



GAugLLM: Improving Graph Contrastive Learning for Text-Attributed Graphs with Large Language Models

Yi Fang
SFSC of AI and DL
New York University(Shanghai)
Shanghai, China
yf2722@nyu.edu

Daochen Zha
Department of Computer Science
Rice University
Huston, USA
daochen.zha@rice.edu

Dongzhe Fan
SFSC of AI and DL
New York University(Shanghai)
Shanghai, China
df2362@nyu.edu

Qiaoyu Tan
SFSC of AI and DL
New York University(Shanghai)
Shanghai, China
qiaoyu.tan@nyu.edu

Abstract

This work studies self-supervised graph learning for text-attributed graphs (TAGs) where nodes are represented by textual attributes. Unlike traditional graph contrastive methods that perturb the numerical feature space and alter the graph's topological structure, we aim to improve view generation through **language supervision**. This is driven by the prevalence of textual attributes in real applications, which complement graph structures with rich semantic information. However, this presents challenges because of two major reasons. First, text attributes often vary in length and quality, making it difficult to perturb raw text descriptions without altering their original semantic meanings. Second, although text attributes complement graph structures, they are not inherently well-aligned. To bridge the gap, we introduce GAugLLM, a novel framework for augmenting TAGs. It leverages advanced large language models like Mistral to enhance self-supervised graph learning. Specifically, we introduce a mixture-of-prompt-expert technique to generate augmented node features. This approach adaptively maps multiple prompt experts, each of which modifies raw text attributes using prompt engineering, into numerical feature space. Additionally, we devise a collaborative edge modifier to leverage structural and textual commonalities, enhancing edge augmentation by examining or building connections between nodes. Empirical results across five benchmark datasets spanning various domains underscore our framework's ability to enhance the performance of leading contrastive methods (e.g., BGRL, GraphCL, and GBT) as a plug-in tool. Notably, we observe that the augmented features and graph structure can also enhance the performance of standard generative methods (e.g., GraphMAE and S2GAE), as well as popular graph neural networks (e.g., GCN and GAT).

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

KDD '24, August 25–29, 2024, Barcelona, Spain

© 2024 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 979-8-4007-0490-1/24/08
<https://doi.org/10.1145/3637528.3672035>

The open-sourced implementation of our GAugLLM is available at <https://github.com/NYUSHCS/GAugLLM>.

Keywords

Graph contrastive learning, LLM for graph augmentation, Text-attributed graphs, Graph neural networks

ACM Reference Format:

Yi Fang, Dongzhe Fan, Daochen Zha, and Qiaoyu Tan. 2024. GAugLLM: Improving Graph Contrastive Learning for Text-Attributed Graphs with Large Language Models. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '24)*, August 25–29, 2024, Barcelona, Spain. ACM, New York, NY, USA, 12 pages. <https://doi.org/10.1145/3637528.3672035>

1 Introduction

Graph data is ubiquitous across various domains, including traffic, e-commerce, chemistry, and bioinformatics. Unlike grid-like data such as images and text, graphs are non-Euclidean structures that capture intricate relationships between nodes, featuring diverse connection patterns. To address the complexities of graph data, Graph Neural Networks (GNNs) have emerged as specialized tools for representation learning [28, 35, 50]. GNNs possess the capability to iteratively update node representations by aggregating information from neighboring nodes and themselves. Traditionally, the majority of GNN research has concentrated on supervised learning scenarios, where an ample amount of labeled graph data is available. However, annotating graph data is a laborious and expensive task.

Consequently, recent attention [35] has shifted towards self-supervised graph learning, where the goal is to pre-train GNNs by generating training signals from unlabeled data itself. Once pre-trained, these models can serve as strong initializations for downstream supervised tasks with limited labeled samples [6, 9, 24, 27, 30, 36, 38, 53], such as semi-supervised or few-shot learning scenarios. Graph contrastive learning (GCL), a prominent area in self-supervised graph learning, has shown remarkable effectiveness in pre-training GNNs [35]. Existing GCL research, exemplified by GraphCL [43] and BGRL [31], operate by creating two augmented views of the input graph and subsequently training GNN encoder to produce similar representations for both views of the same node. Various GCL methods [19, 37] differ in their designs for feature- and

structure-level augmentation [4, 9] and employ different contrastive learning objectives, e.g., InfoNCE [23] and Barlow Twins [44].

Despite the numerous GCL methods proposed in recent years [1, 13, 34, 45, 54], they exhibit limitations when applied to graphs enriched with textual descriptions, often referred to as text-attributed graphs (TAGs). A typical example of TAGs is citation networks, where each node represents a research paper and includes text attributes like titles and abstracts. These text attributes offer valuable information for enhancing graph learning due to their expressiveness, capturing intricate semantic nuances. However, previous GCL efforts simply utilize textual attributes to derive numerical features using shallow embedding models such as Word2vec [20] or Bag-of-Words (BoW) [8]. Subsequently, they perform feature-level perturbation on this transformed feature space. While conceptually simple, this feature augmentation strategy is inherently suboptimal. It cannot fully capture the complexity of semantic features [2, 10], and the quality of augmented features is constrained by the text transformation function used. Furthermore, these methods perform structure augmentation in an attribute-agnostic manner, relying solely on stochastic perturbation functions like edge masking. Nevertheless, as previously discussed in [7, 17, 36], randomly perturbing edges in the original graph can be risky. Therefore, text attributes represent a valuable resource to advance graph augmentations for effective contrastive learning.

However, leveraging text attributes for effective graph augmentation presents several challenges. **Firstly, maintaining original semantic meanings while performing text augmentation is difficult**, as text attributes in real-world graphs often vary in length and quality (see Table 1). Traditional heuristic augmentation strategies, such as random word replacement, insertion and swap, may be sub-optimal in such cases. **Secondly, mapping augmented text attributes into numerical space poses another challenge**. Unlike traditional GCL methods that transform text data into feature vectors in the pre-processing step, directly perturbing input text attributes requires a principled text transformation function capable of capturing the disparity between augmented and original text attributes. Moreover, this transformation function should be personalized w.r.t. each node, as nodes in a graph often exhibit different characteristics. **Thirdly, augmenting topological structure solely based on text attributes is ineffective and inefficient**, due to the heterogeneity of text attributes and graph structure. While an intuitive solution is to estimate edge weights between nodes by calculating their similarity in the text space and generating an augmented graph by sampling over the edge space using estimated edge weights, this approach suffers from scalability issues. The complexity is quadratic to the graph size, which could be millions or even billions in practice. Moreover, it may lead to a sub-par augmented graph with connection patterns significantly different from the original graph topology since text attributes and graph structure are not well aligned in general. Hence, an effective structure augmentation strategy should jointly consider both text attributes and the original graph structure.

To fill this research gap, in this work, we present GAugLLM, a novel graph augmentation framework for self-supervised learning on graphs. The key idea is to utilize advanced large language models (LLMs), such as Mistral and LLaMa, to perturb and extract valuable information in the text space, enabling effective feature-

structure-level augmentation. Specifically, to address the first two challenges, we introduce a mixture-of-prompt-expert technique to perturb original text attributes based on diverse prompt experts, each representing a specific prompt template tailored to an LLM. Subsequently, a smaller LLM (e.g., BERT) is fine-tuned to dynamically integrate multiple augmented text attributes into the feature space. This transformation considers node statistics and adopts observed node connections as training supervision. To tackle the third challenge, we propose a collaborative edge modifier strategy. This approach reduces augmentation complexity by prioritizing the most spurious and likely connections between each node and others from a structural perspective. Then an LLM is adopted to identify the most promising connections in the context of text attributes. Overall, our main contributions are summarized below:

- We introduce a novel graph augmentation approach, namely GAugLLM, designed for text-attributed graphs. Unlike standard GCL methods that solely transform text attributes into feature vectors and conduct feature- and edge-level perturbation independently, GAugLLM leverages rich text attributes with LLMs to jointly perform perturbation in both feature and edge levels.
- We propose a mixture-of-prompt-expert method to generate augmented features by directly perturbing on the input text attributes. Unlike heuristic-based random perturbation, we utilize powerful LLMs to disturb text attributes from diverse prompt aspects, which are then dynamically integrated into a unified feature space as augmented features.
- We devise a collaborative edge modifier scheme to leverage text attributes for structural perturbation. Unlike traditional edge perturbation functions, e.g., random masking, we offer a principled approach that adds and deletes node connections by jointly looking at the textual and structural spaces.
- We extensively experiment on various TAG benchmarks across different scales and domains to validate the effectiveness of GAugLLM. Our empirical results demonstrate that GAugLLM improves the performance of leading contrastive methods (e.g., BGRL, GraphCL, and GBT), with up to 12.3% improvement. Additionally, we consistently observe gains by utilizing the augmented features and structures of our model on popular generative methods (e.g., GraphMAE and S2GAE) and graph neural networks (e.g., GCN and GAT).

