitude [29], we first set \( s^e_{\text{max}} = \log w^e_{\text{max}} \), to alleviate the impact of nodes with heavily dense connections. The probabilities can then be obtained after a normalization step that transform the values into probabilities, which is defined as

\[
p^e_{uv} = \min \left\{ \frac{s^e_{\text{max}} - s^e_{uv}}{s^e_{\text{max}} - s^e_{\text{min}}} \cdot p_e, \ p_e \right\},
\]

where \( p_e \) is a hyperparameter that controls the overall probability of removing edges. \( s^e_{\text{max}} \) and \( s^e_{\text{min}} \) is the maximum and average of \( s^e_{uv} \), and \( p_e < 1 \) is a cut-off probability, used to truncate the probabilities since extremely high removal probabilities will lead to overly corrupted graph structures.

---

**Algorithm 1: The GCA training algorithm**

1. for \( epoch ← 1, 2, \cdots \)
2. Sample two stochastic augmentation functions \( t \sim \mathcal{T} \) and \( t' \sim \mathcal{T} \)
3. Generate two graph views \( \tilde{G}_1 = t(G) \) and \( \tilde{G}_2 = t'(G) \) by performing corruption on \( G \)
4. Obtain node embeddings \( U \) of \( \tilde{G}_1 \) using the encoder \( f \)
5. Obtain node embeddings \( V \) of \( \tilde{G}_2 \) using the encoder \( f \)
6. Compute the contrastive objective \( \mathcal{J} \) with Eq. (2)
7. Update parameters by applying stochastic gradient ascent to maximize \( \mathcal{J} \)
For the choice of the node centrality function, we use the following three centrality measures, including degree centrality, eigenvector centrality, and PageRank centrality due to their simplicity and effectiveness.

**Degree centrality.** Node degree itself can be a centrality measure [29]. On directed networks, we use in-degrees since the influence of a node in directed graphs are mostly bestowed by nodes pointing at it [29]. Despite that the node degree is one of the simplest centrality measures, it is quite effective and illuminating. For example, in citation networks where nodes represent papers and edges represent citation relationships, nodes with the highest degrees are likely to correspond to influential papers.

**Eigenvector centrality.** The eigenvector centrality [2, 29] of a node is calculated as its eigenvector corresponding to the largest eigenvalue of the adjacency matrix. Unlike degree centrality, which assumes that all neighbors contribute equally to the importance of the node, eigenvector centrality also takes the importance of neighboring nodes into consideration. By definition, the eigenvector centrality of each node is proportional to the sum of centralities of its neighbors, nodes that are either connected to many neighbors or connected to influential nodes will have high eigenvector centrality values. On directed graphs, we use the right eigenvector to compute the centrality, which corresponds to incoming edges. Note that since only the leading eigenvector is needed, the computational burden for calculating the eigenvector centrality is negligible.

**PageRank centrality.** The PageRank centrality [29, 30] is defined as the PageRank weights computed by the PageRank algorithm. The algorithm propagates influence along directed edges, and nodes gathered the most influence are regarded as important nodes. Formally, the centrality values are defined by

\[
\sigma = \alpha AD^{-1} \sigma + 1, \tag{5}
\]

where \( \sigma \in \mathbb{R}^N \) is the vector of PageRank centrality scores for each node and \( \alpha \) is a damping factor that prevents sinks in the graph from absorbing all ranks from other nodes connected to the sinks. We set \( \alpha = 0.85 \) as suggested in Page et al. [30]. For undirected graphs, we execute PageRank on transformed directed graphs, where each undirected edge is converted to two directed edges.

To gain an intuition of these proposed adaptive structural augmentation schemes, we calculate edge centrality scores of the famous Karate club dataset [51], containing two groups of students leading by two coaches respectively. The edge centrality values calculated by different schemes are visualized in Figure 2. As can be seen in the figure, though the three schemes exhibit subtle differences, all of the augmentation schemes tend to emphasize edges that connect the two coaches (in orange) inside the two groups and put less attention to links between peripheral nodes across groups. This verifies that the proposed node-centrality-based adaptive topology augmentation scheme can recognize fundamental structures of the graph.

3.3.2 **Node-attribute-level augmentation.** On the node attribute level, similar to the salt-and-pepper noise in digital image processing [10], we add noise to node attributes via randomly masking a fraction of dimensions with zeros in node features. Formally, we first sample a random vector \( \tilde{m} \in \{0, 1\}^F \) where each dimension of it independently is drawn from a Bernoulli distribution independently, i.e., \( \tilde{m}_i \sim \text{Bern}(1 - p_f) \), \( \forall i \). Then, the generated node features \( \tilde{X} \) is computed by

\[
\tilde{X} = [x_1 \circ \tilde{m}; x_2 \circ \tilde{m}; \ldots; x_N \circ \tilde{m}]^\top. \tag{6}
\]

Here \([\cdot; \cdot]\) is the concatenation operator, and \( \circ \) is the element-wise multiplication.

