Toward General and Robust LLM-enhanced Text-attributed Graph Learning

Anonymous ACL submission

Abstract

Recent advancements in Large Language Models (LLMs) and the proliferation of Text-Attributed Graphs (TAGs) across various domains have positioned LLM-enhanced TAG learning as a critical research area. However, the field faces significant challenges: (1) the absence of a unified framework to systematize the diverse optimization perspectives, and (2) the lack of a robust method capable of handling real-world TAGs, which often suffer from texts and edge sparsity, leading to suboptimal performance.To address these challenges, we propose UltraTAG, a unified pipeline for LLM-enhanced TAG learning. UltraTAG provides a unified comprehensive and domain-adaptive framework in the field. Building on this framework, we propose UltraTAG-S, a robust instantiation of UltraTAG designed to tackle the inherent sparsity issues in real-world TAGs with the technology of LLM-based text propagation, text augmentation, and edge reconfiguration strategies. Our extensive experiments demonstrate that UltraTAG-S significantly outperforms existing baselines, achieving improvements of 2.12% and 17.47% in ideal and sparse settings, respectively. Moreover, as the data sparsity ratio increases, the performance improvement of UltraTAG-S also rises.

1 Introduction

In recent years, the advancements in large language models (LLMs) (Brown et al., 2020) have driven the evolution of graph ML, particularly in Text-Attributed Graphs (TAGs) (He et al., 2024a), which combine nodes, edges, and textual data for applications in social networks, recommendation systems etc. While graph neural networks (GNNs) (Li et al., 2024a) excel at capturing structural information, they struggle with textual data, necessitating the integration of GNNs and LLMs for TAG learning (Zhu et al., 2024; Duan et al., 2023). Despite progress, existing TAG learning methods still face several limitations:

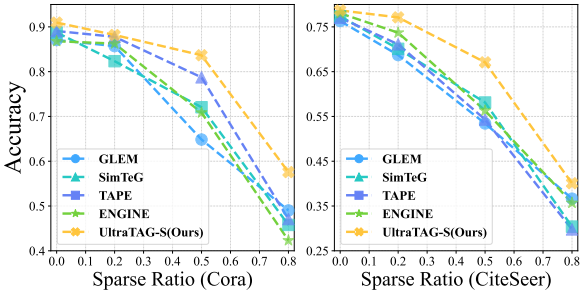

Figure 1: Performance of diffirent LLM-enhanced TAG learning methods in sparse scenarios.

Limitation 1: Lack of a unified LLM-enhanced TAG learning framework. As for the current innovation directions of LLM-enhanced TAG learning are disorganized, we recapitulate them from a new perspective: (1) *Preprocessing: Data Augmentation* (He et al., 2024a; Chen et al., 2024; Wang et al., 2024; Pan et al., 2024), which leverages LLMs to generate enhanced textual representations like soft labels for text augmentation. (2) *Feature Engineering: Improved Text Encoder* (Chien et al., 2022; Duan et al., 2023), which uses LLMs/LMs to enhance node feature representation. (3) *Training: Joint Training Mechanism* (Zhao et al., 2023; Zhu et al., 2024; Wen and Fang, 2023; Huang et al., 2024), which enhance performance by interactive training mechanism between GNNs and LMs. However, the diverse optimization strategies and goals without a systematic standard hinder unified objectives, slowing progress in TAG learning.

Solution 1: UltraTAG: A Unified Pipeline toward General and Robust LLM-enhanced TAG Learning. To address Limitation 1, we propose UltraTAG, as detailed in Sec. 3. UltraTAG is composed of three modules: Data Augmentation, Text Encoder, and Training Mechanism, as shown in Figure 2. The modules integrate three directions of LLM-enhanced TAG learning, translating innovative methods into specific optimization objectives.