2 Related Work

Our work is closely related to the following two directions. Readers, who are interested in GNNs and LLMs, please refer to [40] and [21, 49] for a comprehensive review.

Self-supervised learning on graphs. Self-supervised learning has become a compelling paradigm for learning representations from graph-structured data without explicit annotations. The existing work can be mainly divided into two categories: contrastive learning methods and generative methods. Contrastive learning approaches learn graph representations by maximizing the similarity between positive pairs while minimizing the similarity between negative pairs. Previous research, such as GraphCL [43], has further advanced contrastive learning methods by introducing various graph data augmentation techniques. These methods generally rely on effective strategies for positive and negative sample pairing and

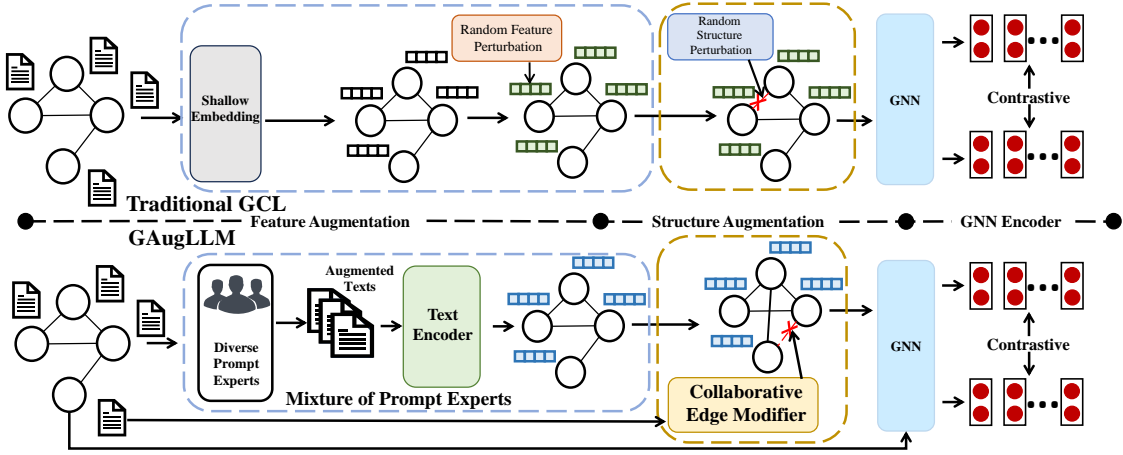


Figure 1: The learning paradigm of GAugLLM vs. traditional GCL methods on TAGs. While standard GCL methodologies rely on text attributes primarily to generate numerical node features via shallow embedding models, such as word2vec, our GAugLLM endeavors to advance contrastive learning on graphs through advanced LLMs. This includes the direct perturbation of raw text attributes for feature augmentation, facilitated by a novel mixture-of-prompt-experts technique. Additionally, GAugLLM harnesses both structural and textual commonalities to effectively perturb edges deemed most spurious or likely to be connected, thereby enhancing structure augmentation.

robust Graph Neural Network (GNN) architectures to extract graph features. More recently, GPA [47] provides personalized augmentation methods for for graphs. Generative methods focus on learning graph representations by predicting unseen parts of the graph. For instance, S2GAE [29] masks edges in the graph and predicts missing links, while GraphMAE [11] utilizes GNN models as the encoder and decoder to reconstruct masked node features. Recently, GiGaMAE [26] learns more generalized and comprehensive knowledge by considering embeddings encompassing graph topology and attribute information as reconstruction targets. Generative methods encourage the model to capture the intrinsic structure and evolution patterns of graphs, leading to richer and more insightful graph representations.

Representation learning on TAGs. Text-attributed graphs have recently received significant attention in both academia and industry. Initially, representation learning on TAGs relied on shallow embedding methods. Although these approaches provided a foundation for representation learning on TAGs, they are limited by their inability to deeply integrate text and graph structure information. GIANT [3] represents a leap forward by more effectively integrating deep textual information with graph topology. By doing so, GIANT can capture complex dependencies and interactions between text and structure, significantly improving performance on downstream tasks. Recently, some studies have been focused on leveraging the sophisticated capabilities of LLMs to enhance the understanding and analysis of TAGs. TAPE [10] leverages LLMs for generating explanations as features, which then serve as inputs for graph neural networks (GNNs), thereby enriching the representation of TAGs. GLEM [48] proposes a novel approach that combines GNNs and LMs within a variational Expectation-Maximization (EM) framework for node representation learning in TAGs. However, they mainly focus on supervised training.

3 Preliminary

In this section, we introduce notations, formalize the research problem of this work, and illustrate prospective opportunities for harnessing language models to enhance contrastive learning on TAGs. **Text-Attributed Graphs.** We are given a TAG $\mathcal{G} = \{\mathcal{V}, \mathcal{S}, \mathbf{A}\}$ with N nodes, where \mathcal{V} denotes the node set, and $\mathbf{A} \in \mathbb{R}^{N \times N}$ represents the adjacency matrix. For each node $v \in \mathcal{V}$ is associated with a textual attribute S_v , and $\mathcal{S} = \{S_v | v \in \mathcal{V}\}$ is the attribute set.

In this work, we study self-supervised learning on TAGs. Specifically, the goal is to pre-train a mapping function $f_\theta : \mathcal{S} \times \mathbf{A} \rightarrow \mathbb{R}^d$, so that the semantic information in \mathcal{S} and the topological structure in \mathbf{A} could be effectively captured in the d -dimensional space in a self-supervised manner.

Graph Neural Networks. For graph-structure data, graph neural networks (GNNs) are often applied to instantiate f_θ . Specifically, the goal of GNNs is to update node representation by aggregating messages from its neighbors, expressed as:

$$\mathbf{h}_v^{(k)} = \text{COM}(\mathbf{h}_v^{(k-1)}, \text{AGG}(\{\mathbf{h}_u^{(k-1)} : u \in \mathcal{N}_v\})), \quad (1)$$

where $\mathbf{h}_v^{(k)}$ denotes the representation of node v at the k -th layer and $\mathcal{N}_v = \{u | \mathbf{A}_{v,u} = 1\}$ is a direct neighbor set of v . In particular, we have $\mathbf{h}_v^{(0)} = \mathbf{x}_v$, in which $\mathbf{x}_v = \text{Emb}(S_v) \in \mathbb{R}^F$ is a F -dimensional numerical vector extracted from v 's textual attribute S_v and $\text{Emb}(\cdot)$ stands for embedding function. The function AGG is used to aggregate features from neighbors [16], and function COM is used to combine the aggregated neighbor information and its own node embedding from the previous layer [32].

Graph Contrastive Learning on TAGs. Let $\tau_f : \mathbb{R}^F \rightarrow \mathbb{R}^F$ and $\tau_s : \mathcal{V} \times \mathcal{V} \rightarrow \mathcal{V} \times \mathcal{V}$ represent the feature-level and structure-level perturbation functions, respectively. An example of τ_f is feature masking [15], while for τ_s , edge masking [52] serves as a typical illustration. Previous GCL endeavors [41, 42, 46] typically start by

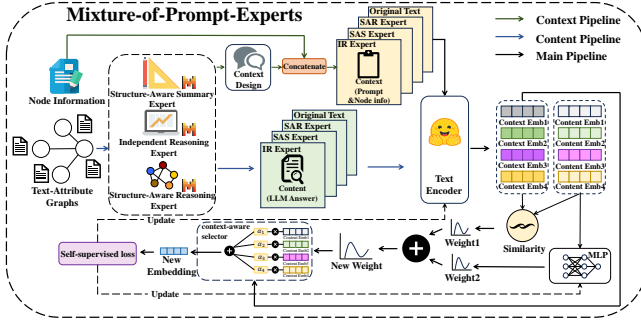


Figure 2: The pipeline of the mixture-of-prompt-experts for feature augmentation. It takes a TAG as input and then utilizes multiple prompt experts to perturb the original text attributes, generating diverse augmented attributes. These augmented text attributes are then integrated into a unified augmentation feature by considering the graph statistics as attention context.

employing a shallow embedding function $g : S \rightarrow \mathbb{R}^F$, such as Word2vec and BoW, to transform text attributes into numerical feature vectors, i.e., $\mathbf{x}_v = g(S_v)$ as a preprocessing step. Subsequently, they generate two augmented graphs, $\mathcal{G}_1 = (\mathbf{A}_1, \mathbf{X}_1)$ and $\mathcal{G}_2 = (\mathbf{A}_2, \mathbf{X}_2)$, by applying perturbation functions to the transformed feature space \mathbf{X} and graph structure \mathbf{A} . Here, $\mathbf{X}_1 = \{\tau_f^1(\mathbf{x}_v) | v \in \mathcal{V}\}$, $\mathbf{A}_1 = \tau_s^1(\mathbf{A})$, $\mathbf{X}_2 = \{\tau_f^2(\mathbf{x}_v) | v \in \mathcal{V}\}$, and $\mathbf{A}_2 = \tau_s^2(\mathbf{A})$. Then, two sets of node representations are acquired for the two views using a shared GNN encoder, denoted as \mathbf{H}_1 and \mathbf{H}_2 , respectively. Finally, the GNN encoder is trained to maximize the similarity between \mathbf{H}_1 and \mathbf{H}_2 on a node-wise basis. In this study, we mainly focus on three state-of-the-art methods, namely GraphCL [43], BGRL [31], and GBT [1], for experimentation.

