

Article

CoACL: Coupled Augmentation for Contrastive Learning on Text-Attributed Graphs Under Semantic Supervision from Large Language Models

Hailun Kang ^{1,2} , Kexin Zhao ^{1,2,*}, Shuying Du ^{1,2}, Xi Wu ^{1,2}, Zhong Zhang ^{1,2}, Jiquan Peng ^{1,2,3} ,
Zhongping Zhang ^{1,2,3} and Jibing Gong ^{1,2,3}

¹ School of Artificial Intelligence (School of Software), Yanshan University, Qinhuangdao 066004, China; kanghailun1998@163.com (H.K.)

² Hebei Key Laboratory of Computer Virtual Technology and System Integration, Yanshan University, Qinhuangdao 066004, China

³ The Shenzhen Research Institute of Yanshan University, Shenzhen 518063, China

* Correspondence: kexinz858@gmail.com

Abstract

Text-attributed graphs (TAGs) couple graph topology with node-level text, but real data often contain spurious edges, missing links, and text–structure mismatch that destabilize learning under scarce labels. We propose CoACL (Coupled Augmentation for Contrastive Learning), a framework that uses LLM semantic supervision to denoise structural and textual information and alleviate data sparsity. CoACL first prunes the candidate edge space using structural similarity and then queries an LLM to discard suspicious edges and confirm plausible links, yielding semantically consistent positive and negative pairs. We further introduce keyword-focused text augmentations and learn coupled representations by optimizing a joint text–graph contrastive objective guided by semantics. Experiment results on Cora, PubMed, and the Open Graph Benchmark Arxiv dataset (OGBN-Arxiv) show that CoACL consistently outperforms strong baselines and yields up to 7.1% absolute improvement in node classification accuracy, with the largest gains in low-label regimes. By constraining LLM evaluation to similarity-based candidates, CoACL targets neighborhood-level noise with controlled cost.

Keywords: text-attributed graphs; graph contrastive learning; large language models; semantics-aware augmentation; noise correction



Academic Editor: Arkaitz Zubiaga

Received: 17 January 2026

Revised: 11 February 2026

Accepted: 14 February 2026

Published: 16 February 2026

Copyright: © 2026 by the authors.

Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and

conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

1. Introduction

Graphs are a fundamental form of structured data and arise in a wide range of applications. In many real-world graphs, nodes are associated with textual attributes. For example, ogbn-products from the Open Graph Benchmark (OGB) is an e-commerce product graph in which each node represents a product and its textual description serves as the node attribute. In graph learning, graphs equipped with node-level text are commonly referred to as Text-Attributed Graphs (TAGs) [1]. Recent evidence also suggests that directly applying Large Language Models (LLMs) to graph analysis remains challenging, motivating principled integration of LLM semantics with graph learning [2].

Text-Attributed Graph Learning (TAGL), which lies at the intersection of Graph Representation Learning (GRL) and Natural Language Processing (NLP), has been widely

adopted in real-world applications. TAGL aims to jointly exploit graph topology and textual semantics, enabling more expressive modeling than unimodal graph-only or text-only approaches. In social network analysis, incorporating user relations and textual content (e.g., tweets and comments) is effective for tasks such as fake-account (bot) detection and community detection [3]. In recommender systems, jointly modeling product descriptions and user–item interaction graphs can alleviate data sparsity and improve recommendation quality [4,5]. In biomedical knowledge discovery, integrating scientific text with drug–target interaction networks has facilitated drug repositioning prediction [6]. These applications suggest that topology and semantics provide complementary signals in TAGL.

From a modeling perspective, textual attributes provide node-level semantic cues that can be encoded by LLMs, while graph structure captures relational proximity and homophily that can be modeled by Graph Neural Networks (GNNs) via message passing. Beyond local propagation, recent studies also explore structural information measures to characterize global organization and uncertainty in graphs under weak or noisy supervision [7]. Motivated by the complementarity between semantic and structural information, recent TAGL methods increasingly integrate LLMs and GNNs in a co-training paradigm. However, in real-world TAGs, noisy edges and ambiguous text can lead to misalignment between semantic and structural signals, yielding unreliable supervision and degraded representations.

Despite substantial progress, the effectiveness of TAGL is still limited by two pervasive challenges in real-world scenarios: noise interference and data sparsity. Existing TAGL methods improve downstream performance via joint modeling of text and topology [8], while topology-aware contrastive learning further treats textual and structural views as complementary for representation learning [9]. First, many contrastive designs still underuse contextualized text semantics, relying on bag-of-words signals or shallow text encoders that are insufficient for fine-grained semantic discrimination. Second, common structural perturbations, such as random edge masking or topology-only edits, are often imposed without semantic constraints, undermining text–structure consistency. Third, connectivity-driven negative sampling can misclassify semantically related nodes as negatives, inducing false negatives and weakening separability, which in turn hampers stable contrastive optimization [10,11].

To address these limitations, we propose CoACL (Coupled Augmentation for Contrastive Learning on Text-Attributed Graphs), an LLM-supervised coupled augmentation framework that denoises both graph structure and node text for robust contrastive learning. Specifically, CoACL employs LLM-guided semantic supervision to assess and refine noisy relations and to construct coupled text–structure augmentations with consistent positive/negative semantics. By aligning semantic-aware augmentations across text and graph views, CoACL improves the robustness of representation learning under noisy structures, ambiguous textual attributes, and sparse supervision.

Different from prior LLM-enhanced text-attributed graph methods that primarily treat large language models as feature generators or textual encoders, CoACL leverages LLMs as semantic validators to supervise coupled augmentation across structural and textual views. Instead of directly injecting LLM-generated features into graph models, we use LLM reasoning to assess the semantic plausibility of candidate relations and guide the construction of consistent positive and negative samples. This semantic supervision enables principled denoising and alignment between text and structure, which is fundamentally different from existing “LLM-as-feature-enhancer” paradigms.

Our contributions are threefold: (i) We propose CoACL, a coupled augmentation framework for contrastive learning on text-attributed graphs. (ii) We introduce LLM semantic supervision to denoise both structural relations and textual information, enabling

coupled text–structure views with consistent semantics. (iii) We demonstrate that CoACL effectively alleviates data sparsity and improves robustness under noise through extensive experiments on three benchmark TAG datasets, achieving accuracy improvements of up to 7.1% on node classification.

2. Related Work

2.1. Text-Attributed Graph Learning Method

Graphs are a fundamental form of structured data with extensive real-world applications. In many graph-structured domains, nodes are often associated with rich textual attributes. For instance, in e-commerce scenarios [12], each node represents a product, and its textual description serves as the node’s attribute. Such graphs, where textual content characterizes nodes, are known as TAGs [13].

In our experiments, we compare CoACL with a broad and representative set of graph learning baselines. Specifically, we include topology-driven graph contrastive learning methods such as GraphCL and bootstrapped representation learning approaches like BGRL, as well as recent text-attributed graph models, including GIANT and PLM-based variants (LLM-plain and LLM-tune), which leverage a pre-trained language model for enhanced text encoding. In addition, to account for early and widely used text modeling strategies, we also consider classical statistical text representations such as Bag-of-Words (BoW) [14], which transform raw textual node attributes into fixed-dimensional vectors based on word occurrence statistics. Together, these methods cover classical text-based baselines, topology-centric self-supervised learning, and modern LLM-enhanced approaches, providing a comprehensive baseline set for evaluating the effectiveness of semantics-aware coupled augmentation.

Text-attributed graph learning lies at the intersection of graph representation learning and natural language processing. By jointly modeling topological structures and semantic information, it provides a new paradigm for understanding complex systems. For example, in social network analysis, integrating user interaction graphs with textual content has been shown effective for fake account and social bot detection under limited supervision [3,15,16]. In recommender systems, combining product descriptions with user–item interaction graphs mitigates data sparsity [4]. In biomedical applications, integrating scientific literature with drug–target networks has enabled accurate drug repurposing prediction, achieving substantial performance gains [6].

These studies highlight the power of TAGs in leveraging the complementarity between structural and semantic information. Textual data encode rich contextual semantics effectively modeled by LLMs, while GNNs capture relational dependencies through message passing. Recently, joint training paradigms combining LLMs and GNNs have emerged as a dominant trend, driving significant advances in text-attributed graph learning.

2.2. Graph Contrastive Learning

Graph contrastive learning (GCL) has emerged as an effective self-supervised learning paradigm, demonstrating notable advantages in mitigating noise interference and label scarcity in graph data. GraphCL [8] was among the first to introduce contrastive learning concepts from the vision domain into GNNs, employing perturbation-based view generation to perform contrastive training. This approach significantly improved the generalization capability of graph representations and laid the foundation for subsequent GCL research, which is typically evaluated on citation network benchmarks such as Cora using standard GNN backbones, including GCN [17], GAT [18], and GraphSAGE [19].

As research progressed, it became increasingly evident that while randomly perturbed views enhance robustness, they may also introduce noise or disrupt the intrinsic graph

structure, thereby degrading representation quality. Consequently, recent studies have focused on semantic consistency-aware data augmentation strategies. For instance, Con-GCL [20] first introduced structural consistency constraints to enforce consistency between augmented views and preserve topological similarity to the original graph.

In addition, learning-based graph augmentation methods have gained attention, including adversarial augmentation [21] and graph structure learning [22], which have shown promising results in enhancing model robustness. However, these approaches often fall short of directly augmenting the inherent textual attributes of TAGs and primarily focus on unimodal graph structure augmentation, isolating the language and graph models. This separation limits knowledge transfer between modalities and hampers joint representation learning.