Similar to topology-level augmentation, the probability \( p_f \) should reflect the importance of the \( i \)-th dimension of node features. We assume that feature dimensions frequently appearing in influential nodes should be important, and define the weights of feature dimensions as follows. For sparse one-hot nodes features, i.e. \( x_{ui} \in \{0, 1\} \) for any node \( u \) and feature dimension \( i \), we calculate the weight of dimension \( i \) as

\[
w_f^i = \sum_{u \in V} x_{ui} \cdot \varphi_e(u), \tag{7}
\]

where \( \varphi_e(\cdot) \) is a node centrality measure that is used to quantify node importance. The first term \( x_{ui} \in \{0, 1\} \) indicates the occurrence of dimension \( i \) in node \( u \), and the second term \( \varphi_e(u) \) measures the node importance of each occurrence. To provide some intuition behind the above definition, consider a citation network where each feature dimension corresponds to a keyword. Then, keywords that frequently appear in a highly influential paper should be considered informative and important.

For dense, continuous node features \( x_{ui} \) of node \( u \), where \( x_{ui} \) denotes feature value at dimension \( i \), we cannot directly count the occurrence of each one-hot encoded value. Then, we turn to measure the magnitude of the feature value at dimension \( i \) of node \( u \) by its absolute value \(|x_{ui}|\). Formally, we calculate the weights by

\[
w_f^i = \sum_{u \in V} |x_{ui}| \cdot \varphi_e(u). \tag{8}
\]

Similar to topology augmentation, we perform normalization on the weights to obtain the probability representing feature importance. Formally,

\[
p^i = \min \left( \frac{x_f^i}{x_f^{\max}}, \frac{x_f^i}{x_f^{\min}} \right), \tag{9}
\]

Figure 2: Visualization of edge centrality computed by three schemes in the Karate club dataset, where centrality values are shown in terms of the thickness of edges. Node colors indicate two classes inside the network; two coaches are in orange.
where $s^f_i = \log w_i^f$, $s^\text{max}_i$ is the maximum and the average value of $s^f_i$ respectively, and $\mu^f_i$ is a hyperparameter that controls the overall magnitude of feature augmentation.

Finally, we generate two corrupted graph views $\mathcal{G}_1, \mathcal{G}_2$ by jointly performing topology- and node-attribute-level augmentation. In GCA, the probability $p_e$ and $p_f$ is different for generating the two views to provide a diverse context for contrastive learning, where the probabilities for the first and the second view are denoted by $p_{e,1}, p_{f,1}$ and $p_{e,2}, p_{f,2}$ respectively.

In this paper, we propose and evaluate three model variants, denoted as GCA-DE, GCA-EV, and GCA-PR. The three variants employ degree, eigenvector, and PageRank centrality measures respectively. Note that all centrality and weight measures are only dependent on the topology and node attributes of the original graph. Therefore, they only need to be computed once and do not bring much computational burden.

### 3.4 Theoretical Justification

In this section, we provide theoretical justification behind our model from two perspectives, i.e. MI maximization and the triplet loss. Detailed proofs can be found in Appendix B.

**Connections to MI maximization.** Firstly, we reveal the connections between our loss and MI maximization between node features and the embeddings in the two views. The InfoMax principle has been widely applied in representation learning literature [1, 35, 41, 42]. MI quantifies the amount of information obtained about one random variable by observing the other random variable.

**Theorem 1.** Let $X_i = \{x_i \}_{j \in N(i)}$ be the neighborhood of node $v_i$ that collectively maps to its output embedding, where $N(i)$ denotes the set of neighbors of node $v_i$ specified by GNN architectures, and $X$ be the corresponding random variable with a uniform distribution $p(X_i) = 1/N$. Given two random variables $U, V \in \mathbb{R}^d$ being the embedding in the two views, with their joint distribution denoted as $p(U,V)$, our objective $J$ is a lower bound of MI between encoder input $X$ and node representations in two graph views $U, V$. Formally,

$$J \leq I(X; U, V).$$

**Proof sketch.** We first observe that our objective $J$ is a lower bound of the InfoNCE objective [35, 43], defined by $I_{\text{NCE}}(U; V) \triangleq \mathbb{E}[1, p(u,v)] \left[ \sum_{i=1}^N \log \frac{e^{f(u_i,v)}}{\sum_{j=1}^N e^{f(u_i,v)}} \right]$. Since the InfoNCE estimator is a lower bound of the true MI, the theorem directly follows from the application of data processing inequality [5], which states that $I(U; V) \leq I(X; U, V)$. \qed

**Remark.** Theorem 1 reveals that maximizing $J$ is equivalent to explicitly maximizing a lower bound of the MI $I(X; U, V)$ between input node features and learned node representations. Recent work further provides empirical evidence that optimizing a stricter bound of MI may not lead to better downstream performance on visual representation learning [41, 42], which further highlights the importance of the design of data augmentation strategies.