Opportunity. Existing GNN studies have been restricted in their utilization of text attributes, which are both informative and valuable in TAGs [39]. First, the shallow embedding function g is limited in its ability to comprehend the semantic information of text attributes, particularly when compared with LLMs like Mistral and LLaMa. Second, it is well understood that node attributes and graph structure are complementary to each other [14, 18]. Therefore, merely perturbing the graph structure without considering their semantic similarity may result in a suboptimal augmented graph, whose semantic meaning diverges significantly from the original structure [17]. Motivated by the above opportunities for improvement, in this work, we explore the following research question: *Can we leverage text attributes to enhance the performance of graph contrastive learning from the perspective of graph augmentation?*

4 Methodology

In this section, we present the proposed GAugLLM shown in Figure 1. We first discuss how to perturb raw text attributes for effective feature augmentation (in Section 4.1). Then, we elaborate on a tailored collaborative edge modifier to effectively add or delete edges for structure augmentation (in Section 4.2). Finally, we show how the proposed feature- and structure-level augmentation strategies can be extended to the standard GCL pipeline (in Section 4.3).

4.1 Mixture-of-Prompt-Experts

As discussed above, traditional GCL methods are limited in leveraging rich text attributes for feature augmentation, as they solely rely on a shallow embedding model to transform text attributes into the feature space during a pre-processing step. These transformed features are then fed into a perturbation function τ_s for feature perturbation. To make full use of text attributes for feature augmentation, we propose a novel framework called mixture-of-prompt-experts.

Figure 2 depicts the overall architecture, which offers an elegant approach to directly perturb text attributes and map them into the feature space. Given a TAG $\mathcal{G} = (\mathcal{V}, \mathcal{S}, \mathbf{A})$ as input, our model initially perturbs the text attribute S_v of node v into diverse augmented texts $(\{\hat{S}_v^i\}_{i=1}^m)$ using different prompt experts $\{f_{pe}^i\}_{i=1}^m$, where m represents the number of total experts. Let $f_{\Theta_{\text{text}}}$ denote the text transformation function with parameters Θ_{text} , and $\hat{\mathbf{x}}_v^i$ indicate the hidden embedding of the i -th augmented text produced by f_{pe}^i .

4.1.1 Prompt experts. Our mixture-of-prompt-experts approach begins by configuring a diverse set of prompt experts to perturb the raw text attribute S_v while preserving its semantic meanings. Motivated by the remarkable success of LLMs (e.g., LLaMA and Mistral) in understanding and generating natural language, we initialize our prompt experts with LLM yet with different prompt designs. Specifically, we design three different prompt templates to perturb the raw text attributes from the structural and reasoning perspectives, as illustrated below.

- **Structure-Aware Summarization (SAS Expert).** Let $\mathcal{S}_v^N = \{S_u | v \in \mathcal{N}_v\}$ represent the textual attribute set of node v 's neighbors. The idea of SAS is to query the LLM to create a summary of the anchor node v by comprehending the semantic information from both its neighbors and itself. The general prompt format is illustrated in Figure 7.
- **Independent Reasoning (IDR Expert).** In contrast to SAS, which concentrates on text summarization, IDR adopts an “open-ended” approach when querying the LLM. This entails instructing the model to make predictions across potential categories and to provide explanations for its decisions. The underlying philosophy here is that such a reasoning task will prompt the LLM to comprehend the semantic significance of the input textual attribute at a higher level, with an emphasis on the most vital and relevant factors [10]. The general prompt format is illustrated in Figure 7.
- **Structure-Aware Reasoning (SAR Expert).** Taking a step beyond IDR, SAR integrates structural information into the reasoning process. The rationale for this lies in the notion that connected nodes can aid in deducing the topic of the anchor node. The general prompt format is given in Figure 7.

Based on the three prompt experts, we can map the text attribute S_v of each node v into three augmented texts $\{\hat{S}_v^i | i \in \{\text{SAS}, \text{IDR}, \text{SAR}\}\}$

4.1.2 Text encoder. After perturbing the raw text attributes, we need to train a text encoder mapping the augmented texts into hidden space. Instead of using shallow embedding algorithm, we aim to fine-tune a smaller LLM (e.g., BERT) to encode the domain-specific text data. In particular, given the augmented text set $\{\hat{S}_v^i | i \in \{\text{SAS}, \text{IDR}, \text{SAR}, \text{Raw}\}\}$ of node v , the text encoder works as follows:

$$\hat{\mathbf{x}}_v^i = f_{\Theta_{\text{text}}}(\hat{S}_v^i), \quad (2)$$

where $\mathbf{x}_v^i \in \mathbb{R}^D$ denotes the feature vector of the i -th prompt expert produced by the text encoder. Therefore, for each node v , we can generate four augmented feature vectors in total, each representing one prompt expert accordingly. Notably, we include the raw text attribute as the fourth prompt expert inspired by [42, 54].

4.1.3 Context-aware selector. Given the m initial augmented feature vectors $\{\hat{\mathbf{x}}_v^i\}_{i=1}^m$ of node v , the next question is how to select the most relevant one for each node. As discussed in study [42], different graphs may benefit from different types of augmentation strategies. Similarly, each prompt expert can be seen as a specific perturbation strategy. Therefore, an intuitive solution is to employ an attention mechanism to dynamically integrate the most relevant expert by computing attention coefficients, formulated as:

$$\alpha_v^i = \frac{\exp(\mathbf{W}_1 \hat{\mathbf{x}}_v^i / \tau)}{\sum_{k=1}^m \exp(\mathbf{W}_1 \hat{\mathbf{x}}_v^k / \tau)}, \quad (3)$$

where $\mathbf{W}_1 \in \mathbb{R}^{1 \times D}$ denote the trainable attention weights, and $\alpha_v \in \mathbb{R}^m$ is the attention vector for node v . τ is the temperature parameter used to adjust the sharpness of the attention distribution.

While effective, Eq. (3) neglects the node statistics when integrating various prompt experts. To address this, we introduce the notion of *context prompt*, which describes the functionality of each prompt expert and the node statistics, such as degree information. We report the context prompt for different prompt experts in Appendix 7.2. Let $S_v^{(c,i)}$ denote the context prompt of node v for the i -th prompt expert, we calculate the context-aware attention distribution of node v as follows:

$$\alpha_v^{c,i} = \frac{\exp(f_{\Theta_{\text{text}}}(S_v^{(c,i)}) \mathbf{W}_2 \hat{\mathbf{x}}_v^i / \tau)}{\sum_{k=1}^m \exp(f_{\Theta_{\text{text}}}(S_v^{(c,k)}) \mathbf{W}_2 \hat{\mathbf{x}}_v^k / \tau)}. \quad (4)$$

$\alpha_v^c \in \mathbb{R}^m$ is context-aware attention vector for node v , $\mathbf{W}_2 \in \mathbb{R}^{D \times D}$ is the model weights. Eq. (4) offers the flexibility to incorporate both node-level and prompt expert-level prior knowledge into the attention process. Finally, we integrate the two attention mechanisms and rewrite Eq. (3) as:

$$\alpha_v^i = \frac{\exp((\mathbf{W}_1 \hat{\mathbf{x}}_v^i + f_{\Theta_{\text{text}}}(S_v^{(c,i)}) \mathbf{W}_2 \hat{\mathbf{x}}_v^i / \tau)}{\sum_{k=1}^m \exp((\mathbf{W}_1 \hat{\mathbf{x}}_v^k + f_{\Theta_{\text{text}}}(S_v^{(c,i)}) \mathbf{W}_2 \hat{\mathbf{x}}_v^k / \tau)}, \quad (5)$$

Based on Eq. (5), we obtain the final augmented feature vector $\hat{\mathbf{x}}_v$ of node v as: $\hat{\mathbf{x}}_v = \sum_i \alpha_v^i \hat{\mathbf{x}}_v^i$.