To overcome these limitations, recent studies have explored cross-modal contrastive learning for TAGs, introducing LLMs to enable deeper integration of semantic reasoning and structural information. Specifically, the team led by Hooi [1] proposed leveraging LLMs to generate fine-grained explanatory texts, which are then used as node features for GNNs, with cross-modal alignment optimized via contrastive learning. Similarly, LLMs were used as a textual attribute enhancer [23], encoding both the augmented and original attributes to serve as input features for GNNs.

However, these approaches still maintain a clear separation between textual and graph models, restricting mutual knowledge exchange. As a result, they struggle to fully fuse semantic and structural information. Additionally, challenges remain in optimizing the quality of negative samples, often leading to insufficient contrast between positive and negative pairs, which ultimately hinders learning effectiveness.

2.3. LLM Prompt Tuning

Prompt tuning has emerged as a parameter-efficient paradigm for adapting pre-trained language models (PLMs) to downstream tasks without full fine-tuning [24]. Existing approaches can be broadly categorized into *discrete prompting*, which reformulates tasks using manually or automatically designed natural language templates [25,26], and *continuous prompting*, which represents task instructions as learnable vectors optimized via backpropagation while keeping PLMs frozen [27–30]. These techniques have shown strong performance in few-shot and low-resource settings, but they primarily focus on input-level task adaptation and do not explicitly address representation-level alignment or noise-robust supervision for graph-structured data. In contrast, rather than tuning prompts, our work leverages LLMs as *semantic supervisors* to validate candidate relations and guide coupled augmentation for text-attributed graph denoising and contrastive alignment, enabling more structured and robust representation learning. Representative prompt templates and decoding/parsing rules used for semantic supervision are provided in Appendix A for reproducibility.

3. Methodology

Real-world TAGs often contain structural noise, attribute noise, and limited labeled data, which jointly hinder effective TAGL. To address these challenges, we propose CoACL, which leverages an LLM-guided coupled-augmentation mechanism to identify reliable edges and semantically relevant links, enforce attribute–structure consistency, and enable joint text–graph representation learning. In the following, we describe CoACL in detail.

The notations in Table 1 are consistently used throughout this section, ensuring that all mathematical expressions in the following subsections are self-contained and unambiguous.

Table 1. Notation summary.

Notation	Definition
S_n, S_n^+, S_n^-	Textual embeddings of node v_n in the original, positive, and negative views.
G_n, G_n^+, G_n^-	Graph embeddings of node v_n in the original, positive, and negative views.
$f_{\text{cls}}^{(g)}$	Classification heads for graph modality.
$\phi(a, b)$	Cosine similarity between vectors a and b , i.e., $\phi(a, b) = \frac{a^\top b}{\ a\ \ b\ }$.
τ	Temperature coefficient controlling contrastive sharpness.
$\mathcal{B}$	Mini-batch of nodes used for contrastive learning.
N, C	Number of nodes and number of classes.
λ, γ	Weighting factors for text and graph coupling losses.
$\mathcal{L}_T, \mathcal{L}_G$	Total text-side and graph-side loss functions.
$\mathcal{L}^{\text{pos}}, \mathcal{L}^{\text{neg}}$	Positive and negative contrastive loss terms.

3.1. CoACL Model

The CoACL model contains three core modules: noise correction module, text–structure coupled augmentation module, and joint contrastive learning. The overall architecture is shown in Figure 1. First, the noise correction module utilizes LLMs to evaluate the semantic relevance of node pairs in the graph structure, mining the potential connections and pruning redundant edges to improve the quality of the graph structure. Second, the coupled augmentation module fuses the embedding representations of different candidate node sets under semantic constraints, constructs high-quality positive and negative samples, emphasizes the coupling interaction between textual attribute and graph structure, and mitigates the semantic drift in structure perturbation. Finally, the joint contrastive learning module minimizes the positive inner-sample distance and maximizes the negative inner-sample through contrastive learning, realizes seamless cross-modal contrastive learning, and outputs node classification prediction results. Overall, the CoACL model significantly improves the learning performance of TAGs in complex scenarios from three aspects: text augmentation, structure optimization, and negative sample construction.

3.2. Description of the CoACL Architecture

3.2.1. Noise Correction Based on Large Language Models

Real-world TAGs often contain spurious edges and missing links. Using structure alone is unreliable, while directly scoring all node pairs with an LLM is prohibitively expensive. We therefore first narrow the candidate space via structural similarity and then let an LLM validate ambiguous relations with textual evidence.

We compute a structure-based representation for each node and the cosine similarity between nodes in Equation (1):

$$\begin{aligned} \mathbf{h}_n &= \text{DeepWalk}(n), \\ \phi(\mathbf{h}_n, \mathbf{h}_m) &= \frac{\mathbf{h}_n \cdot \mathbf{h}_m}{\|\mathbf{h}_n\| \|\mathbf{h}_m\|}. \end{aligned} \quad (1)$$

where $\mathbf{h}_n$ is the structural embedding of node n and $\phi(\mathbf{h}_n, \mathbf{h}_m)$ measures structural similarity between nodes n and m .

Let $\mathcal{N}_n^{\text{nei}}$ and $\mathcal{N}_n^{\text{non}}$ denote the neighbor and non-neighbor sets of node n , respectively. Based on connectivity and the similarity score $\phi(\cdot, \cdot)$, we construct four non-overlapping candidate sets in Equation (2):

$$\begin{aligned}
\mathcal{N}_n^{\text{pos}} &= \text{SelectTopK}_{m \in \mathcal{N}_n^{\text{nei}}}(\phi(n, m), K, \text{descending}) \\
\mathcal{N}_n^{\text{spu}} &= \text{SelectTopK}_{m \in \mathcal{N}_n^{\text{nei}}}(\phi(n, m), K, \text{ascending}) \\
\mathcal{N}_n^{\text{ver}} &= \text{SelectTopK}_{m \in \mathcal{N}_n^{\text{non}}}(\phi(n, m), K, \text{descending}) \\
\mathcal{N}_n^{\text{neg}} &= \text{SelectTopK}_{m \in \mathcal{N}_n^{\text{non}}}(\phi(n, m), K, \text{ascending})
\end{aligned} \tag{2}$$

where $\text{SelectTopK}(\cdot)$ returns the top- K nodes under the specified order, and ties in ϕ are broken deterministically (e.g., by node IDs) to ensure a unique partition. This construction yields four candidate sets: $\mathcal{N}_n^{\text{pos}}$ (reliable neighbors), $\mathcal{N}_n^{\text{neg}}$ (easy negatives), $\mathcal{N}_n^{\text{spu}}$ (suspicious edges), and $\mathcal{N}_n^{\text{ver}}$ (potential missing links). Notably, $\mathcal{N}_n^{\text{ver}}$ forms a hard-negative/missing-link candidate pool and is later disambiguated by the LLM.

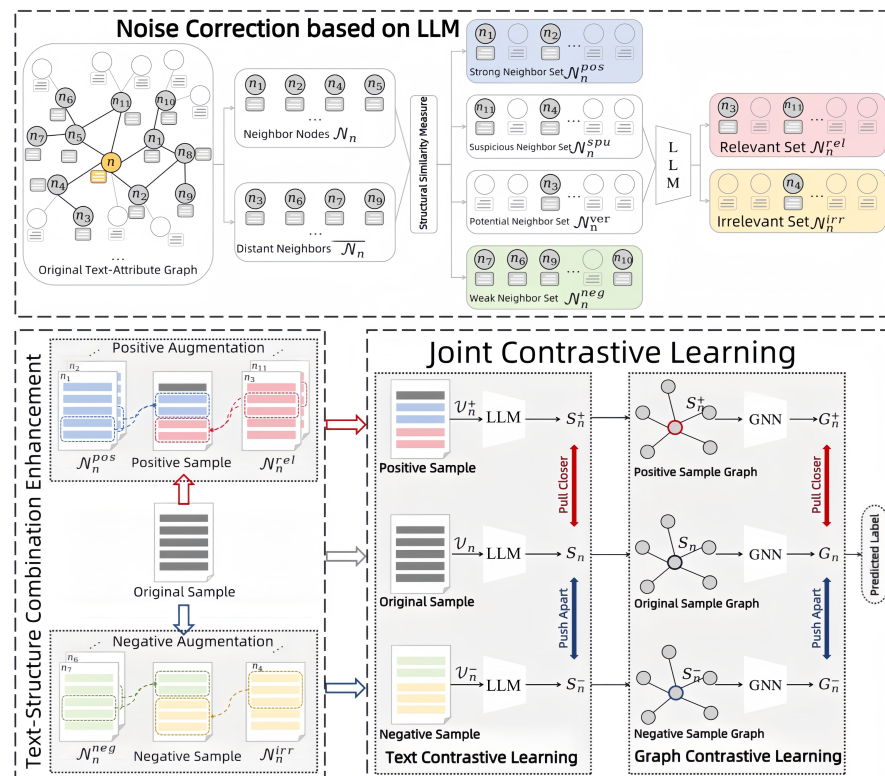


Figure 1. Overall architecture of Coupled Augmentation for Contrastive Learning (CoACL).

We choose the candidate size K subject to the feasibility constraint in Equation (3):

$$K \in \mathbb{Z}^+, \quad 1 \leq K \leq \min\left(\frac{|\mathcal{N}_n^{\text{nei}}|}{2}, \frac{|\mathcal{N}_n^{\text{non}}|}{2}\right) \tag{3}$$

where $|\cdot|$ denotes set cardinality. In practice, we clip K to the maximum allowable value when necessary.