When optimizing $I(U; V)$, a lower bound of $I(X; U, V)$, we encourage the model to encode shared information between the two views. From the amortized perspective, corrupted views will follow a skewed distribution where important link structures and features are emphasized. By contrasting the two views, the model is enforced to encode the emphasized information into representations, which improves embedding quality.

However, as the objective is not defined specifically on negative samples generated by the augmentation function, it remains challenging to derive the relationship between specific augmentation functions and the lower bound. We shall leave it for future work.

**Connections to the triplet loss.** Alternatively, we may also view the optimization problem in Eq. (2) as a classical triplet loss, commonly used in deep metric learning.

**Theorem 2.** When the projection function $g$ is the identity function and we measure embedding similarity by simply taking the inner product, i.e. $s(u, v) = u^\top v$, and further assuming that positive pairs are far more aligned than negative pairs, i.e. $u_i^\top v_i < u_i^\top v_j$ and $u_i^\top u_j < u_i^\top v_i$, minimizing the pairwise objective $f(u_i, u_j)$ coincides with maximizing the triplet loss, as given in the sequel

$$f(u_i, u_j) \approx$$

$$4\tau + \sum_{j \neq i} \left( ||v_i - v_j||^2 - ||u_i - v_j||^2 + ||u_i - v_i||^2 - ||u_i - u_j||^2 \right).$$

**Remark.** Theorem 2 draws connections between the objective and the classical triplet loss. In other words, we may regard the problem in Eq. (2) as learning graph convolutional encoders to encourage positive samples being further away from negative samples in the embedding space. Moreover, by viewing the objective from the metric learning perspective, we highlight the importance of appropriate data augmentation schemes, which is often neglected in previous InfoMax-based methods. Specifically, as the objective pulls together representation of each node in the two corrupted views, the model is enforced to encode information in the input graph that is insensitive to perturbation. Since the proposed adaptive augmentation schemes tend to keep important link structures and node attributes intact in the perturbation, the model is guided to encode essential structural and semantic information into the representation, which improves the quality of embeddings. Last, the contrastive objective used in GCA is cheap to optimize, since we do not have to generate negative samples explicitly and all computation can be performed in parallel. In contrast, the triplet loss is known to be computationally expensive [38].

### 4 EXPERIMENTS

In this section, we conduct experiments to evaluate our model through answering the following questions.

- **RQ1.** Does our proposed GCA outperform existing baseline methods on node classification?
- **RQ2.** Do all proposed adaptive graph augmentation schemes benefit the learning of the proposed model? How does each graph augmentation scheme affect model performance?
- **RQ3.** Is the proposed model sensitive to hyperparameters? How do key hyperparameters impact the model performance?

We begin with a brief introduction of the experimental setup, and then we proceed to details of experimental results and their analysis.
4.1 Experimental Setup

4.1.1 Datasets. For comprehensive comparison, we use five widely-used datasets, including Wiki-CS, Amazon-Computers, Amazon-Photo, Coauthor-CS, and Coauthor-Physics, to study the performance of transductive node classification. The datasets are collected from real-world networks from different domains; their detailed statistics is summarized in Table 2.

- Wiki-CS [27] is a reference network constructed based on Wikipedia. The nodes correspond to articles about computer science and edges are hyperlinks between the articles. Nodes are labeled with ten classes each representing a branch of the field. Node features are calculated as the average of pre-trained GloVe [33] word embeddings of words in each article.
- Amazon-Computers and Amazon-Photo [39] are two networks of co-purchase relationships constructed from Amazon, where nodes are goods and two goods are connected when they are frequently bought together. Each node has a sparse bag-of-words feature encoding product reviews and is labeled with its category.
- Coauthor-CS and Coauthor-Physics [39] are two academic networks, which contain co-authorship graphs based on the Microsoft Academic Graph from the KDD Cup 2016 challenge. In these graphs, nodes represent authors and edges indicate co-authorship relationships; that is, two nodes are connected if they have co-authored a paper. Each node has a sparse bag-of-words feature based on paper keywords of the author. The label of an author corresponds to their field. Node features are calculated as the average of pre-trained GloVe [33] word embeddings of words in each article.

Among these datasets, Wiki-CS has dense numerical features, while the other four datasets only contain sparse one-hot features. For the Wiki-CS dataset, we evaluate the models on the public splits shipped with the dataset [27]. Regarding the other four datasets, since they have no public splits available, we instead randomly split the datasets, where 10%, 10%, and the rest 80% of nodes are selected for the training, validation, and test set, respectively.