Training objective. To effectively fine-tune the pre-trained smaller LLM ($f_{\Theta_{\text{text}}}$) within our text attribute space, we train $f_{\Theta_{\text{text}}}$ to reconstruct the observed connections. Specifically, given node v and its corresponding row in the adjacency matrix $\mathbf{A}_{v,:}$, we frame the fine-tuning task as a multi-label classification problem. However, directly fine-tuning $f_{\Theta_{\text{text}}}$ on a high-dimensional output space of size $|\mathcal{V}|$ is computationally infeasible. To address this challenge, we employ the extreme multi-label classification (XMC) technique used in GAIN [3] for efficient optimization.

4.2 Collaborative Edge Modifier

Up to this point, we have discussed the process of obtaining augmented feature vectors $\{\hat{\mathbf{x}}_v\}$ using text attributes. Now, we will explore how text attributes can be utilized for effective structure

perturbation. In essence, the aim of edge perturbation is to enhance the diversity between the original and augmented structures while maintaining their structural patterns. In our context, edge perturbation faces two major hurdles: 1) the quadratic growth of the edge search space relative to the graph size, resulting in huge computational costs when querying LLM; 2) the semantic disparity between the text space and observed topological structure, making it suboptimal to rely solely on one of them for edge perturbation.

To tackle this challenge, we propose a text-aware edge perturbation framework, called collaborative edge modifier. As outlined in Algorithm 1 of Appendix 7.3, it leverages the commonalities between both data modalities for edge perturbation. The first stage involves structure-aware top candidate generation. Specifically, we adopt a standard network embedding algorithm (e.g., DeepWalk) to map nodes into a hidden space using only structure data. Subsequently, we assess the similarity between any two nodes based on their network embeddings. For each node v , we then create two disjoint edge sets $\mathcal{E}_v^{\text{spu}}$ and $\mathcal{E}_v^{\text{mis}}$. The former contains the top K least similar edges among the observed links, representing the most spurious connections. The latter comprises top K most similar edges among the disconnected links in the original graph, indicating likely/missing connections.

After obtaining the two candidate sets $\mathcal{E}_v^{\text{spu}}$ and $\mathcal{E}_v^{\text{mis}}$ of node v , the second stage aims to modify the two sets using text attributes. In particular, we define a simple edge modifier prompt to query LLM determining whether two nodes should be connected by interpreting their semantic similarity. The detailed template for this prompt is reported in Section 7.3 of the Appendix. Let $S_{v,u}$ denote the query prompt for nodes v and u , we define the addition and deletion operations below.

4.2.1 Edge deletion. This operation is designed for the potential spurious set $\mathcal{E}_v^{\text{spu}}$. We ask the LLM to estimate the likelihood of each edge $e \in \mathcal{E}_v^{\text{spu}}$ using corresponding query prompt, resulting in an action sequence $a_v^{\text{del}} \in \mathbb{R}^{|\mathcal{E}_v^{\text{spu}}|}$. Here, $a_v^{\text{del}}(i) = 1$ if the LLM believes the two nodes should be disconnected and $a_v^{\text{del}}(i) = 0$ otherwise.

4.2.2 Edge addition. In addition to edge deletion, we also define the addition operation to add potential missing links in $\mathcal{E}_v^{\text{mis}}$. We query the LLM to assess the likelihood of each edge $e \in \mathcal{E}_v^{\text{mis}}$ using the corresponding query prompt, leading to an action sequence $a_v^{\text{add}} \in \mathbb{R}^{|\mathcal{E}_v^{\text{mis}}|}$. $a_v^{\text{add}}(i) = 1$ if the LLM believes the two nodes should be connected; $a_v^{\text{add}}(i) = 0$ otherwise.

Remark. The two stages offer a principled approach to determining the connections between two nodes based on structural and textual aspects, leveraging the commonalities of the two modalities. Furthermore, by focusing on the two action sets $\mathcal{E}_v^{\text{spu}}$ and $\mathcal{E}_v^{\text{mis}}$, the potential query space on the LLM is significantly reduced from the complexity of $O(|\mathcal{V}|^2)$ to $O(K)$. K is a hyperparameter, such as 10 in practice. In summary, the output of the proposed collaborative edge modifier is a set of action sequences $\{a_v | v \in \mathcal{V}\}$, where $a_v = a_v^{\text{del}} || a_v^{\text{add}}$ and $||$ stands for concatenation operation. It is worth noting that this process is conducted “off-the-fly”.

Table 1: Dataset statistics of five text-attributed graphs (TAGs).

Data	# Nodes	# Edges	# Features	# Classes	# Average Text	# Longest Text	# Shortest Text
PubMed	19, 717	44, 338	500	3	1649.25	5732	18
Ogbn-Arxiv	169343	1, 166, 243	128	40	1177.993	9712	136
Books-History	41, 551	400, 125	768	12	1427.397	103130	27
Electronics-Computers	87, 229	808, 310	768	10	492.767	2011	3
Electronics-Photo	48, 362	549, 290	768	12	797.822	32855	5

Table 2: Semi-supervised accuracy results of state-of-the-art GCL methods advanced. "SE" denotes the feature matrix obtained by shallow embedding models. "GIANT" indicates that the text transformation is implemented by the method proposed in [3].

	Method	BGRL	GBT	GraphCL	GCN	GAT
PubMed	SE	80.6±1.0(+3.60%)	79.44±1.31(+5.34%)	79.8±0.5(+2.79%)	77.8±2.9(+3.59%)	78.7±2.3(+0.88%)
	GIANT	82.75±0.28(+0.91%)	81.13±0.82(+3.14%)	81.21±0.22(+1.01%)	79.32±0.45(+1.60%)	78.80±0.52(+0.75%)
	GAugLLM	83.50±0.84	83.68±1.90	82.03±1.74	80.59±0.82	79.39±1.13
Arxiv	SE	71.64±0.12(+2.89%)	70.12±0.18(+1.68%)	70.18±0.17(+1.23%)	71.74 ± 0.29(+2.58%)	71.59±0.38(+2.18%)
	GIANT	73.14±0.14(+0.78%)	70.66±0.07(+0.91%)	70.94±0.06(+0.15%)	73.29±0.10(+0.41%)	74.15±0.05(-1.34%)
	GAugLLM	73.71±0.08	71.3±0.18	71.05±0.14	73.59±0.10	73.15±0.05
Photo	SE	57.98±0.09(+31.8%)	68.56±0.95(+14.0%)	53.21±0.47(+36.3%)	60.31±0.71(+26.7%)	59.03±0.59(+28.6%)
	GIANT	71.65±0.61(+6.64%)	74.65±0.69(+4.72%)	71.40±0.62(+1.55%)	71.83±0.38(+6.35%)	71.44±0.49(+6.27%)
	GAugLLM	76.41±0.64	78.17±0.54	72.51±0.78	76.39±0.62	75.92±0.42
Computers	SE	69.53±0.26(+20.5%)	70.67±0.54(+14.6%)	53.51±0.27(+51.7%)	59.43±0.90(+41.5%)	58.17±0.67(+43.7%)
	GIANT	74.23±0.56(+12.3%)	76.87±0.36(+5.37%)	74.24±0.24(+8.88%)	76.72±0.22(+9.61%)	75.63±0.49(+10.5%)
	GAugLLM	83.8±0.34	82.74±0.45	80.83±0.36	84.10±0.20	83.60±0.18
History	SE	69.84±0.42(+9.29%)	71.62±0.38(+6.27%)	57.26±0.44(+32.2%)	58.14±1.76(+33.1%)	66.39±0.82(+17.65%)
	GIANT	74.16±0.83(+2.93%)	71.89±0.63(+5.90%)	71.14±0.38(+6.45%)	75.99±0.10(+1.87%)	74.67±0.39(+3.44%)
	GAugLLM	76.33±0.88	76.11±0.4	75.73±0.35	77.41±0.32	78.11±0.52

4.3 Graph Contrastive Learning for TAGs

Given the augmented feature matrix \hat{X} and the set of edge perturbations $\{a_v|v \in \mathcal{V}\}$, we can enhance the performance of existing GCL methods by replacing their augmentation strategies with ours. Specifically, prior studies aim to maximize the mutual information between two augmented views, denoted by (A_1, X_1) and (A_2, X_2) . Now we can pre-train a GNN encoder to maximize the mutual information between (A, X) and (\hat{X}, \hat{A}) . Here, X is the feature matrix obtained based on raw text attributes, i.e., $X_v = f_{\theta_{\text{text}}}(S_v)$, and \hat{A} is constructed by random sampling (e.g., with uniform distribution) some actions from $\{a_v|v \in \mathcal{V}\}$ in a simple wise fashion per iteration. Notably, due to the randomness in edge action selection, the augmented views (\hat{X}, \hat{A}) will vary across different iterations, albeit in a consistent manner thanks to the definition of these action sequences. Additionally, as the augmented feature matrix \hat{X} builds upon the original text attributes, it is generally more effective than X and can incentivize the GNN encoder to learn more valuable textual information.