The sets $\mathcal{N}_n^{\text{spu}}$ and $\mathcal{N}_n^{\text{ver}}$ exhibit a conflict between connectivity and structural similarity, so we further judge them using textual semantics. Given a prompt constructor $\mathcal{P}(n, \mathcal{X})$ that packs the text of node n and candidates $\mathcal{X}$ into a fixed template, the LLM returns a binary decision for each candidate. We formalize this semantic partition in Equation (4):

$$\begin{aligned}
\Psi(n, \mathcal{X}) &= \{m \in \mathcal{X} \mid [F_{\text{LLM}}(\mathcal{P}(n, \mathcal{X}))]_m = 1\} \\
\bar{\Psi}(n, \mathcal{X}) &= \{m \in \mathcal{X} \mid [F_{\text{LLM}}(\mathcal{P}(n, \mathcal{X}))]_m = 0\}
\end{aligned} \tag{4}$$

where $F_{\text{LLM}}(\mathcal{P}(n, \mathcal{X})) \in \{0, 1\}^{|\mathcal{X}|}$ returns a binary decision for each candidate in $\mathcal{X}$, and $[\cdot]_m$ denotes the entry corresponding to candidate m .

Using these mapping operators, we refine ambiguous candidates into two interpretable sets in Equation (5):

$$\begin{aligned}\mathcal{N}_n^{\text{rel}} &= \Psi(n, \mathcal{N}_n^{\text{spu}}) \cup \Psi(n, \mathcal{N}_n^{\text{ver}}) \\ \mathcal{N}_n^{\text{irr}} &= \bar{\Psi}(n, \mathcal{N}_n^{\text{spu}}) \cup \bar{\Psi}(n, \mathcal{N}_n^{\text{ver}})\end{aligned}\quad (5)$$

where $\mathcal{N}_n^{\text{rel}}$ denotes semantically relevant candidates (kept/added as positives) and $\mathcal{N}_n^{\text{irr}}$ denotes semantically irrelevant candidates (treated as negatives).

We finally treat $\mathcal{N}_n^{\text{pos}} \cup \mathcal{N}_n^{\text{rel}}$ as positive candidates and $\mathcal{N}_n^{\text{neg}} \cup \mathcal{N}_n^{\text{irr}}$ as negative candidates for subsequent coupled contrastive learning. For directed graphs, semantic evaluation is performed in a node-conditional manner without enforcing symmetry between (n, m) and (m, n) . For undirected graphs, the semantic decision is shared by both directions.

3.2.2. Text–Structure Coupling Enhancements

We next construct coupled text–structure views to support contrastive learning. Specifically, we build text and structure augmentations that share consistent positive/negative semantics, using the refined candidate sets $\mathcal{N}_n^{\text{pos}}$, $\mathcal{N}_n^{\text{neg}}$, $\mathcal{N}_n^{\text{rel}}$, and $\mathcal{N}_n^{\text{irr}}$.

Text Branch

For node n , we augment its textual attributes by sampling keywords from its own text and from semantically consistent nodes. We construct the positive and negative textual views in Equation (6):

$$\begin{aligned}\mathcal{V}_n^+ &= \{t_1, t_2, \dots, t_k\} \\ t_i &\in \bigcup_{m \in \mathcal{N}_n^{(0)} \cup \mathcal{N}_n^{\text{pos}} \cup \mathcal{N}_n^{\text{rel}}} \text{TopD}_{t \in \mathcal{T}_m}(\text{TF-IDF}(t, \mathcal{T}_m)) \\ \mathcal{V}_n^- &= \{t_1, t_2, \dots, t_k\} \\ t_i &\in \bigcup_{m \in \mathcal{N}_n^{\text{neg}} \cup \mathcal{N}_n^{\text{irr}}} \text{TopD}_{t \in \mathcal{T}_m}(\text{TF-IDF}(t, \mathcal{T}_m))\end{aligned}\quad (6)$$

where $\mathcal{T}_m$ denotes the description text of node m , $\text{TopD}(\cdot)$ returns the top- D terms ranked by TF-IDF score, k is the number of sampled keywords, and $\mathcal{N}_n^{(0)}$ denotes the root node itself. Here, $\mathcal{N}_n^{\text{pos}}$, $\mathcal{N}_n^{\text{rel}}$, $\mathcal{N}_n^{\text{neg}}$, and $\mathcal{N}_n^{\text{irr}}$ are node-conditional candidate sets representing semantically equivalent, semantically related, hard negative, and semantically irrelevant nodes with respect to n , respectively, and are constructed in an asymmetric manner. The positive view $\mathcal{V}_n^+$ draws keywords from semantically consistent nodes, while the negative view $\mathcal{V}_n^-$ samples from hard negatives and irrelevant candidates.

Structure Branch

Analogously, we construct coupled structural positives and negatives by reusing the same semantics in Equation (7):

$$\begin{aligned}\mathcal{N}_n^+ &= \mathcal{N}_n^{\text{pos}} \cup \mathcal{N}_n^{\text{rel}} \\ \mathcal{N}_n^- &= \mathcal{N}_n^{\text{neg}} \cup \mathcal{N}_n^{\text{irr}}\end{aligned}\quad (7)$$

This definition treats semantically consistent candidates as structural positives and semantically inconsistent candidates as negatives, thereby aligning textual semantics with graph connectivity and providing consistent positives/negatives for the subsequent contrastive objectives.

3.2.3. Joint Contrastive Learning

Conventional TAG contrastive methods often optimize text and graph encoders independently. CoACL jointly optimizes both branches with coupled objectives, where the text-side and graph-side objectives are defined in Equation (8):

$$\begin{aligned}\mathcal{L}_{\mathcal{T}} &= \mathcal{L}_{\mathcal{T}}^{\text{original}} + \lambda \mathcal{L}_{\mathcal{T}}^{\text{couple}} \\ \mathcal{L}_{\mathcal{G}} &= \mathcal{L}_{\mathcal{G}}^{\text{original}} + \gamma \mathcal{L}_{\mathcal{G}}^{\text{couple}}\end{aligned}\quad (8)$$

where $\mathcal{L}_{\mathcal{T}}^{\text{original}}$ and $\mathcal{L}_{\mathcal{G}}^{\text{original}}$ are supervised classification losses, $\mathcal{L}_{\mathcal{T}}^{\text{couple}}$ and $\mathcal{L}_{\mathcal{G}}^{\text{couple}}$ are coupled contrastive losses for the text and graph branches, and λ and γ are trade-off coefficients. These objectives combine supervised learning and coupled contrastive learning to improve robustness under noise and limited labels.

We decompose the coupled losses into positive and negative terms in Equation (9):

$$\begin{aligned}\mathcal{L}_{\mathcal{T}}^{\text{couple}} &= \mathcal{L}_{\mathcal{T}}^{\text{pos}} + \mathcal{L}_{\mathcal{T}}^{\text{neg}} \\ \mathcal{L}_{\mathcal{G}}^{\text{couple}} &= \mathcal{L}_{\mathcal{G}}^{\text{pos}} + \mathcal{L}_{\mathcal{G}}^{\text{neg}}\end{aligned}\quad (9)$$

where $\mathcal{L}^{\text{pos}}$ is an InfoNCE-style term that aligns the original view with the positive augmented view using in-batch negatives, while $\mathcal{L}^{\text{neg}}$ explicitly penalizes similarity to the negative augmented view. This shared decomposition is used in both branches.

Text Branch

Given the original and keyword-based augmented textual views, we obtain text embeddings $\mathbf{S}_n = f_{\text{enc}}^{(t)}(\mathcal{V}_n)$, $\mathbf{S}_n^+ = f_{\text{enc}}^{(t)}(\mathcal{V}_n^+)$, and $\mathbf{S}_n^- = f_{\text{enc}}^{(t)}(\mathcal{V}_n^-)$. The supervised classification loss is defined in Equation (10):

$$\mathcal{L}_{\mathcal{T}}^{\text{original}} = -\frac{1}{N} \sum_{n=1}^N \sum_{c=1}^C y_{n,c} \log \hat{y}_{n,c}^{(t)} \quad (10)$$

where $\hat{\mathbf{y}}_n^{(t)} = f_{\text{clf}}^{(t)}(\mathbf{S}_n) \in \mathbb{R}^C$ is the class-probability prediction of node n in the text branch, $\hat{y}_{n,c}^{(t)}$ is its c -th entry, $y_{n,c} \in \{0, 1\}$ is the one-hot ground-truth label, N is the number of labeled nodes, and C is the number of classes.

The positive and negative text contrastive terms are defined in Equation (11):

$$\begin{aligned}\mathcal{L}_{\mathcal{T}}^{\text{pos}} &= -\frac{1}{|\mathcal{B}|} \sum_{n \in \mathcal{B}} \log \frac{\exp(\phi(\mathbf{S}_n, \mathbf{S}_n^+)/\tau)}{\sum_{k \in \mathcal{B}} \exp(\phi(\mathbf{S}_n, \mathbf{S}_k^+)/\tau)} \\ \mathcal{L}_{\mathcal{T}}^{\text{neg}} &= \frac{1}{|\mathcal{B}|} \sum_{n \in \mathcal{B}} \phi(\mathbf{S}_n, \mathbf{S}_n^-)\end{aligned}\quad (11)$$

where $\mathcal{B}$ is the mini-batch of nodes, $\phi(\cdot, \cdot)$ is cosine similarity, and τ is the temperature hyperparameter. Note that $\mathcal{L}_{\mathcal{T}}^{\text{pos}}$ is an InfoNCE-style term that uses in-batch samples as negatives via the denominator, while $\mathcal{L}_{\mathcal{T}}^{\text{neg}}$ explicitly pushes the representation away from the negative augmented view. Together, these losses encourage consistent positives and informative negatives in the text branch.