4.1.2 Evaluation protocol. For every experiment, we follow the linear evaluation scheme as introduced in Veličković et al. [45], where each model is firstly trained in an unsupervised manner; then, the resulting embeddings are used to train and test a simple $\ell_2$-regularized logistic regression classifier. We train the model for twenty runs for different data splits and report the averaged performance on each dataset for fair evaluation. Moreover, we measure performance in terms of accuracy in these experiments.

4.1.3 Baselines. We consider representative baseline methods belonging to the following two categories: (1) traditional methods including DeepWalk [34] and node2vec [11] and (2) deep learning methods including Graph Autoencoders (GAE, VGAE) [22], Deep Graph Infomax (DGI) [45], Graphical Mutual Information Maximization (GMI) [32], and Multi-View Graph Representation Learning (MVGRL) [16]. Furthermore, we report the performance obtained using a logistic regression classifier on raw node features and DeepWalk with embeddings concatenated with input node features. To directly compare our proposed method with supervised counterparts, we also report the performance of two representative models Graph Convolutional Networks (GCN) [23] and Graph Attention Networks (GAT) [44], where they are trained in an end-to-end fashion. For all baselines, we report their performance based on their official implementations.

4.1.4 Implementation details. We employ a two-layer GCN [23] as the encoder for all deep learning baselines due to its simplicity. The encoder architecture is formally given by

$$
GC_j(X, A) = \sigma(\hat{D}^{-\frac{1}{2}} \hat{A} \hat{D}^{-\frac{1}{2}} X W_j),
$$

$$
f(X, A) = GC_2(GC_1(X, A), A),
$$

where $\hat{A} = A + I$ is the adjacency matrix with self-loops. $\hat{D}$ is the degree matrix, $\sigma(\cdot)$ is a nonlinear activation function, e.g., ReLU(·) = max(0, ·), and $W_j$ is a trainable weight matrix. For experimental specifications, including details of the configurations of the optimizer and hyperparameter settings, we refer readers of interest to Appendix A.

4.2 Performance on Node Classification (RQ1)

The empirical performance is summarized in Table 3. Overall, from the table, we can see that our proposed model shows strong performance across all five datasets. GCA consistently performs better than unsupervised baselines by considerable margins on the transductive node classification task. The strong performance verifies the superiority of the proposed contrastive learning framework. On the two Coauthor datasets, we note that existing baselines have already obtained high enough performance; our method GCA still pushes that boundary forward. Moreover, we particularly note that GCA is competitive with models trained with label supervision on all five datasets.

We make other observations as follows. Firstly, the performance of traditional contrastive learning methods like DeepWalk is inferior to the simple logistic regression classifier that only uses raw features on some datasets (Coauthor-CS and Coauthor-Physics), which suggests that these methods may be ineffective in utilizing node features. Unlike traditional work, we see that GCN-based methods, e.g., GAE, are capable of incorporating node features when learning embeddings. However, we note that on certain datasets (Wiki-CS), their performance is still worse than DeepWalk + feature, which we believe can be attributed to their naive method of selecting negative samples that simply chooses contrastive pairs based on edges. This fact further demonstrates the important role of selecting negative features.

Table 2: Statistics of datasets used in experiments.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>#Nodes</th>
<th>#Edges</th>
<th>#Features</th>
<th>#Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wiki-CS</td>
<td>11,701</td>
<td>216,123</td>
<td>300</td>
<td>10</td>
</tr>
<tr>
<td>Amazon-Computers</td>
<td>13,752</td>
<td>245,861</td>
<td>767</td>
<td>10</td>
</tr>
<tr>
<td>Amazon-Photo</td>
<td>7,650</td>
<td>119,081</td>
<td>745</td>
<td>8</td>
</tr>
<tr>
<td>Coauthor-CS</td>
<td>18,333</td>
<td>81,894</td>
<td>6,805</td>
<td>15</td>
</tr>
<tr>
<td>Coauthor-Physics</td>
<td>34,491</td>
<td>247,962</td>
<td>8,415</td>
<td>5</td>
</tr>
</tbody>
</table>

1 https://github.com/pmernyei/wiki-cs-dataset/raw/master/dataset
2 https://github.com/shchur/gnn-benchmark/raw/master/data/npz/amazon_electronics_photo.npz
3 https://github.com/shchur/gnn-benchmark/raw/master/data/npz/amazon_electronics_photo.npz
4 https://github.com/shchur/gnn-benchmark/raw/master/data/npz/ms_academic_cs.npz
5 https://github.com/shchur/gnn-benchmark/raw/master/data/npz/ms_academic_phy.npz
samples based on augmented graph views in contrastive representation learning. Moreover, compared to existing baselines DGI, GMI, and MVGRL, our proposed method performs strong, adaptive data augmentation in constructing negative samples, leading to better performance. Note that, although MVGRL employs diffusion to incorporate global information into augmented views, it still fails to consider the impacts of different edges adaptively on input graphs. The superior performance of GCA verifies that our proposed adaptive data augmentation scheme is able to help improve embedding quality by preserving important patterns during perturbation.