In addition to GCL methods, we have observed that our model could also be extended to enhance the performance of other popular graph generative models (e.g., GraphMAE and S2GAE), as well as standard GNN methods such as GCN and GAT, simply by leveraging the augmented features and structures as input. We empirically analyze this applicability in Section 5.2.

5 Experiments

Throughout the experiments, we aim to address the following research questions. **RQ1:** Can GAugLLM enhance the performance of standard graph contrastive learning methods? **RQ2:** How does GAugLLM perform when applied to other GNN learning scenarios, such as generative pre-training and supervised learning? **RQ3:** How does each component of GAugLLM, i.e., different prompt templates of mixture-of-prompt-experts, attention mechanism, and the collaborative edge modifier, contribute to the performance? **RQ4:** Is the proposed collaborative edge modifier sensitive to the random sampling process in each iteration?

5.1 Experimental Setup

Datasets. We evaluate the proposed GAugLLM framework using five publicly available TAG datasets. These datasets encompass two citation networks, namely PubMed [25] and Ogbn-Arxiv (Arxiv) [12], and three E-commerce datasets extracted from Amazon [22], including Electronics-Computers (Compt), Books-History (Hist), and Electronics-Photography (Photo). For all of these datasets, we adhere to the standard data splits used in prior research. In our experiments, we opt to utilize raw texts directly rather than processed text features so that the textual semantics are preserved. The statistical details of these datasets are outlined in Table 1.

Table 3: Accuracy results of generative methods on TAGs.

	Method	S2GAE	GraphMAE
pubmed	SE	81.66±1.32	81.1±0.4
	GIANT	82.43±0.61	80.16±0.08
	GAugLLM	83.02±0.94	82.98±0.77
arxiv	SE	68.38±0.13	71.75±0.11
	GIANT	70.91±0.09	72.58±0.15
	GAugLLM	71.23±0.08	73.4±0.13
Photo	SE	76.12±0.75	67.49±0.59
	GIANT	77.89±0.48	71.66±0.48
	GAugLLM	76.77±0.22	74.11±0.37
Computers	SE	82.70±0.27	70.90±0.38
	GIANT	84.37±0.42	73.91±0.17
	GAugLLM	84.32±0.36	78.57±0.3
History	SE	71.80±0.82	71.77±0.24
	GIANT	73.56±0.92	75.59±0.62
	GAugLLM	74.84±1.02	76.84±0.33

Baselines. We compare GAugLLM with two textual feature extraction methods. **Shallow Embedding (SE)** is the standard way of generating textual features with shallow embedding models (i.e., Word2vec [20] or Bag-of-Words (BoW) [8]). SE serves as the baseline result of a GCL or GNN algorithm. **Graph Information Aided Node feature exTraction (GIANT)** [3] is a state-of-the-art graph-agnostic feature extraction algorithm tailored for raw texts in graphs. It fine-tunes a language model with self-supervised learning and then fuses the textual embedding with the graph structure information to make predictions.

Experimental Details. We conduct experiments upon three state-of-the-art GCL methods, namely GraphCL [43], BGRL [31], and GBT [1], and two standard GNNs methods: GCN [16] and GAT [33]. For the reproducibility of our experiments, we employ GNN implementations from the PyG [5] package. For the GraphCL, BGRL, and GBT methods, we closely adhere to the procedures outlined in [52]. For each experiment, we run 5 times and report the mean result and the standard deviation. By default, we use the open-sourced LLM model – Mixtral 8*7b version. We provide detailed experimental configurations in Section 7.1 of Appendix.

5.2 Overall Evaluation

To answer **RQ1**, We conduct extensive experiments on five benchmark TAG datasets in standard semi-supervised node classification tasks. Table 2 presents the results for three popular GCL backbones and two standard GNN methods. From these results, we make the following observations.

① **GAugLLM can significantly boost the performance of state-of-the-art GCL methods across all datasets.** In Table 2, GAugLLM consistently outperforms SE and GIANT across all 15 testing scenarios (i.e., columns of BGRL, GBT, and GraphCL). Specifically, while GIANT performs notably better than the SE method due to its utilization of a smaller LLM for transforming text attributes into the feature space, GAugLLM surpasses GIANT in all cases. This superiority can be attributed to the advantage of the proposed

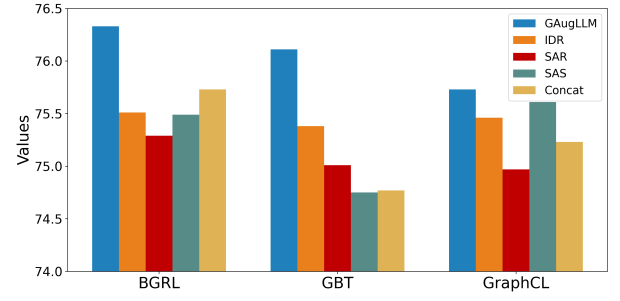


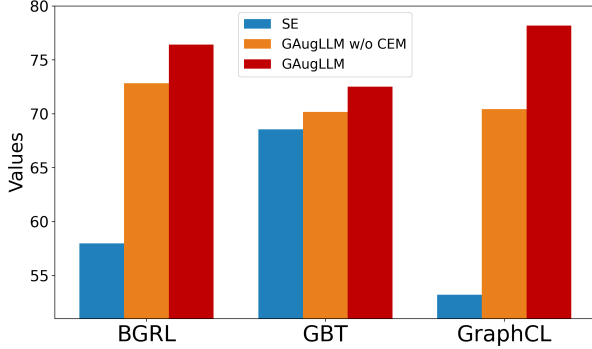
Figure 3: Ablation study of GAugLLM on the History dataset. “IDR”, “SAR”, and “SAS” denote scenarios where we only employ the corresponding prompt expert for feature augmentation. “Concat” means we directly aggregate the hidden representations of all prompt experts as the final output.

mixture-of-prompt-experts, which augments the raw text attributes from diverse aspects. Notably, GAugLLM achieves improvements of +20.5% and +12.3% over SE and GIANT, respectively, when training BGRL on the Computers dataset. Moreover, ② **GCL methods generally outperform standard GNNs when using different textual feature extractors.** This is expected because GCL methods have the potential to learn superior representations and effectively utilize unlabeled data. Our GAugLLM further enhances the learned representations of GCL methods by more effectively encoding textual information into the model. These results demonstrate the effectiveness of GAugLLM in harnessing rich textual features.

In addition to the contrastive learning scenario, we also test the applicability of the learned augmented features on other GNN learning settings, such as generative pre-training and supervised learning (**RQ2**). Table 2 and Table 3 summarize the results on supervised GNN methods and generative pre-training methods, respectively. We observed that ③ **GAugLLM is primarily designed for enhancing GCL, it also significantly improves the performance of standard GNN methods.** In the last two columns of Table 2, GAugLLM consistently outperforms SE in all testing cases and surpasses GIANT in 9 out of 10 testing scenarios. Particularly on the Computers dataset, GAugLLM outperforms the standard GAT and GAT+GIANT by +43.7% and +10.5%, respectively. This strong performance can be attributed to the utilization of a mixture of prompt experts, which enable the incorporation of informative textual semantics enhanced by advanced LLM into model training, thereby benefiting various GNN methods. Furthermore, ④ **simply by substituting the original feature matrix with our augmented feature matrix, the performance of state-of-the-art generative pre-training methods can be further enhanced.** In Table 3, we observe that our method outperforms the SE variant in all cases. Even when compared with a strong baseline method (i.e., GIANT), GAugLLM prevails in 8 out of 10 scenarios, draws in 1, and falls behind in 1 scenarios. These results indicate that our mixture-of-prompt-expert technique can serve as an effective feature learning method in TAGs for graph generative models.

Table 4: Ablation study of GAugLLM w.r.t. attention designs.