Graph Branch

Similarly, the supervised classification loss in the graph branch is defined in Equation (12):

$$\mathcal{L}_{\mathcal{G}}^{\text{original}} = -\frac{1}{N} \sum_{n=1}^N \sum_{c=1}^C y_{n,c} \log \hat{y}_{n,c}^{(g)} \quad (12)$$

The prediction vector and the positive/negative graph contrastive terms are defined in Equation (13):

$$\begin{aligned}\hat{\mathbf{y}}_n^{(g)} &= f_{cls}^{(g)}(\mathbf{G}_n^{(L)}) \in \mathbb{R}^C \\ \mathcal{L}_G^{\text{pos}} &= -\frac{1}{|\mathcal{B}|} \sum_{n \in \mathcal{B}} \log \frac{\exp(\phi(\mathbf{G}_n^{(L)}, (\mathbf{G}_n^+)^{(L)})/\tau)}{\sum_{k \in \mathcal{B}} \exp(\phi(\mathbf{G}_n^{(L)}, (\mathbf{G}_k^+)^{(L)})/\tau)} \\ \mathcal{L}_G^{\text{neg}} &= \frac{1}{|\mathcal{B}|} \sum_{n \in \mathcal{B}} \phi(\mathbf{G}_n^{(L)}, (\mathbf{G}_n^-)^{(L)})\end{aligned}\quad (13)$$

Our negative semantic term is designed as a lightweight cosine-similarity penalty that operates in the same normalized embedding space as InfoNCE, thus remaining fully consistent with the similarity measure used for contrastive alignment. Unlike margin-based contrastive or triplet losses, which typically introduce an additional margin hyperparameter and rely on explicit triplet construction/mining (often sensitive to sampling and batch composition), this formulation directly discourages undesired semantic alignment by suppressing high cosine similarity for negative semantic pairs. In practice, it complements InfoNCE by providing an explicit “push-away” signal for semantically irrelevant pairs without complicating optimization or requiring careful margin tuning.

In the above equation, $\mathbf{G}_n^{(L)}$ is the final-layer GNN representation of node n , $\hat{\mathbf{y}}_n^{(g)}$ is the graph-branch prediction vector, and $\hat{y}_{n,c}^{(g)}$ is its c -th entry. The other symbols ($\mathcal{B}$, $\phi(\cdot, \cdot)$, τ , N , C , and $y_{n,c}$) follow the same definitions as in the text branch.

Jointly minimizing the text and graph objectives encourages consistent positives and informative negatives across modalities, thereby improving robustness under noisy structures and limited supervision.

3.3. Description of the CoACL Algorithm Process

The workflow of the proposed CoACL is presented in Algorithm 1.

Algorithm 1 TAG contrastive learning algorithm based on coupling enhancement

Input:

- $\mathcal{G} = (\mathcal{N}, \mathcal{E}, A, \mathcal{T})$: Text-attributed graphs
- n : Any node
- K : Candidate set size
- $\{y_i\}_{i=1}^N$: Label set
- T : Training rounds
- $\mathcal{B}$: The batch size of training data
- τ : Temperature coefficient
- γ, λ : Coupling loss weights

Output: Predicted node labels and test accuracy (ACC)

- 1: Initialize text model and graph model parameters
 - 2: **Precompute and cache semantic partitions using the LLM**
 - 3: **for** each node $n \in \mathcal{V}$ **do**
 - 4: Divide adjacency set $\mathcal{N}_n^{\text{nei}}$ and non-adjacent set $\mathcal{N}_n^{\text{non}}$
 - 5: Calculate node embedding $\mathbf{h}_n$ by Equation (1)
 - 6: Compute cosine similarity $\phi(n, m)$ by Equation (1)
 - 7: Construct four candidate sets:
 - 8: Strong association sets $\mathcal{N}_n^{\text{pos}}$
 - 9: Suspicious association set $\mathcal{N}_n^{\text{spu}}$
-

Algorithm 1 *Cont.*

```

10:   Potential association sets  $\mathcal{N}_n^{ver}$ 
11:   Weak association sets  $\mathcal{N}_n^{neg}$ 
12:   Create prompt constructor  $\mathcal{P}(n, \mathcal{N}_n^{spu})$  and  $\mathcal{P}(n, \mathcal{N}_n^{ver})$  using LLM
13:   Distinguish  $\mathcal{N}_n^{rel}$  and  $\mathcal{N}_n^{irr}$  by Equations (4) and (5)
14: end for
15: while early stopping condition is not met and epoch  $\leq T$  do
16:   for each mini-batch  $\mathcal{B} \subseteq \mathcal{V}_{train}$  do
17:     for each node  $n \in \mathcal{B}$  do
18:       for each node  $n \in \mathcal{V}$  do
19:         Generate positive/negative samples  $\mathcal{V}_n^+$  and  $\mathcal{V}_n^-$  by Equation (6)
20:         Calculate text embeddings:
21:            $\mathbf{S}_n = f_{cls}(\mathcal{V}_n)$ ,  $\mathbf{S}_n^+ = f_{cls}(\mathcal{V}_n^+)$ ,  $\mathbf{S}_n^- = f_{cls}(\mathcal{V}_n^-)$ 
22:         Calculate graph embeddings:  $\mathbf{G}_n, \mathbf{G}_n^+, \mathbf{G}_n^-$ 
23:       end for
24:       Compute  $\mathcal{L}_T$  and  $\mathcal{L}_G$  by Equation (8)
25:       Update model parameters based on  $\mathcal{L}_T$  and  $\mathcal{L}_G$ 
26:       Perform early stop judgment
27:     end for
28:   end for
29:   Evaluate on validation set
30: end while
31: Evaluate the trained model on  $\mathcal{V}_{test}$  and report ACC by Equation (14)

```

4. Experiments

4.1. Experimental Setup

All methods are implemented in Python3.10 using PyTorch, along with standard NLP and GNN toolchains. We run all experiments on a server equipped with NVIDIA A100 GPUs (80 GB each) $\times 2$, an AMD EPYC 7642 48-Core CPU, and 512 GB RAM, running Ubuntu 20.04 LTS, as summarized in Table 2. Unless otherwise specified, we use the default hyperparameter settings of each baseline and report the average performance over multiple runs with different random seeds.

Table 2. Hardware and software environment (CPU: central processing unit; GPU: graphics processing unit; RAM: random-access memory; OS: operating system; LTS: long-term support).

Item	Specification
CPU	AMD EPYC 7642 48-Core Processor
GPU	NVIDIA A100 (80 GB each) $\times 2$
RAM	512 GB
OS	Ubuntu 20.04 LTS (Linux)
Disk	20 TB
Python	3.10
PyTorch	2.12.0

4.1.1. Implementation Details

Unless otherwise specified, we repeat each experiment three times with different random seeds and report the mean $\pm$ standard deviation over the three runs. The maximum number of training epochs is set to 100 with early stopping based on validation accuracy. We optimize all models using Adam with a learning rate of 1×10^{-3} and a mini-batch size of 32. For hyperparameter selection, we search the candidate set size K in $\{2, 6, 8, 10, 20, 40, 60, 80, 100\}$, and the coupling loss weights λ and γ in $\{0, 0.5, 1, 1.5, 2\}$. For the LLM-based semantic evaluation in the noise-correction module, we use ChatGLM-130B. For LLM-based baselines, we initialize the encoder with bert-base-uncased.

To ensure fair comparison, all baseline methods and CoACL are trained and evaluated using exactly the same preprocessed node texts. No baseline is provided with additional textual information or alternative preprocessing beyond the standardized pipeline described above.

4.1.2. Datasets

To comprehensively evaluate the effectiveness of CoACL, we conduct node classification on three real-world text-attributed graph datasets: Cora [31], OGBN-Arxiv [12], and PubMed [31]. For all datasets, we apply a unified text preprocessing pipeline to ensure consistency and fairness across methods. The pipeline includes lowercasing, removal of non-informative symbols, normalization of whitespace, and truncation to a fixed maximum length. To ensure fair comparison, all baseline methods and CoACL are trained and evaluated using exactly the same preprocessed node texts. No baseline is provided with additional textual information or alternative preprocessing beyond the standardized pipeline described above. The main statistics of the processed datasets are summarized in Table 3.

- **Cora Dataset:** This dataset is a widely used citation network in the field of machine learning, originally provided by Repositories Group. It contains 2708 scientific papers, each represented by a 1433-dimensional binary word vector indicating the presence or absence of specific terms in a vocabulary. Citation relationships between papers form 5429 edges, creating a citation graph where papers are classified into seven categories. For the Cora dataset, additional cleaning is required because the original node text contains incomplete keyword lists and occasional formatting artifacts. Specifically, we remove duplicated keywords, filter malformed tokens, and complete missing textual fields when possible using the dataset-provided metadata. After preprocessing, all node texts are standardized into the same text-attributed graph format.
- **OGBN-Arxiv Dataset:** This dataset constructs a directed graph representing citation relationships among arXiv papers. Each paper is manually labeled by arXiv authors and moderators with one of 40 categories selected from diverse academic fields such as computer science, mathematics, and physics, reflecting significant interdisciplinary characteristics. To ensure the scientific rigor and reproducibility of the experiments, we strictly followed the official dataset split rules to maintain standardization and consistency during data processing.
- **PubMed Dataset:** This dataset is constructed from the PubMed database and consists of 19,717 nodes and 44,338 edges, forming a citation network that reflects relationships among biomedical papers. Each node represents a scientific paper related to diabetes and is characterized by a 500-dimensional TF-IDF weighted word frequency vector. The dataset is divided into three categories, corresponding to different research directions on diabetes. To ensure data quality and experimental reproducibility, Our framework adhered to the dataset split rules established in prior works.

Table 3. Dataset Details.