Secondly, we observe that all three variants with different node centrality measures of GCA outperform the other baselines on all datasets. We also notice that GCA–DE and GCA–PR with the degree and PageRank centrality respectively are two strong variants that achieve the best or competitive performance on all datasets. Please kindly note that the result indicates that our model is not limited to specific choices of centrality measures and verifies the effectiveness and generality of our proposed framework.

In summary, the superior performance of GCA compared to existing state-of-the-art methods verifies the effectiveness of our proposed GCA framework that performs data augmentation adaptive to the graph structure and attributes.

### 4.3 Ablation Studies (RQ2)

In this section, we substitute the proposed topology and attribute level augmentation with their uniform counterparts to study the impact of each component of GCA. GCA–T–A denotes the model with uniform topology and node attribute augmentation schemes, where the probabilities of dropping edge and masking features are set to the same for all nodes. The variants GCA–T and GCA–A are defined similarly except that we substitute the topology and attribute augmentation scheme with uniform sampling in the two models respectively. Degree centrality is used in all the variants for fair comparison. Please kindly note that the downgraded GCA–T–A fallbacks to our preliminary work GRACE [52].

The results are presented in Table 4, where we can see that both topology-level and node-attribute-level adaptive augmentation scheme improve model performance consistently on all datasets. In addition, the combination of adaptive augmentation schemes on the two levels further benefits the performance. On the Amazon-Computers dataset, our proposed GCA gains 1.5% absolute improvement compared to the base model with no adaptive augmentation enabled. The results verify the effectiveness of our adaptive augmentation schemes on both topology and node attribute levels.

---

**Table 3:** Summary of performance on node classification in terms of accuracy in percentage with standard deviation. Available data for each method during the training phase is shown in the second column, where X, A, Y correspond to node features, the adjacency matrix, and labels respectively. The highest performance of unsupervised models is highlighted in boldface; the highest performance of supervised models is underlined. OOM indicates Out-Of-Memory on a 32GB GPU.

<table>
<thead>
<tr>
<th>Method</th>
<th>Training Data</th>
<th>Wiki-CS</th>
<th>Amazon-Computers</th>
<th>Amazon-Photo</th>
<th>Coauthor-CS</th>
<th>Coauthor-Physics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Raw features</td>
<td>X</td>
<td>71.98 ± 0.00</td>
<td>73.81 ± 0.00</td>
<td>73.53 ± 0.00</td>
<td>90.37 ± 0.00</td>
<td>93.58 ± 0.00</td>
</tr>
<tr>
<td>node2vec</td>
<td>A</td>
<td>71.79 ± 0.05</td>
<td>84.39 ± 0.08</td>
<td>89.67 ± 0.12</td>
<td>85.08 ± 0.03</td>
<td>91.19 ± 0.04</td>
</tr>
<tr>
<td>DeepWalk</td>
<td>A</td>
<td>74.35 ± 0.06</td>
<td>85.68 ± 0.06</td>
<td>89.44 ± 0.11</td>
<td>84.61 ± 0.22</td>
<td>91.77 ± 0.15</td>
</tr>
<tr>
<td>GCA–T + features</td>
<td>X, A</td>
<td>77.21 ± 0.03</td>
<td>86.28 ± 0.07</td>
<td>90.05 ± 0.08</td>
<td>87.70 ± 0.04</td>
<td>94.90 ± 0.09</td>
</tr>
<tr>
<td>GAE</td>
<td>X, A</td>
<td>70.15 ± 0.01</td>
<td>85.27 ± 0.19</td>
<td>91.62 ± 0.13</td>
<td>90.01 ± 0.71</td>
<td>94.92 ± 0.07</td>
</tr>
<tr>
<td>VGAE</td>
<td>X, A</td>
<td>75.63 ± 0.19</td>
<td>86.37 ± 0.21</td>
<td>92.20 ± 0.11</td>
<td>92.11 ± 0.09</td>
<td>94.52 ± 0.00</td>
</tr>
<tr>
<td>DGI</td>
<td>X, A</td>
<td>75.35 ± 0.14</td>
<td>83.95 ± 0.47</td>
<td>91.61 ± 0.22</td>
<td>92.15 ± 0.63</td>
<td>94.51 ± 0.52</td>
</tr>
<tr>
<td>GMI</td>
<td>X, A</td>
<td>74.85 ± 0.08</td>
<td>82.21 ± 0.31</td>
<td>90.68 ± 0.17</td>
<td>OOM</td>
<td>OOM</td>
</tr>
<tr>
<td>MVGRL</td>
<td>X, A</td>
<td>77.52 ± 0.08</td>
<td>87.52 ± 0.11</td>
<td>91.74 ± 0.07</td>
<td>92.11 ± 0.12</td>
<td>95.33 ± 0.03</td>
</tr>
<tr>
<td>GCA–DE</td>
<td>X, A</td>
<td>78.30 ± 0.00</td>
<td>87.85 ± 0.31</td>
<td>92.49 ± 0.09</td>
<td>93.10 ± 0.01</td>
<td>95.68 ± 0.05</td>
</tr>
<tr>
<td>GCA–PR</td>
<td>X, A</td>
<td>78.35 ± 0.05</td>
<td>87.80 ± 0.23</td>
<td>92.53 ± 0.16</td>
<td>93.06 ± 0.03</td>
<td>95.72 ± 0.03</td>
</tr>
<tr>
<td>GCA–EV</td>
<td>X, A</td>
<td>78.23 ± 0.04</td>
<td>87.54 ± 0.49</td>
<td>92.24 ± 0.21</td>
<td>92.95 ± 0.13</td>
<td>95.73 ± 0.03</td>
</tr>
<tr>
<td>GCN</td>
<td>X, A, Y</td>
<td>77.19 ± 0.12</td>
<td>86.51 ± 0.54</td>
<td>92.42 ± 0.22</td>
<td>93.03 ± 0.31</td>
<td>95.65 ± 0.16</td>
</tr>
<tr>
<td>GAT</td>
<td>X, A, Y</td>
<td>77.65 ± 0.11</td>
<td>86.93 ± 0.29</td>
<td>92.56 ± 0.35</td>
<td>92.31 ± 0.24</td>
<td>95.47 ± 0.15</td>
</tr>
</tbody>
</table>