	Method	BGRL	GraphCL	GBT
PubMed	w/o context	80.59±2.21	77.17±2.17	79.93±1.35
	w/ context	83.50±0.84	81.68±1.74	83.68±1.90

**Figure 4: Ablation study of GAugLLM w.r.t. collaborative edge modifier on Photo dataset.****Table 5: The impact of different LLMs on GAugLLM.**

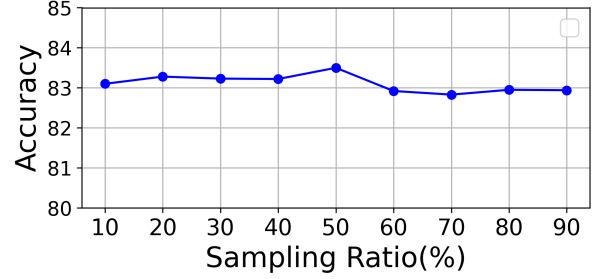
	Backbones	BGRL	GraphCL	GBT
PubMed	Mistral 8*7b	83.50±0.84	81.68±1.74	83.68±1.90
	ChatGPT-3.5	82.62±0.87	80.34±0.65	80.46±0.91
	LLaMA2-13b	81.89±0.75	79.79±2.02	81.93±0.96
History	Mistral 8*7b	76.33±0.88	75.11±0.4	76.11±0.4
	ChatGPT-3.5	75.92±1.02	74.84±0.53	76.67±0.55
	LLaMA2-13b	75.56±0.93	75.26±0.46	75.78±0.39

5.3 Ablation Study

To answer **RQ3**, we conduct a series of ablation studies to verify the contributions of different components in our model design. Specifically, we first test the impact of each individual prompt expert and reports the results in Figure 3. Then, we evaluate the contribution of the context-aware attention design in Eq. (5) in Table 4. Finally, we analyze the influence of the collaborative edge modifier in Figure 4. We make the following observations.

⑤ **GAugLLM benefits from integrating multiple diverse prompt experts for feature augmentation.** As illustrated in Figure 3, GAugLLM consistently outperforms all four variants by a significant margin across three GCL backbones. Notably, even though both GAugLLM and the "Concat" variant utilize all prompt experts as input, GAugLLM outperforms "Concat" in all cases. The possible reason is that different nodes may prefer partial prompt experts for integrating the final augmented features. This comparison verifies our motivation to dynamically combine diverse prompt experts in a learnable way.

⑥ **By incorporating context information, GAugLLM provides an improved approach to integrating multiple prompt experts.** From Table 4, we can see that GAugLLM consistently generates more effective augmented features for state-of-the-art GCL

**Figure 5: Sensitive analysis of GAugLLM w.r.t. the sampling ratio in collaborative edge modifier.**

methods. Notably, when the context-aware attention mechanism in Eq. (5) is not utilized, the performance of GAugLLM significantly declines. This outcome underscores the effectiveness of our proposed context-aware attention strategy in leveraging graph statistics.

⑦ **The proposed collaborative edge modifier scheme could significantly enhance the performance of GAugLLM compared to traditional masking strategies.** As depicted in Figure 4, we observe a substantial performance drop across three GCL methods when using the standard random edge masking for structure perturbation, whereas GAugLLM benefits significantly from the collaborative edge modifier. This comparison underscores the effectiveness of our proposed approach.

In addition to the main components, we also present an ablation study on the impact of different LLM backbones in Table 5. From the table, we observe that ⑧ **the performance gap between open-sourced and closed LLMs on GAugLLM is marginal.** In table 5, we can see that GAugLLM performs generally much better on Mistral 8*7b and ChatGPT-3.5 compared with LLaMA2. More specifically, GAugLLM exhibits competitive or even superior performance on Mistral compared to ChatGPT. Since ChatGPT is a closed-sourced tool, this comparison validates the potential impact of our model in real-world scenarios as one can use the open-sourced LLM (i.e., Mistral 8*7b) without sacrificing performance.

5.4 Sensitive Analysis

To answer **RQ4**, we investigate the impact of different random sampling processes on GAugLLM. Specifically, we varied the sampling probability of the sample function in the collaborative edge modifier from 10% to 90% with a step size of 10%. Figure 5 reports the results. We observe that ⑨ **The proposed collaborative edge modifier is robust to changes in the sampling ratio.** From Figure 5, we can see that GAugLLM performs the best when the sampling ratio is 50%. We note that GAugLLM delivers very consistent accuracies across a wide range of sampling ratios, showing stability as the ratio increases from 10% to 90%, which would be desirable in real-world applications.

6 Conclusion and Future Work

In this work, we delve into graph contrastive learning for text-attributed graphs (TAGs). While extensive endeavors have been proposed recently aimed at enhancing contrastive learning on graphs, these approaches are limited in harnessing the rich text attributes.

This is because they simply utilize a shallow embedding model, such as word2vec, to transform the text attributes into feature space during pre-processing. To address this shortfall, we present GAugLLM, a pioneering graph augmentation framework that harnesses advanced LLMs for feature-level and structure-level augmentations. GAugLLM comprises two pivotal modules: the mixture-of-prompt-expert and collaborative edge modifier. The former dynamically integrates multiple prompt experts, each perturbing raw text attributes via prompt engineering, into the feature space for effective augmentation. The latter focuses on modifying connections in the original graph, either by deletion or addition, leveraging both structural and textual commonalities. Building upon these novel techniques, GAugLLM directly enhances the performance of leading contrastive learning methods (e.g., BGRL, GraphCL, and GBT). Interestingly, empirical findings indicate that GAugLLM can be readily applied to other GNN learning scenarios, including generative pre-training and supervised training. We hope our GAugLLM and experimental findings can motivate and pave the path for future research in leveraging LLMs for text-attributed graphs. In the future, we plan to extend GAugLLM to other graph-related tasks, such as graph generation, graph structure learning [51] and their applications in other domains.

Acknowledgments

The work is, in part, supported by Shanghai Frontiers Science Center of Artificial Intelligence and Deep Learning and the Startup fund at NYU Shanghai.