Dataset	# Nodes	# Edges	# Classes	Train/Val/Test (%)
Cora	2708	5429	7	60/20/20
OGBN-Arxiv	169,343	1,166,243	40	54/18/28
PubMed	19,717	44,338	3	80/10/10

4.1.3. Baselines

We evaluate combinations of text encoders and GNN backbones for node classification on TAGs. The text encoder is chosen from shallow embeddings (SE; e.g., Bag-of-Words (BoW)) [14], a frozen language model (LLM-plain), a fine-tuned language model

(LLM-tune), and GIANT, a recently proposed TAG learning baseline that enhances node representations with external textual signals. The GNN backbone is selected from Graph Convolutional Network (GCN) [17], Graph Attention Network (GAT) [18], and GraphSAGE (SAGE) [19]. For CoACL, we use ChatGLM-130B to provide LLM-based semantic supervision in the augmentation modules. For baselines that require dense text embeddings (e.g., LLM-plain and LLM-tune), we use BERT-base-uncased [32,33] to obtain initial text representations.

- **BoW**: Bag-of-Words features as sparse shallow text embeddings (SE).
- **GCN**: A standard message-passing backbone with neighborhood aggregation.
- **GAT**: A message-passing backbone with attention-based neighbor weighting.
- **SAGE**: An inductive backbone with sampling-based neighborhood aggregation for scalability.

4.1.4. Evaluation Method

We evaluate node classification on TAGs using accuracy (ACC). Given a standard split of nodes into $\mathcal{V}_{\text{train}}$, $\mathcal{V}_{\text{val}}$, and $\mathcal{V}_{\text{test}}$, we report ACC on $\mathcal{V}_{\text{test}}$, defined in Equation (14):

$$\text{ACC} = \frac{1}{|\mathcal{V}_{\text{test}}|} \sum_{i \in \mathcal{V}_{\text{test}}} \mathbb{I}(y_i = \hat{y}_i) \quad (14)$$

where $|\mathcal{V}_{\text{test}}|$ is the number of test nodes, y_i is the ground-truth label of node i , $\hat{y}_i$ is the predicted label, and $\mathbb{I}(\cdot)$ is an indicator function that equals 1 if the statement is true and 0 otherwise. ACC directly measures the fraction of correctly predicted labels and is used as the primary metric in our experiments.

4.2. Performance Evaluation

To validate the effectiveness of CoACL, we conducted extensive comparative experiments against a variety of baselines using node-classification ACC. Tables 4–6 report the results on three datasets under labeling ratios of 1%, 5%, and 10%. In these tables, *Gain* denotes the absolute ACC improvement of CoACL over the strongest baseline under each labeling ratio.

Table 4. Comparison of node-classification ACC on the OGBN-Arxiv dataset under different labeling ratios.

Embedding	Methods	OGBN-Arxiv		
		1%	5%	10%
SE	GCN	41.4 ± 0.03	44.7 ± 0.13	56.3 ± 0.05
	GAT	40.4 ± 0.09	43.5 ± 0.07	55.7 ± 0.03
	SAGE	34.6 ± 0.08	41.7 ± 0.08	53.8 ± 0.09
LLM-plain	GCN	47.4 ± 0.07	49.3 ± 0.10	60.3 ± 0.09
	GAT	46.9 ± 0.15	52.3 ± 0.06	65.5 ± 0.10
	SAGE	46.2 ± 0.11	47.8 ± 0.08	63.4 ± 0.06
LLM-tune	GCN	53.2 ± 0.14	54.6 ± 0.19	65.2 ± 0.14
	GAT	49.6 ± 0.12	53.5 ± 0.18	66.3 ± 0.12
	SAGE	51.4 ± 0.15	56.6 ± 0.16	68.9 ± 0.10
GIANT	GCN	45.9 ± 0.07	53.5 ± 0.08	65.9 ± 0.07
	GAT	41.8 ± 0.08	51.1 ± 0.02	67.7 ± 0.07
	SAGE	48.4 ± 0.03	54.8 ± 0.04	67.2 ± 0.09
CoACL (Ours)		57.8 ± 0.16	63.7 ± 0.08	72.7 ± 0.05
Gain		4.6	7.1	3.8

Table 5. Comparison of node-classification ACC on the PubMed dataset under different labeling ratios.

Embedding	Methods	PubMed		
		1%	5%	10%
SE	GCN	36.8 ± 0.08	47.1 ± 0.16	59.6 ± 0.10
	GAT	44.1 ± 0.04	47.2 ± 0.15	59.9 ± 0.12
	SAGE	44.1 ± 0.04	51.6 ± 0.07	54.4 ± 0.06
LLM-plain	GCN	43.3 ± 0.05	55.4 ± 0.09	60.7 ± 0.13
	GAT	45.0 ± 0.07	61.8 ± 0.11	65.7 ± 0.15
	SAGE	43.9 ± 0.13	59.8 ± 0.11	62.6 ± 0.13
LLM-tune	GCN	48.3 ± 0.10	62.4 ± 0.11	72.4 ± 0.13
	GAT	46.2 ± 0.08	62.8 ± 0.05	69.1 ± 0.11
	SAGE	45.9 ± 0.10	63.1 ± 0.09	73.4 ± 0.14
GIANT	GCN	41.9 ± 0.5	59.2 ± 1.0	72.5 ± 1.0
	GAT	44.8 ± 0.2	61.0 ± 0.9	72.7 ± 1.1
	SAGE	45.0 ± 0.1	61.5 ± 0.9	72.8 ± 1.2
CoACL (Ours)		51.4 ± 0.2	69.8 ± 0.9	77.6 ± 0.3
Gain		3.1	6.7	4.2

Table 6. Comparison of node-classification accuracy ACC on the Cora dataset under different labeling ratios.

Embedding	Methods	Cora		
		1%	5%	10%
SE	GCN	64.5 ± 0.05	71.0 ± 0.10	73.3 ± 0.06
	GAT	59.0 ± 0.05	63.5 ± 0.08	67.0 ± 0.04
	SAGE	57.0 ± 0.09	65.0 ± 0.06	70.7 ± 0.06
LLM-plain	GCN	69.9 ± 0.03	75.2 ± 0.10	76.3 ± 0.12
	GAT	69.0 ± 0.08	75.7 ± 0.08	73.5 ± 0.06
	SAGE	63.6 ± 0.09	70.7 ± 0.12	73.3 ± 0.15
LLM-tune	GCN	74.4 ± 0.08	78.4 ± 0.06	83.8 ± 0.08
	GAT	71.0 ± 0.24	76.9 ± 0.16	79.8 ± 0.18
	SAGE	73.1 ± 0.19	77.6 ± 0.11	80.5 ± 0.15
GIANT	GCN	67.2 ± 0.03	77.2 ± 0.03	79.4 ± 0.07
	GAT	66.4 ± 0.04	76.4 ± 0.09	79.3 ± 0.08
	SAGE	66.1 ± 0.07	76.2 ± 0.09	78.8 ± 0.07
CoACL (Ours)		78.2 ± 0.17	84.7 ± 0.12	86.2 ± 0.08
Gain		3.8	6.3	2.4

Taken together, these results show that CoACL consistently outperforms the baselines across datasets and labeling ratios. Compared with the strongest baseline in each setting, CoACL improves node classification accuracy by roughly 3–8%, with the largest gains appearing under noisier or lower-label regimes. These results suggest that jointly contrasting text and structure, together with semantics-aware augmentation, better captures complementary information in TAGs and improves robustness to noise.

CoACL is also robust in low-label regimes. On Cora, when the labeling ratio decreases from 10% to 1%, CoACL drops from 86.2 to 78.2, while the strongest baseline (LLM-tune with GCN) decreases from 83.8 to 74.4; CoACL therefore preserves a 3.8-point advantage under 1% labels. Similar robustness is observed on PubMed and OGBN-Arxiv, where CoACL improves the best baseline by 3.1 and 4.6 points under 1% labeling, respectively. This robustness stems

from the coupled augmentation strategy, which strengthens view consistency and reduces the impact of noisy neighborhoods during contrastive optimization.

4.3. Ablation Study

The CoACL model consists of three core components: noise correction, node attribute augmentation and graph structure augmentation based on the LLM. In order to deeply analyze the impact of each component on the model performance and comprehensively reveal the working principle of the model, a series of meticulous ablation experiments are carried out in our method. All ablation experiments are conducted using the same preprocessed text as in the main experiments, ensuring that observed performance differences are caused by architectural or objective-function changes rather than text quality variations.

All ablation experiments are conducted using the same preprocessed text as in the main experiments, ensuring that observed performance differences are caused by architectural or objective-function changes rather than text quality variations.

4.3.1. Effect of LLM-Based Noise Correction

The results of the ablation experiment to assess the effectiveness of the noise correction mechanism based on the LLM for TAGL are shown in Figure 2.

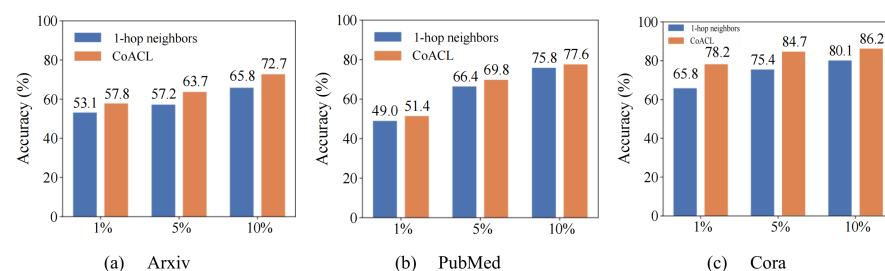


Figure 2. Effectiveness of CoACL noise correction mechanism based on large language model on three datasets.