**Table 4:** Performance of model variants on node classification in terms of accuracy in percentage with standard deviation. We use the degree centrality in all variants. The highest performance is highlighted in boldface.

<table>
<thead>
<tr>
<th>Variant</th>
<th>Topology</th>
<th>Attribute</th>
<th>Wiki-CS</th>
<th>Amazon-Computers</th>
<th>Amazon-Photo</th>
<th>Coauthor-CS</th>
<th>Coauthor-Physics</th>
</tr>
</thead>
<tbody>
<tr>
<td>GCA–T–A</td>
<td>Uniform</td>
<td>Uniform</td>
<td>78.19 ± 0.01</td>
<td>86.25 ± 0.25</td>
<td>92.15 ± 0.24</td>
<td>92.93 ± 0.01</td>
<td>95.26 ± 0.02</td>
</tr>
<tr>
<td>GCA–T</td>
<td>Uniform</td>
<td>Adaptive</td>
<td>78.23 ± 0.02</td>
<td>86.72 ± 0.49</td>
<td>92.20 ± 0.26</td>
<td>93.07 ± 0.01</td>
<td>95.59 ± 0.04</td>
</tr>
<tr>
<td>GCA–A</td>
<td>Adaptive</td>
<td>Uniform</td>
<td>78.25 ± 0.02</td>
<td>87.66 ± 0.30</td>
<td>92.23 ± 0.20</td>
<td>93.02 ± 0.01</td>
<td>95.54 ± 0.02</td>
</tr>
<tr>
<td>GCA</td>
<td>Adaptive</td>
<td>Adaptive</td>
<td>78.30 ± 0.01</td>
<td>87.85 ± 0.31</td>
<td>92.49 ± 0.09</td>
<td>93.10 ± 0.01</td>
<td>95.68 ± 0.05</td>
</tr>
</tbody>
</table>
We thus conclude that, overall, our model is insensitive to these
noise to unimportant feature dimensions to emphasize the under-
lying semantic information. We have conducted comprehensive
experiments using various real-world datasets. Experimental results
demonstrate that our proposed GCA method consistently outper-
forms existing state-of-the-art methods and even surpasses several
supervised counterparts.

ACKNOWLEDGMENTS
The authors would like to thank anonymous reviewers for their
insightful comments. The authors would also like to thank Mr.
Tao Sun and Mr. Sirui Lu for their valuable discussion. This work
is jointly supported by National Natural Science Foundation of
China (U19B2038, 61772528) and Beijing National Natural Science
Foundation (4182066).

DISCUSSIONS ON BROADER IMPACT
This paper presents a novel graph contrastive learning framework,
and we believe it would be beneficial to the graph machine learn-
ing community both theoretically and practically. Our proposed
self-supervised graph representation learning techniques help al-
leviate the label scarcity issue when deploying machine learning
applications in real-world, which saves a lot of efforts on human
annotating. For example, our GCA framework can be plugged into
existing recommender systems and produces high-quality embed-
dings for users and items to resolve the cold start problem. Note
that our work mainly serves as a plug-in module for existing ma-
chine learning pipelines, it does not bring new ethical concerns.
However, the GCA model may still give biased outputs (e.g., gender
bias, ethnicity bias), as the provided data itself may be strongly
biased during the processes of data measurement and collection,
graph construction, etc.