References

- [1] Piotr Bielak, Tomasz Kajdanowicz, and Nitesh V Chawla. 2022. Graph barlow twins: A self-supervised representation learning framework for graphs. *Knowledge-Based Systems* 256 (2022), 109631.
- [2] Zhikai Chen, Haitao Mao, Hang Li, Wei Jin, Hongzhi Wen, Xiaochi Wei, Shuaiqiang Wang, Dawei Yin, Wenqi Fan, Hui Liu, et al. 2023. Exploring the potential of large language models (llms) in learning on graphs. *arXiv preprint arXiv:2307.03393* (2023).
- [3] Eli Chien, Wei-Cheng Chang, Cho-Jui Hsieh, Hsiang-Fu Yu, Jiong Zhang, Olga Milenkovic, and Inderjit S Dhillon. 2022. Node feature extraction by self-supervised multi-scale neighborhood prediction. In *International Conference on Learning Representations*.
- [4] Kaize Ding, Zhe Xu, Hanghang Tong, and Huan Liu. 2022. Data augmentation for deep graph learning: A survey. *ACM SIGKDD Explorations Newsletter* 24, 2 (2022), 61–77.
- [5] Matthias Fey and Jan Eric Lenssen. 2019. Fast graph representation learning with PyTorch Geometric. *arXiv preprint arXiv:1903.02428* (2019).
- [6] Xumeng Gong, Cheng Yang, and Chuan Shi. 2023. MA-GCL: Model Augmentation Tricks for Graph Contrastive Learning. In *AAAI*.
- [7] Xumeng Gong, Cheng Yang, and Chuan Shi. 2023. Ma-gcl: Model augmentation tricks for graph contrastive learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 37. 4284–4292.
- [8] Zellig S Harris. 1954. Distributional structure. *Word* 10, 2-3 (1954), 146–162.
- [9] Kaveh Hassani and Amir Hosein Khasahmadi. 2020. Contrastive multi-view representation learning on graphs. In *ICML*. PMLR, 4116–4126.
- [10] Xiaoxin He, Xavier Bresson, Thomas Laurent, and Bryan Hooi. 2023. Explanations as Features: LLM-Based Features for Text-Attributed Graphs. *arXiv preprint arXiv:2305.19523* (2023).
- [11] Zhenyu Hou, Xiao Liu, Yukuo Cen, Yuxiao Dong, Hongxia Yang, Chunjie Wang, and Jie Tang. 2022. Graphmae: Self-supervised masked graph autoencoders. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 594–604.
- [12] Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. 2020. Open graph benchmark: Datasets for machine learning on graphs. *Advances in neural information processing systems* 33 (2020), 22118–22133.
- [13] Ziniu Hu, Yuxiao Dong, Kuansan Wang, Kai-Wei Chang, and Yizhou Sun. 2020. Gpt-gnn: Generative pre-training of graph neural networks. In *KDD*. 1857–1867.
- [14] Xiao Huang, Jundong Li, and Xia Hu. 2017. Label informed attributed network embedding. In *Proceedings of the tenth ACM international conference on web search and data mining*. 731–739.
- [15] Wei Jin, Tyler Derr, Haochen Liu, Yiqi Wang, Suhang Wang, Zitao Liu, and Jiliang Tang. 2020. Self-supervised learning on graphs: Deep insights and new direction. *arXiv preprint arXiv:2006.10141* (2020).
- [16] Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907* (2016).
- [17] Namkyeong Lee, Junseok Lee, and Chanyoung Park. 2022. Augmentation-free self-supervised learning on graphs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 36. 7372–7380.
- [18] Lizi Liao, Xiangnan He, Hanwang Zhang, and Tat-Seng Chua. 2018. Attributed social network embedding. *IEEE Transactions on Knowledge and Data Engineering* 30, 12 (2018), 2257–2270.
- [19] Yixin Liu, Ming Jin, Shirui Pan, Chuan Zhou, Yu Zheng, Feng Xia, and S Yu Philip. 2022. Graph self-supervised learning: A survey. *IEEE Transactions on Knowledge and Data Engineering* 35, 6 (2022), 5879–5900.
- [20] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).
- [21] Bonan Min, Hayley Ross, Elior Sulem, Amir Pouran Ben Veyseh, Thien Huu Nguyen, Oscar Sainz, Eneko Agirre, Ilana Heintz, and Dan Roth. 2023. Recent advances in natural language processing via large pre-trained language models: A survey. *Comput. Surveys* 56, 2 (2023), 1–40.
- [22] Jianmo Ni, Jiacheng Li, and Julian McAuley. 2019. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*. 188–197.
- [23] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748* (2018).
- [24] Jiezhong Qiu, Qibin Chen, Yuxiao Dong, Jing Zhang, Hongxia Yang, Ming Ding, Kuansan Wang, and Jie Tang. 2020. Gcc: Graph contrastive coding for graph neural network pre-training. In *KDD*. 1150–1160.
- [25] Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. 2008. Collective classification in network data. *AI magazine* 29, 3 (2008), 93–93.
- [26] Yucheng Shi, Yushun Dong, Qiaoyu Tan, Jundong Li, and Ninghao Liu. 2023. Gigamae: Generalizable graph masked autoencoder via collaborative latent space reconstruction. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*. ACM, 2259–2269.
- [27] Susheel Suresh, Pan Li, Cong Hao, and Jennifer Neville. 2021. Adversarial graph augmentation to improve graph contrastive learning. *Advances in Neural Information Processing Systems* 34 (2021), 15920–15933.
- [28] Qiaoyu Tan, Ninghao Liu, and Xia Hu. 2019. Deep representation learning for social network analysis. *Frontiers in big Data* 2 (2019), 2.
- [29] Qiaoyu Tan, Ninghao Liu, Xiao Huang, Soo-Hyun Choi, Li Li, Rui Chen, and Xia Hu. 2023. S2GAE: Self-Supervised Graph Autoencoders are Generalizable Learners with Graph Masking. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining*. 787–795.
- [30] Qiaoyu Tan, Xin Zhang, Xiao Huang, Hao Chen, Jundong Li, and Xia Hu. 2023. Collaborative graph neural networks for attributed network embedding. *IEEE Transactions on Knowledge and Data Engineering* (2023).
- [31] Shantanu Thakoor, Corentin Tallec, Mohammad Gheshlaghi Azar, Mehdi Azabou, Eva L Dyer, Remi Munos, Petar Veličković, and Michal Valko. 2021. Large-Scale Representation Learning on Graphs via Bootstrapping.
- [32] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NeurIPS*. 5998–6008.
- [33] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. (2018).
- [34] Petar Veličković, William Fedus, William L Hamilton, Pietro Liò, Yoshua Bengio, and R Devon Hjelm. 2018. Deep graph infomax. *arXiv preprint arXiv:1809.10341* (2018).
- [35] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. 2020. A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems* 32, 1 (2020), 4–24.
- [36] Jun Xia, Lirong Wu, Jintao Chen, Bozhen Hu, and Stan Z Li. 2022. Simgrace: A simple framework for graph contrastive learning without data augmentation. In *Proceedings of the ACM Web Conference 2022*. 1070–1079.
- [37] Yaochen Xie, Zhao Xu, Jintun Zhang, Zhengyang Wang, and Shuiwang Ji. 2022. Self-supervised learning of graph neural networks: A unified review. *IEEE transactions on pattern analysis and machine intelligence* 45, 2 (2022), 2412–2429.
- [38] Dongkuan Xu, Wei Cheng, Dongsheng Luo, Haifeng Chen, and Xiang Zhang. 2021. InfoGCL: Information-Aware Graph Contrastive Learning. *NeurIPS* 34 (2021).
- [39] Hao Yan, Chaozhao Li, Ruosong Long, Chao Yan, Jianan Zhao, Wenwen Zhuang, Jun Yin, Peiyan Zhang, Weihao Han, Hao Sun, et al. 2023. A Comprehensive Study on Text-attributed Graphs: Benchmarking and Rethinking. In *Thirty-seventh*

Conference on Neural Information Processing Systems Datasets and Benchmarks Track.

- [40] Jingfeng Yang, Hongye Jin, Ruixiang Tang, Xiaotian Han, Qizhang Feng, Haoming Jiang, Bing Yin, and Xia Hu. 2023. Harnessing the power of llms in practice: A survey on chatgpt and beyond. *arXiv preprint arXiv:2304.13712* (2023).
- [41] Yihang Yin, Qingzhong Wang, Siyu Huang, Haoyi Xiong, and Xiang Zhang. 2022. Autogcl: Automated graph contrastive learning via learnable view generators. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 36. 8892–8900.
- [42] Yuning You, Tianlong Chen, Yang Shen, and Zhangyang Wang. 2021. Graph contrastive learning automated. In *International Conference on Machine Learning*. PMLR, 12121–12132.
- [43] Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. 2020. Graph contrastive learning with augmentations. *Advances in neural information processing systems* 33 (2020), 5812–5823.
- [44] Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny. 2021. Barlow twins: Self-supervised learning via redundancy reduction. In *International Conference on Machine Learning*. PMLR, 12310–12320.
- [45] Hengrui Zhang, Qitian Wu, Junchi Yan, David Wipf, and Philip S Yu. 2021. From canonical correlation analysis to self-supervised graph neural networks. *Advances in Neural Information Processing Systems* 34 (2021), 76–89.
- [46] Sixiao Zhang, Hongxu Chen, Haoran Yang, Xiangguo Sun, Philip S Yu, and Guandong Xu. 2022. Graph Masked Autoencoders with Transformers. *arXiv e-prints* (2022), arXiv–2202.
- [47] Xin Zhang, Qiaoyu Tan, Xiao Huang, and Bo Li. 2024. Graph contrastive learning with personalized augmentation. *IEEE Transactions on Knowledge and Data Engineering* (2024).
- [48] Jianan Zhao, Meng Qu, Chaozhuo Li, Hao Yan, Qian Liu, Rui Li, Xing Xie, and Jian Tang. 2023. Learning on Large-scale Text-attributed Graphs via Variational Inference. In *Proceedings of the International Conference on Learning Representations (ICLR)*. International Conference on Learning Representations.
- [49] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223* (2023).
- [50] Jie Zhou, Ganqu Cui, Shengding Hu, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. 2020. Graph neural networks: A review of methods and applications. *AI open* 1 (2020), 57–81.
- [51] Zhiyao Zhou, Sheng Zhou, Bochao Mao, Xuanyi Zhou, Jiawei Chen, Qiaoyu Tan, Daochen Zha, Yan Feng, Chun Chen, and Can Wang. 2024. OpenGSL: A Comprehensive Benchmark for Graph Structure Learning. *Advances in Neural Information Processing Systems* 36 (2024).
- [52] Yanqiao Zhu, Yichen Xu, Qiang Liu, and Shu Wu. 2021. An empirical study of graph contrastive learning. *arXiv preprint arXiv:2109.01116* (2021).
- [53] Yanqiao Zhu, Yichen Xu, Feng Yu, Qiang Liu, Shu Wu, and Liang Wang. 2020. Deep graph contrastive representation learning. *arXiv preprint arXiv:2006.04131* (2020).
- [54] Yanqiao Zhu, Yichen Xu, Feng Yu, Qiang Liu, Shu Wu, and Liang Wang. 2021. Graph contrastive learning with adaptive augmentation. In *Proceedings of the Web Conference 2021*. 2069–2080.

7 Appendix

7.1 Experimental Configurations

For baselines, we report the baseline model results based on their provided codes with official settings or results reported in previous researches. If their settings or results are not available, we conduct a hyper-parameter search. Table 6 is the hyper-parameters for our own method GAugLLM in GCLs. Table 7 is the default setting for mix-of-prompt-expert module. One exception is that the epoch for arxiv is set to 1.