In all three datasets, with the increase in the proportion of labeled data, the classification accuracies of both methods show a rising trend, but the CoACL method always outperforms the method that only uses the first-order neighbors (1-hop neighbors), which fully verifies the effectiveness of the noise correction mechanism based on the LLM. Specifically, in the Arxiv dataset, the classification accuracy of the CoACL method is improved by about 4 to 7 percentage points compared with that of the 1-hop neighbors method, while in the PubMed dataset, although the overall improvement is relatively small, it still reaches 2 to 4 percentage points, which indicates that the noise-correction mechanism has a positive impact on the performance of the model. Notably, in the Cora dataset, the CoACL method demonstrates the most significant improvement in accuracy, with an overall improvement of 6–12 percentage points, which further highlights its advantage in the high-noise environment. This phenomenon also reflects the differences in noise levels among the three datasets, and it is speculated that the Cora dataset may have unintentionally introduced more noise during preprocessing due to information supplementation, while the PubMed dataset is the dataset with the least noise impact.

In summary, the LLM-based noise correction module improves robustness, especially under higher noise levels, by filtering unreliable connections and enriching semantically consistent associations. This strengthens the quality of augmented views and supports effective representation learning for TAGs.

4.3.2. Effect of Coupled Augmentation on Textual Attributes

The results of the ablation experiments evaluating the effectiveness of the coupled-augmentation based contrastive learning mechanism in improving the quality of text representation are shown in Figure 3. CoACL consistently outperforms other text embedding approaches in representing node attributes. This gain is mainly attributed to the coupled augmentation module, which aligns textual semantics with graph structural signals.

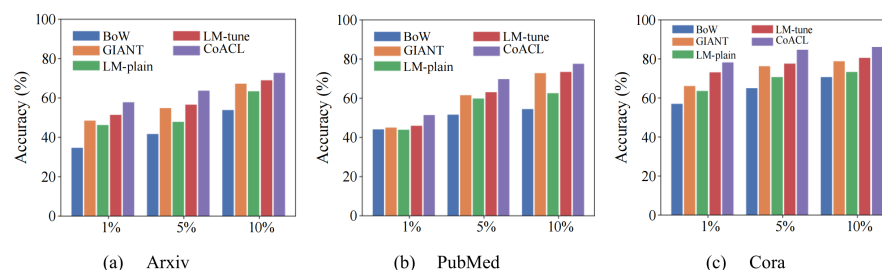


Figure 3. CoACL text learning effectiveness on three datasets.

From the results, it can be seen that CoACL's text learning ability is stronger than other methods in all datasets and labeling ratios, and this performance lead is not purely dependent on more complex model stacking, but rather stems from the optimization of the text representation learning process by the proposed text–structure coupled augmentation module. In addition, a cross-sectional observation of the experimental results also reveals that: the conventional BoW representation performs the worst in all datasets and annotation rates, indicating that the bag-of-words model, which lacks the ability of contextual modeling, can hardly meet the demand of semantic expression of graph structures; LLM-plain and LLM-tune, although they introduce a LLM, still suffer from insufficient generalization without considering the semantic consistency of graph structures; GIAN fuses graph and text, but its bias towards local structures affects the modeling of global semantics.

Collectively, CoACL's coupled augmentation mechanism better bridges semantics and structure, yielding higher-quality representations in complex and noisy graph settings.

4.3.3. Effect of Coupled Augmentation on Graph Structure

The ablation results are reported in Table 7. They quantify how the coupled augmentation objective contributes to learning on text-attributed graph structures. We compare CoACL with Graph Contrastive Learning (GraphCL) and Bootstrapped Graph Representation Learning (BGRL).

Table 7. CoACL structure learning effectiveness on three datasets

Datasets		GraphCL	BGRL	CoACL	Gain
OGBN-Arxiv	1%	47.5 ± 0.05	53.9 ± 0.07	57.8 ± 0.06	3.9
	5%	51.0 ± 0.09	54.3 ± 0.08	63.7 ± 0.08	9.4
	10%	63.1 ± 0.08	65.0 ± 0.08	72.7 ± 0.05	7.7
PubMed	1%	44.8 ± 0.03	46.9 ± 0.16	51.4 ± 0.12	4.5
	5%	62.5 ± 0.05	63.6 ± 0.04	69.8 ± 0.09	6.2
	10%	71.4 ± 0.04	73.7 ± 0.07	77.6 ± 0.03	3.9
Cora	1%	68.5 ± 0.02	70.3 ± 0.03	78.2 ± 0.07	7.9
	5%	75.4 ± 0.05	76.1 ± 0.03	84.7 ± 0.12	8.6
	10%	80.4 ± 0.04	81.0 ± 0.03	86.2 ± 0.08	5.2

The results show that CoACL achieves the best performance, highlighting the benefit of semantics-aware, coupled augmentation for representation learning on TAGs. CoACL's

advantage is particularly pronounced at low labeling rates, suggesting that the coupling mechanism makes better use of limited supervision to extract more discriminative structural signals. Conventional graph contrastive learning methods typically rely on generic augmentations (e.g., node dropping, edge perturbation, and subgraph sampling), which can distort graph semantics and introduce representation bias. By contrast, CoACL uses LLM-guided semantic evidence to refine candidate relations, leading to more stable and informative message propagation.

4.3.4. Effect of Negative Semantic Penalty

To further examine the contribution of the proposed negative semantic penalty, we conduct an ablation study at the objective level by comparing two training variants: (1) CoACL trained with the standard InfoNCE loss only, and (2) CoACL trained with InfoNCE augmented by the proposed cosine-similarity-based penalty for negative semantic pairs. All other components, including LLM-based noise correction and coupled augmentations, are kept identical.

This comparison aims to isolate the effect of explicitly penalizing semantically negative pairs beyond the implicit negative competition in InfoNCE. The results on three datasets under different labeling ratios are summarized in Table 8.

Table 8. Effect of negative semantic penalty on three datasets.

Dataset	Label Ratio	InfoNCE Only	InfoNCE + NegPenalty	Gain
OGBN-Arxiv	1%	54.6 ± 0.08	57.8 ± 0.06	3.2
	5%	60.8 ± 0.10	63.7 ± 0.08	2.9
	10%	70.4 ± 0.07	72.7 ± 0.05	2.3
PubMed	1%	47.8 ± 0.15	51.4 ± 0.12	3.6
	5%	66.1 ± 0.11	69.8 ± 0.09	3.7
	10%	74.5 ± 0.06	77.6 ± 0.03	3.1
Cora	1%	73.9 ± 0.09	78.2 ± 0.07	4.3
	5%	80.9 ± 0.14	84.7 ± 0.12	3.8
	10%	82.7 ± 0.10	86.2 ± 0.08	3.5

4.4. Parameter Sensitivity

Three key parameters in the CoACL model: the length of the augmented candidate set K for the noise correction module based on the LLM, and the coupled loss weights λ and γ for the text and graph models in joint contrastive learning. In order to analyze in depth the extent to which each of the parameters affects the performance of the model, further experimental validation of its important parameters is carried out in our method.

In Figure 4, we report how the candidate-set size K affects training efficiency and CoACL performance. As K increases, accuracy improves rapidly at first and then saturates. For example, accuracy is 49.2% at $K = 2$ and peaks at 72.7% around $K = 10$ (a + 23.5% gain). Small K provides insufficient structural context, limiting representation learning and reducing the effectiveness of LLM-based correction. Increasing K supplies richer candidates and helps the LLM identify semantically consistent relations, but overly large K yields diminishing returns because the added candidates become less informative.

Training time also increases with K . As the candidate set grows, the noise-correction module must score many more node pairs with the LLM, substantially increasing computational overhead. In practice, $K \approx 10$ offers a good trade-off: it achieves near-optimal accuracy while keeping training cost within a reasonable range. We therefore set $K = 10$ in the LLM-based noise-correction module.

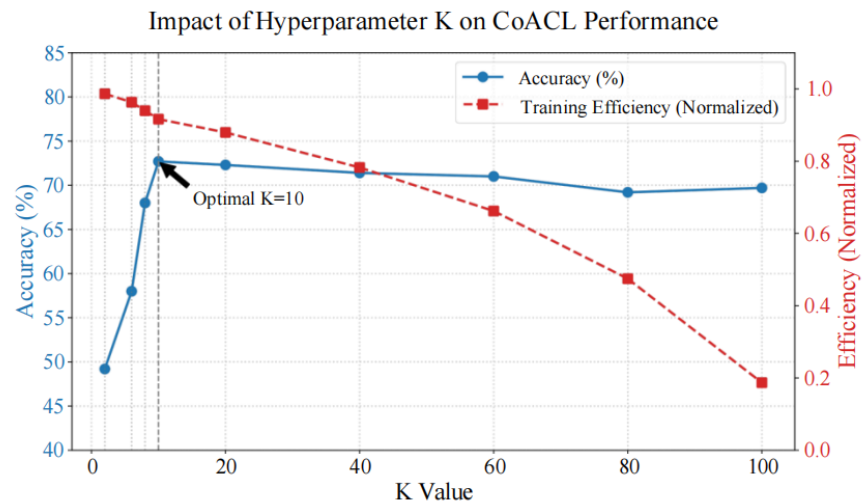


Figure 4. Effect of parameter K on CoACL performance and training efficiency on Arxiv 10%.

For the coupling-loss weights λ and γ , we evaluate different hyperparameter combinations on Cora with a 10% labeling rate. Figure 5 shows that setting $\lambda = \gamma = 0$ yields the lowest accuracy, confirming the importance of both coupling components. The best performance is achieved around $\lambda = 1.5$ (text weight) and $\gamma = 1$ (graph weight). Moreover, the presence of a broad high-performing region indicates that CoACL is not overly sensitive to moderate changes in these hyperparameters.