A IMPLEMENTATION DETAILS
A.1 Computing Infrastructures

Software infrastructures. All models are implemented using Py-
Torch Geometric 1.6.1 [7], PyTorch 1.6.0 [31], and NetworkX 2.5
[13]. All datasets used throughout experiments are available in
PyTorch Geometric libraries.

Hardware infrastructures. We conduct experiments on a com-
puter server with four NVIDIA Tesla V100S GPUs (with 32GB
memory each) and twelve Intel Xeon Silver 4214 CPUs.

A.2 Hyperparameter Specifications
All model parameters are initialized with Glorot initialization [9],
and trained using the Adam SGD optimizer [21] on all datasets. The
\(\ell_2\) weight decay factor is set to \(10^{-5}\) and the dropout rate [40] is
set to zero on all datasets. The probability parameters controlling
the sampling process, \(p_e,1, p_f,1\), for the first view and \(p_e,2, p_f,2\) for
the second view, are all selected between 0.0 and 0.4 in order to
prevent the original graph from being overly corrupted. Note that
to generate different contexts for nodes in the two views, we set
\(p_e,1\) and \(p_e,2\) to be distinct, and the same for \(p_f,1\) and \(p_f,2\). We
summarize all dataset-specific hyperparameter configurations in
Table 5.
We further define the InfoNCE estimator as a lower bound of MI between encoder input and node representations in two graph views. Formally, the InfoNCE objective, which is defined as

\[ J \leq \mathcal{I}(X;U,V). \]  

**Proof.** We first show the connection between our objective \( J \) and the InfoNCE objective \([35, 43]\), which is defined as

\[ \mathcal{I}_{\text{NCE}}(U,V) \triangleq \mathbb{E}_{\mathcal{P}(u,v)} \left[ \frac{1}{N} \sum_{i=1}^{N} \log \left( \frac{e^{\theta(u_i,v_i)/\tau}}{\rho_c(u_i) / \rho_c(v_i)} \right) \right], \]

where the critic function is defined as \( \theta(x,y) = s(g(x),g(y)) \). We further define \( \rho_c(u_i) = \sum_{i=1}^{N} \exp(\theta(u_i,u_j)/\tau) \) and \( \rho_c(v_i) = \sum_{i=1}^{N} \exp(\theta(u_i,v_j)/\tau) \) for convenience of notation. \( \rho_c(u_i) \) and \( \rho_c(v_i) \) can be defined symmetrically. Then, our objective \( J \) can be rewritten as

\[ J = \mathbb{E}_{\mathcal{P}(u,v)} \left[ \frac{1}{N} \sum_{i=1}^{N} \log \left( \frac{e^{\theta(u_i,v_i)/\tau}}{\rho_c(u_i)} \right) \right]. \]

Using the notation of \( \rho_c \), the InfoNCE estimator \( \mathcal{I}_{\text{NCE}} \) can be written as

\[ \mathcal{I}_{\text{NCE}}(U,V) = \mathbb{E}_{\mathcal{P}(u,v)} \left[ \frac{1}{N} \sum_{i=1}^{N} \log \left( \frac{1 + \rho_r(u_i)}{\rho_c(u_i)} \right) \right]. \]

Therefore,

\[ 2J = \mathcal{I}_{\text{NCE}}(U,V) - \mathbb{E}_{\mathcal{P}(u,v)} \left[ \frac{1}{N} \sum_{i=1}^{N} \log \left( 1 + \frac{\rho_r(u_i)}{\rho_c(u_i)} \right) \right] \]

\[ + \mathcal{I}_{\text{NCE}}(V,U) - \mathbb{E}_{\mathcal{P}(u,v)} \left[ \frac{1}{N} \sum_{i=1}^{N} \log \left( 1 + \frac{\rho_r(v_i)}{\rho_c(v_i)} \right) \right] \]

\[ \leq \mathcal{I}_{\text{NCE}}(U,V) + \mathcal{I}_{\text{NCE}}(V,U). \]

According to Poole et al. [35], the InfoNCE estimator is a lower bound of the true MI, i.e.

\[ \mathcal{I}_{\text{NCE}}(U,V) \leq \mathcal{I}(U;V). \]  

Thus, we arrive at

\[ 2J \leq \mathcal{I}(U;V) + \mathcal{I}(V;U) = 2\mathcal{I}(U;V), \]

which leads to the inequality

\[ J \leq \mathcal{I}(U;V). \]

According to the data processing inequality [5], which states that, for all random variables \( X, Y, Z \) satisfying the Markov relation \( X \rightarrow Y \rightarrow Z \), the inequality \( \mathcal{I}(X;Z) \leq \mathcal{I}(X;Y) \) holds. Then, we observe that \( X, U, V \) satisfy the relation \( U \leftarrow X \rightarrow V \). Since, \( U \) and \( V \) are conditionally independent after observing \( X \), the relation is Markov equivalent to \( X \rightarrow U \rightarrow V \), which leads to \( \mathcal{I}(U;V) \leq \mathcal{I}(X;U) \). We further notice that the relation \( X \rightarrow (U, V) \rightarrow U \) holds, and hence it follows that \( \mathcal{I}(X;U,V) \leq \mathcal{I}(X;U) \). Combining the two inequalities yields the required inequality

\[ \mathcal{I}(U;V) \leq \mathcal{I}(X;U), \]

which concludes the proof.