Table 6: Configurations for each dataset on GCLs

	Setting	BGRL	GraphCL	GBT
PubMed	lr	5e-4	1e-3	2e-3
	encoder_layer	512	512	512
	epoch	8000	1000	1000
Arxiv	lr	1e-2	1e-3	1e-3
	encoder_layer	512	256	256
	epoch	1000	1000	1000
History	lr	1e-3	5e-4	1e-4
	encoder_layer	512	256	256
	epoch	10000	5000	5000
Photo	lr	1e-3	7e-4	5e-4
	encoder_layer	512	256	256
	epoch	10000	5000	5000
Computers	lr	1e-3	1e-3	1e-3
	encoder_layer	512	256	256
	epoch	10000	5000	5000

Table 7: Default setting for mix-of-prompt-expert

Default Setting	
hidden_dropout_prob	0.05
batch_size	32
learning_rate	6e-5
epoch	5/2/1
attention temperature	0.2

7.2 Prompt Expert Design

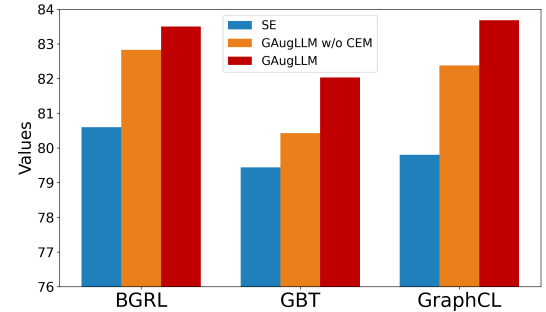


Figure 6: Ablation study of GAugLLM w.r.t. collaborative edge modifier on PubMed dataset.

Given a node v and its textual attribute S_v , traditional GCL methods typically create an augmented feature vector $\hat{\mathbf{x}}_v$ using purely stochastic functions, i.e., $\hat{\mathbf{x}}_v = \tau_f(\mathbf{x}_v) = \tau_f(\text{Emb}(S_v))$. However, this approach only introduces perturbations within the numerical space transformed by the $\text{Emb}(\cdot)$ module, which cannot effectively manipulate the original input textual attribute. To overcome this limitation, we propose to use LLMs to directly perturb the input text S_v and obtain an augmented textual attribute \hat{S}_v through three prompt templates (refer to Figure 7 (left)) outlined below.

Table 8: Context Prompt templates for different Experts.

Expert	Prompt Template
RAW	This is the original text of this node. The degree of this node is ... (Node information)
IDR	This is the explanation for classification based on the original text of this node. The degree of this node is ... We consider nodes with degree more than ... as head nodes. Head nodes have rich structure information in their connections with neighbor nodes.
SAR	This is the explanation for classification based on the original text with the understanding of its neighboring nodes. The degree of this node is ... We consider nodes with degree less than ... as tail nodes. Tail nodes have sparse structure information in their connections with neighbor nodes.
SAS	This is the summarization of the original text with the understanding of its neighboring nodes. The degree of this node is ... We consider degree less than ... and more than as mid nodes.

Structure-Aware Summarization (SAS). Let $S_v^N = \{S_u | v \in N_v\}$ represent the textual attribute set of node v 's neighbors. The idea of SAS is to query the LLM to create a summary of the anchor node v by comprehending the semantic information from both its neighbors and itself. Specifically, for each node v , we construct a prompt that incorporates the textual attributes of the anchor node and its neighbors, denoted as $\{S_v, S_v^N\}$, along with an instruction for revising its textual attribute. The general prompt format is illustrated in the left panel of Figure 7 (left). Finally, we employ these summarized textual attributes to represent the augmented attribute \hat{S}_v .

Independent Reasoning (IDR). In contrast to SAS, which concentrates on text summarization, IDR adopts an "open-ended" approach when querying the LLM. This entails instructing the model to make predictions across potential categories and to provide explanations for its decisions. The underlying philosophy here is that such a reasoning task will prompt the LLM to comprehend the semantic significance of the input textual attribute at a higher level, with an emphasis on the most vital and relevant factors [10]. Following this principle, for each node v , we generate a prompt that takes the textual attribute of the anchor node as input and instructs the LLM to predict the category of this node and provide explanations. The general prompt format is illustrated in the middle panel of Figure 7 (left). We utilize the prediction and explanations to represent the augmented attribute \hat{S}_v .

Structure-Aware Reasoning (SAR). Taking a step beyond IDR, SAR integrates structural information into the reasoning process. The rationale for this lies in the notion that connected nodes can aid in deducing the topic of the anchor node. Specifically, for each node v , we devise a prompt that encompasses the textual attributes of the anchor node S_v and its neighbors S_v^N , along with an open-ended query concerning the potential category of the node. The general prompt format is given in the right panel of Figure 7 (left). Similar to IDR, we employ the prediction and explanations to denote the augmented attribute \hat{S}_v .

To reduce the query overhead of ChatGPT, we randomly sample 10 neighbors for each anchor node in structure-aware prompts (i.e., SAS and SAR) in our experiments.

7.3 Collaborative Edge Modifier

This section is dedicated to elucidating the algorithm behind the Collaborative Edge Modifier. The algorithm operates in two distinct phases. Initially, in the first phase, we deploy a Language Model (LLM) to generate two sets of edges. Subsequently, in the second phase, we proceed to either incorporate or eliminate portions of the graph structure based on the edges produced in the initial phase. For those interested in the finer details, the pseudocode for this process is provided in Algorithm 1.

Algorithm 1 Collaborative Edge Modifier

```

1: procedure STRUCTURE_AUGMENTATION( $G, v, A, LLM$ )
2:   // First stage: structure-aware top candidate generation.
3:    $N_v \leftarrow \text{ConnectedNodes}(v)$ 
4:    $\bar{N}_v \leftarrow \text{DisconnectedNodes}(v)$ 
5:
6:   //network embedding algorithm
7:    $\mathcal{E}_v^{\text{spu}} \leftarrow \text{SelectTopK}(N_v, 10, \text{descending})$ 
8:    $\mathcal{E}_v^{\text{mis}} \leftarrow \text{SelectTopK}(\bar{N}_v, 10, \text{ascending})$ 
9:   prompt_connect  $\leftarrow \text{CreatePrompt}(v, \mathcal{E}_v^{\text{spu}})$ 
10:  prompt_disconnect  $\leftarrow \text{CreatePrompt}(v, \mathcal{E}_v^{\text{mis}})$ 
11:  candidates_discard  $\leftarrow \text{LLM}(\text{prompt\_connect})$ 
12:  candidates_add  $\leftarrow \text{LLM}(\text{prompt\_disconnect})$ 
13:
14:  //Second Stage: Update adjacency matrix based on LLM decisions with a certain accept rate.
15:  for each epoch in contrastive training do
16:    for each node  $u$  in  $v$  do
17:      edges_add  $\leftarrow \text{RandomSelect}(u, \text{candidates\_add}, 0.5)$ 
18:      edges_discard  $\leftarrow \text{RandomSelect}$ 
19:       $(u, \text{candidates\_discard}, 0.5)$ 
20:      Update  $\hat{A}[v][u]$  with edges_add and edges_discard
21:    end for
22:    Use  $A$  and  $\hat{A}$  for contrastive training
23:  end for
24: end procedure

```

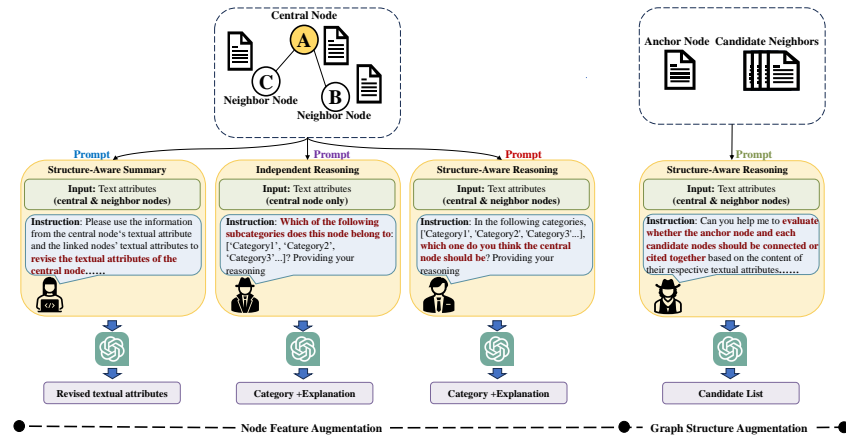


Figure 7: LLM-as-GraphAugmentor. Left: LLMs are employed to perturb node features by influencing the input textual attributes. Right: LLMs are utilized to create new graph structures by modifying and adding edges between nodes.