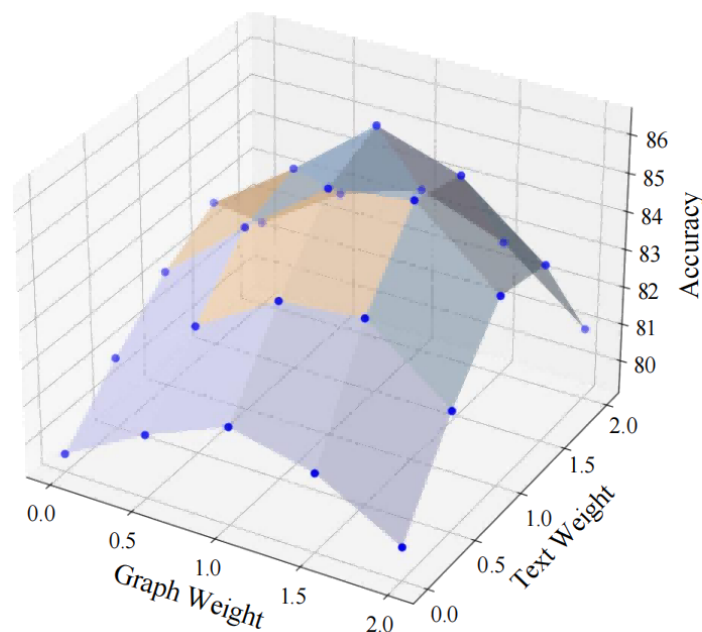


Figure 5. Effect of coupling loss weights on Cora 10 % γ and λ on CoACL performance and training efficiency.

4.5. Limitations and Practical Guidance

LLM-based semantic supervision is most beneficial when the graph structure is noisy (e.g., spurious edges/missing links), textual attributes are ambiguous, or labeled data are scarce, where semantic reasoning can help correct false relations and reduce false negatives.

In contrast, for relatively clean graphs with strong homophily and sufficient supervision, simpler structural or contrastive approaches may already yield competitive performance, and LLM supervision can be optional.

We also acknowledge several practical limitations: (i) potential privacy concerns when querying external LLM services with sensitive node texts, (ii) performance degradation under domain shifts (when text/style differs from the LLM's training distribution), and (iii) occasional semantic errors introduced by imperfect LLM judgments.

In practice, these issues can be mitigated by caching LLM outputs (offline preprocessing), restricting semantic evaluation to candidate subsets, and adopting smaller or local language models for privacy-sensitive or large-scale deployments; for high-stakes settings, lightweight human auditing can further improve reliability.

5. Conclusions

We propose CoACL, an LLM-supervised coupled augmentation framework for contrastive learning on text-attributed graphs. CoACL (i) narrows candidate relations via structural similarity and uses an LLM to validate suspicious edges and potential links, (ii) constructs keyword-focused text augmentations and aligned structural views with consistent semantics, and (iii) jointly optimizes text and graph encoders with coupled contrastive objectives. Experiments on Cora, PubMed, and OGBN-Arxiv show consistent improvements over strong baselines, achieving up to a 7.1% absolute gain in node-classification accuracy, especially in low-label settings. Future work will focus on further reducing the cost of semantic supervision and extending the coupled denoising and augmentation strategy to larger and more complex graphs.

Supplementary Materials: The following supporting information can be downloaded at: <https://www.mdpi.com/article/10.3390/electronics15040844/s1>.

Author Contributions: Conceptualization, H.K. and K.Z.; Methodology, H.K. and S.D.; Software, H.K., Z.Z. (Zhong Zhang), and X.W.; Validation, J.P. and J.G.; Formal analysis, H.K.; Investigation, H.K. and S.D.; Data curation, H.K.; Writing—original draft preparation, H.K.; Writing—review and editing, K.Z.; Supervision, K.Z. and Z.Z. (Zhongping Zhang); Project administration, K.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Central Government Guidance Local Science and Technology Development Fund Project (Grant No. 246Z0306G), the Innovation Capability Improvement Plan Project of Hebei Province (Grant No. 22567626H), and the Key Laboratory for Integrated Publishing Knowledge Mining and Services in Education under Project (Grant No. KT20250802). The authors acknowledge the support from Beijing Knowledge Atlas Technology Joint Stock Company Limited and the Big Data Foundation Model Intelligent Joint Lab of Yanshan University–Zhipu AI.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Public benchmark datasets (Cora, PubMed, and OGBN-Arxiv) are used in this study. The cleaned Cora textual attributes are provided as Supplementary Material (and will be released publicly upon publication) to ensure experimental comparability.

Conflicts of Interest: The authors declare no conflicts of interest.

Appendix A. Prompt Construction and Semantic Decision Process with LLMs

This appendix provides additional details on the prompt construction and large language model (LLM) decision process used in this work. A representative example based on the Cora dataset is presented to improve interpretability and reproducibility, together with a discussion of inference settings and output consistency.

Appendix A.1. Prompt Construction Example

For a target node n , its textual attributes are constructed by concatenating the paper title and abstract. Let

$$\mathcal{X} \in \{\mathcal{N}_n^{sup}, \mathcal{N}_n^{ver}\}$$

denote a candidate node set, where each candidate node is represented by its corresponding textual attributes. A fixed prompt template is adopted to define the prompt construction function $\mathcal{P}(n, \mathcal{X})$.

A representative example of the prompt used in our experiments is shown below.

Example Prompt:

Given a target paper node with the following textual description:

Target node: "Probabilistic methods for information retrieval based on Bayesian inference."

Candidate paper nodes:

1. "A Bayesian framework for text classification and retrieval."
2. "Neural network approaches to handwritten digit recognition."
3. "Graph-based semi-supervised learning methods."

Please identify which candidate papers are most semantically related to the target paper in terms of research topic and content, and return the indices of the relevant nodes.

Based on the constructed prompt, the LLM performs holistic semantic analysis over the candidate set. The output of the LLM is then used to distinguish semantically relevant neighbors from noisy connections.

Appendix A.2. Deterministic Inference Settings

To ensure stable and reproducible outputs, all experiments are conducted using a fixed prompt template and deterministic decoding settings. Specifically, the temperature is set to 0 and top-k to 1, thereby enforcing deterministic inference. Under this configuration, repeated runs with identical inputs produce consistent outputs.

The same LLM checkpoint and inference configuration are used across all experiments to minimize additional sources of variability.

Appendix A.3. Handling Noisy and Inconsistent Outputs

LLMs may occasionally produce ambiguous or uncertain judgments in complex semantic reasoning tasks. To mitigate the impact of such behavior, a conservative filtering strategy is adopted. When the LLM output fails to clearly distinguish semantic relevance or exhibits insufficient confidence, the corresponding candidate edge is treated as a low-confidence connection and excluded from the refined graph structure.

Furthermore, multiple independent inference runs are conducted on a subset of nodes to assess output consistency. The results demonstrate a high level of agreement across runs under fixed prompts and deterministic decoding settings, supporting the robustness of the proposed semantic supervision mechanism.

It is important to note that the proposed method does not rely on fine-grained ranking scores or probability estimates produced by the LLM. Instead, it only leverages the LLM's coarse-grained semantic discrimination capability over candidate connections, which further mitigates the impact of potential output variability on overall performance.

Appendix A.4. Runtime Overhead and LLM Invocation Analysis

In this subsection, we provide a brief analysis of the runtime overhead introduced by the LLM-based semantic supervision module, focusing on the number of LLM invocations and system-level time consumption. Table A1 summarizes the key runtime-related statistics of this module across different datasets, including the dataset name, number of processed nodes, neighborhood size K , average input and output token lengths, and the per-call LLM latency under deterministic decoding (temperature set to 0).

For each target node n , the LLM is invoked once for each candidate set $\mathcal{X} \in \{\mathcal{N}_n^{spu}, \mathcal{N}_n^{ver}\}$. As shown in Table A1, this design leads to a total number of LLM calls that grows linearly with the number of target nodes and remains independent of the neighborhood size K . Increasing K only affects the length of the input prompt, which is reflected in the average input token statistics reported in the table, rather than the frequency of LLM invocations.

Table A1. LLM preprocessing cost on different datasets (ChatGLM-130B).

Dataset	$ \mathcal{V} $	Calls ($=2 \mathcal{V} $)	K	Avg. Input Tokens/Query	Avg. Output Tokens/Query	Avg. Wall-Clock per Call
Cora	2708	5416	10	963	10.4	6.2 s
PubMed	19,717	39,434	10	1068	12.1	8.5 s
OGBN-Arxiv	169,343	338,686	10	1167	14.5	9.3 s

Note: Calls = $2|\mathcal{V}|$. Tokens and time are measured in the offline LLM preprocessing stage and averaged over all queries.

The runtime overhead is measured as the end-to-end wall-clock time per LLM query, including prompt construction and model inference. Under deterministic decoding with a fixed prompt template and temperature set to 0, the per-query runtime is stable across runs, which is consistent with the relatively low variance observed in the per-call latency statistics in Table A1. While larger values of K lead to longer input prompts and a moderate increase in average input token length, the resulting inference time increases smoothly and remains well controlled.

Empirically, Table A1 summarizes the measured per-call latency together with the total invocation counts, indicating that the semantic supervision stage constitutes only a limited portion of the overall computational pipeline.

Across common citation network datasets such as Cora, the introduced overhead does not dominate either training or inference time, indicating that the proposed approach achieves a practical and manageable runtime cost.