**B.2 Proof of Theorem 2**

**Theorem 2.** When the projection function \( g \) is the identity function and we measure embedding similarity by simply taking inner product, and further assuming that positive pairs are far more aligned than negative pairs, \( i.e. \ u_i^2 v_k \ll u_i^2 v_i \) and \( u_i^2 u_k \ll u_i^2 v_i \), minimizing the pairwise objective \( \ell(u_i, v_i) \) coincides with maximizing the triplet loss, as given in the sequel

\[ 4\tau + \sum_{j \neq i} \left( ||u_i - v_i||^2 - ||u_i - v_j||^2 - ||u_i - u_j||^2 \right) \]

\[ = \ell(u_i, v_i). \]

**Proof.** Based on the assumptions, we can rearrange the pairwise objective as

\[ -\ell(u_i, v_i) = -\log \sum_{k \neq i} e^{(u_i^2 v_k)/\tau} = e^{(u_i^2 v_i)/\tau} \sum_{k \neq i} e^{(u_i^2 v_k)/\tau} \]

\[ = \log \left( 1 + \sum_{k \neq i} e^{(u_i^2 v_k - u_i^2 v_i)/\tau} + \sum_{k \neq i} e^{(u_i^2 v_k - u_i^2 v_i)/\tau} \right). \]

Table 5: Hyperparameter specifications.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>( p_{c,1} )</th>
<th>( p_{c,2} )</th>
<th>( p_{f,1} )</th>
<th>( p_{f,2} )</th>
<th>( \tau )</th>
<th>Learning rate</th>
<th>Training epochs</th>
<th>Hidden dimension</th>
<th>Activation function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wiki-CS</td>
<td>0.2</td>
<td>0.4</td>
<td>0.1</td>
<td>0.1</td>
<td>0.7</td>
<td>0.6</td>
<td>0.01</td>
<td>3,000</td>
<td>256</td>
</tr>
<tr>
<td>Amazon-Computers</td>
<td>0.5</td>
<td>0.5</td>
<td>0.2</td>
<td>0.1</td>
<td>0.7</td>
<td>0.1</td>
<td>0.01</td>
<td>1,500</td>
<td>128</td>
</tr>
<tr>
<td>Amazon-Photo</td>
<td>0.3</td>
<td>0.5</td>
<td>0.1</td>
<td>0.1</td>
<td>0.7</td>
<td>0.3</td>
<td>0.1</td>
<td>2,000</td>
<td>256</td>
</tr>
<tr>
<td>Coauthor-CS</td>
<td>0.3</td>
<td>0.2</td>
<td>0.3</td>
<td>0.4</td>
<td>0.7</td>
<td>0.4</td>
<td>0.0005</td>
<td>1,000</td>
<td>256</td>
</tr>
<tr>
<td>Coauthor-Physics</td>
<td>0.4</td>
<td>0.1</td>
<td>0.1</td>
<td>0.4</td>
<td>0.7</td>
<td>0.5</td>
<td>0.01</td>
<td>1,500</td>
<td>128</td>
</tr>
</tbody>
</table>

© 2021, Yanqiao Zhu, Yichen Xu, Feng Yu, Qiang Liu, Shu Wu, and Liang Wang
By Taylor expansion of first order,
\[
- \ell(\mathbf{u}_i, \mathbf{v}_j) \approx \frac{1}{r} \sum_{k \neq i} \left( \|\mathbf{u}_i - \mathbf{u}_k\|^2 + \|\mathbf{u}_i - \mathbf{v}_j\|^2 - 2 \mathbf{u}_i^\top \mathbf{u}_k - 2 \mathbf{u}_i^\top \mathbf{v}_j \right) \tag{25}
\]
which concludes the proof.

REFERENCES


[27] Péter Memics and Catalina Cangea. 2020. Wiki-CS: A Wikipedia-Based Benchmark for Graph Neural Networks. In ICML Workshop on Graph Representation Learning and Beyond.


[36] Jiezhong Qiu, Qibin Chen, Yuxiao Dong, Jing Zhang, Hongxia Yang, Ming Ding, Kuanzan Wang, and Jie Tang. 2020. GCC: Graph Contrastive Coding for Graph Neural Network Pre-Training. In Proceedings of the 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. ACM, 1150–1160.