References

1. He, X.; Bresson, X.; Laurent, T.; Perold, A.; LeCun, Y.; Hooi, B. Harnessing Explanations: LLM-to-LM Interpreter for Enhanced Text-Attributed Graph Representation Learning. *arXiv* **2024**, arXiv:2305.19523. <https://doi.org/10.48550/arXiv.2305.19523>.
2. Li, X.; Chen, W.; Chu, Q.; Li, H.; Sun, Z.; Li, R.; Qian, C.; Wei, Y.; Liu, Z.; Shi, C.; et al. Can Large Language Models Analyze Graphs like Professionals? A Benchmark, Datasets and Models. *Adv. Neural Inf. Process. Syst.* **2024**, *37*, 141045–141070. <https://doi.org/10.5555/3737916.3742394>.
3. Zeng, X.; Peng, H.; Li, A. Proactive Bot Detection Based on Structural Information Principles. *IEEE Trans. Pattern Anal. Mach. Intell.* **2025**, *1*–18. <https://doi.org/10.1109/TPAMI.2025.3646644>.
4. Geng, S.; Liu, S.; Fu, Z.; Ge, Y.; Zhang, Y. Recommendation as Language Processing (RLP): A Unified Pretrain, Personalized Prompt & Predict Paradigm (P5). In *Proceedings of the RecSys '22: Sixteenth ACM Conference on Recommender Systems, Seattle, WA, USA, 18–23 September 2022*; Golbeck, J., Harper, F.M., Murdock, V., Ekstrand, M.D., Shapira, B., Basilico, J., Lundgaard, K.T., Oldridge, E., Eds.; Association for Computing Machinery: New York, NY, USA, 2022; pp. 299–315. <https://doi.org/10.1145/3523227.3546767>.
5. Zhang, J.; Peng, H.; Yang, M.; Yu, P.S. Enhanced Pre-training for Recommendation via Hypergraph Structural Entropy. *ACM Trans. Inf. Syst.* **2025**, *44*, 1–48. <https://doi.org/10.1145/3777385>.

6. Wu, S.; Sun, F.; Zhang, W.; Xie, X.; Cui, B. Graph Neural Networks in Recommender Systems: A Survey. *ACM Comput. Surv.* **2023**, *55*, 1–37. <https://doi.org/10.1145/3535101>.
7. Zeng, G.; Peng, H.; Li, A.; Wu, J.; Liu, C.; Yu, P.S. Scalable Semi-Supervised Clustering via Structural Entropy With Different Constraints. *IEEE Trans. Knowl. Data Eng.* **2024**, *37*, 478–492. <https://doi.org/10.1109/TKDE.2024.3486530>.
8. You, Y.; Chen, T.; Sui, Y.; Chen, T.; Wang, Z.; Shen, Y. Graph contrastive learning with augmentations. In *NIPS '20: Proceedings of the 34th International Conference on Neural Information Processing Systems, Vancouver, BC, Canada, 6–12 December 2020*; Curran Associates Inc.: Red Hook, NY, USA, 2020.
9. Yan, H.; Li, C.; Long, R.; Yan, C.; Zhao, J.; Zhuang, W.; Yin, J.; Zhang, P.; Han, W.; Sun, H.; et al. A comprehensive study on text-attributed graphs: benchmarking and rethinking. In *NIPS '23: Proceedings of the 37th International Conference on Neural Information Processing Systems, New Orleans, LA, USA, 10–16 December 2023*; Curran Associates Inc.: Red Hook, NY, USA, 2023.
10. Yang, Z.; Ding, M.; Huang, T.; Cen, Y.; Song, J.; Xu, B.; Dong, Y.; Tang, J. Does Negative Sampling Matter? A Review with Insights into its Theory and Applications. *IEEE Trans. Pattern Anal. Mach. Intell.* **2024**, *46*, 5692–5711. <https://doi.org/10.1109/TPAMI.2024.3371473>.
11. Yang, Z.; Shao, Z.; Dong, Y.; Tang, J. TriSampler: A Better Negative Sampling Principle for Dense Retrieval. *Proc. AAAI Conf. Artif. Intell. (AAAI)* **2024**, *38*, 9269–9277. <https://doi.org/10.1609/aaai.v38i8.28779>.
12. Hu, W.; Fey, M.; Zitnik, M.; Dong, Y.; Ren, H.; Liu, B.; Catasta, M.; Leskovec, J. Open Graph Benchmark: Datasets for Machine Learning on Graphs. In *Proceedings of the Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, Virtual, 6–12 December 2020*.
13. He, X.; Bresson, X.; Laurent, T.; Hooi, B. Explanations as Features: LLM-Based Features for Text-Attributed Graphs. *arXiv* **2023**, arXiv:2305.19523. <https://doi.org/10.48550/ARXIV.2305.19523>.
14. Tukey, J.W. The future of data analysis. In *Breakthroughs in Statistics: Methodology and Distribution*; Springer: Berlin/Heidelberg, Germany, 1962; pp. 408–452.
15. Myers, S.A.; Sharma, A.; Gupta, P.; Lin, J. Information network or social network?: The structure of the twitter follow graph. In *Proceedings of the 23rd International World Wide Web Conference, WWW '14, Seoul, Republic of Korea, 7–11 April 2014*; Association for Computing Machinery: New York, NY, USA, 2014; pp. 493–498. <https://doi.org/10.1145/2567948.2576939>.
16. Zhou, M.; Feng, W.; Zhu, Y.; Zhang, D.; Dong, Y.; Tang, J. Semi-Supervised Social Bot Detection with Initial Residual Relation Attention Networks. In *Proceedings of the Machine Learning and Knowledge Discovery in Databases; ECML PKDD 2023*; Springer: Cham, Switzerland, 2023. https://doi.org/10.1007/978-3-031-43427-3_13.
17. Kipf, T.N.; Welling, M. Semi-Supervised Classification with Graph Convolutional Networks. *arXiv* **2017**, arXiv:1609.02907.
18. Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Liò, P.; Bengio, Y. Graph Attention Networks. *arXiv* **2018**, arXiv:1710.10903.
19. Hamilton, W.L.; Ying, R.; Leskovec, J. Inductive Representation Learning on Large Graphs. *Adv. Neural Inf. Process. Syst. (NeurIPS)* **2017**, *30*, 1024–1034.
20. Bu, W.; Cao, X.; Zheng, Y.; Pan, S. Improving Augmentation Consistency for Graph Contrastive Learning. *Pattern Recognit.* **2024**, *148*, 110182.
21. Xu, B.; Wang, X.; Liu, Z.; Kang, L. A GAN combined with Graph Contrastive Learning for Traffic Forecasting. In *CNIOT '23: Proceedings of the 2023 4th International Conference on Computing, Networks and Internet of Things, Xiamen, China, 26–28 May 2023*; Association for Computing Machinery: New York, NY, USA, 2023; pp. 866–873.
22. Ji, C.; Xu, Y.; Lu, Y.; Huang, X.; Zhu, Y. Contrastive-Learning-Based Adaptive Graph Fusion Convolution Network with Residual-Enhanced Decomposition Strategy for Traffic Flow Forecasting. *IEEE Internet Things J.* **2024**, *11*, 20246–20259.
23. Chen, Z.; Mao, H.; Li, H.; Jin, W.; Wen, H.; Wei, X.; Wang, S.; Yin, D.; Fan, W.; Liu, H.; et al. Exploring the Potential of Large Language Models (LLMs) in Learning on Graphs. *SIGKDD Explor. Newsl.* **2024**, *25*, 42–61. <https://doi.org/10.1145/3655103.3655110>.
24. Liu, P.; Yuan, W.; Fu, J.; Jiang, Z.; Hayashi, H.; Neubig, G. Pre-train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing. *ACM Comput. Surv.* **2023**, *55*, 1–35. <https://doi.org/10.1145/3560815>.
25. Shin, T.; Razeghi, Y.; Logan, R.L., IV; Wallace, E.; Singh, S. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Online, 16–20 November 2020; pp. 4222–4235. <https://doi.org/10.18653/v1/2020.emnlp-main.346>.
26. Gao, T.; Fisch, A.; Chen, D. Making Pre-trained Language Models Better Few-shot Learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*; Association for Computational Linguistics: Online, 2021; pp. 3816–3830.
27. Qin, G.; Eisner, J. Learning How to Ask: Querying LMs with Mixtures of Soft Prompts. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Online, 6–11 June 2021; pp. 5203–5212. <https://doi.org/10.18653/v1/2021.naacl-main.410>.

28. Li, X.L.; Liang, P. Prefix-Tuning: Optimizing Continuous Prompts for Generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*; Association for Computational Linguistics: Online, 2021; pp. 4582–4597. <https://doi.org/10.18653/v1/2021.acl-long.353>.
29. Lester, B.; Al-Rfou, R.; Constant, N. The Power of Scale for Parameter-Efficient Prompt Tuning. *arXiv* **2021**, arXiv:2104.08691. <https://doi.org/10.48550/arXiv.2104.08691>.
30. Liu, X.; Zheng, Y.; Du, Z.; Ding, M.; Qian, Y.; Yang, Z.; Tang, J. GPT understands, too. *AI Open* **2024**, *5*, 208–215. <https://doi.org/https://doi.org/10.1016/j.aiopen.2023.08.012>.
31. Yang, Z.; Cohen, W.W.; Salakhutdinov, R. Revisiting semi-supervised learning with graph embeddings. In *ICML'16: Proceedings of the 33rd International Conference on International Conference on Machine Learning-Volume 48, New York, NY, USA, 19–24 June 2016*; JMLR.org, Cambridge, MA, USA: 2016; pp. 40–48.
32. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the NAACL-HLT, Minneapolis, MN, USA, 2–7 June 2019*.
33. Wolf, T.; Debut, L.; Sanh, V.; Chaumond, J.; Delangue, C.; Moi, A.; Cistac, P.; Rault, T.; Louf, R.; Funtowicz, M.; et al. Transformers: State-of-the-Art Natural Language Processing. In *Proceedings of the EMNLP: System Demonstrations, Online, 16–20 November 2020*.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.