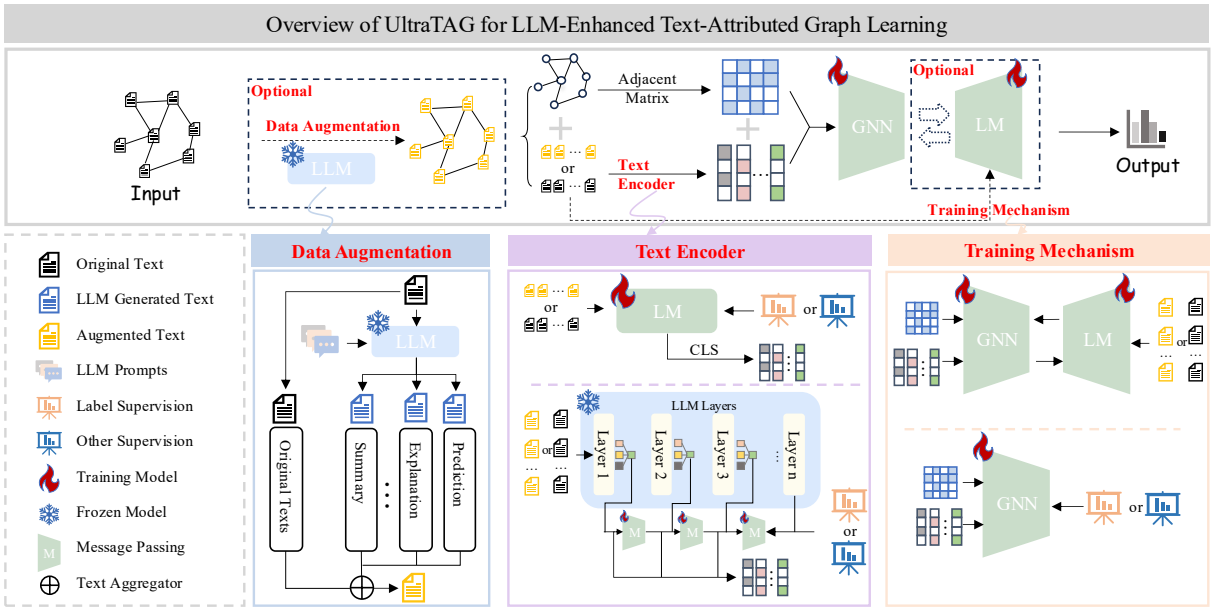

Figure 2: Overview of UltraTAG for LLM-Enhanced Text-Attributed Graph Learning with three modules.

Limitation 2: Lack of a Robust Method. In real-world TAGs, data sparsity in nodes and edges is a common issue. For example, privacy measures (Li et al., 2024b) on social networks may restrict access to users' information. Developing robust methods that maintain performance under such sparse conditions is a challenge. Current approaches often depend on complete text attributes, making them incompatible with sparse graphs and leading to sub-optimal results. This robustness focus is specific to sparsity, which involves only missing nodes or edges, but not including data noise with error of node's text, edge or corresponding label.

Solution 2: UltraTAG-S: An Instance of UltraTAG for Sparse Scenarios. To address Limitation 2, we propose UltraTAG-S, as detailed in Sec. 4. UltraTAG-S is composed of three key modules: (1)LLM-based Robustness Enhancement, (2)LM-based Resilient Representation Learning, and (3)Graph-Enhanced Robust Classifier, as illustrated in Figure 3. To simulate real-world sparse scenarios, we randomly remove node texts and edges from the graph according to a certain ratio. Module 1 includes edge-based text propagation and LLM-based text enhancement to address node sparsity. Module 2 designs a PageRank-based node selector and a LLM-based edge predictor to handle edge sparsity. Module 3 incorporates a graph structure learning module to further enhance robustness.

Our Contributions: (1) *A Unified Framework.* We adopt a novel perspective to systematically examine all existing methods for TAG learning and introduce UltraTAG, a unified and domain-adaptive paradigm which can extend to UltraTAG-X. (2) *A Robust Method.* Expanding on UltraTAG, we propose UltraTAG-S, a robust TAG learning framework designed specifically for sparse scenarios. (3) *SOTA Performance.* Our proposed UltraTAG-S achieves SOTA performance and optimal robustness in evaluations among 7 datasets spanning four distinct domains not only in ideal but also in sparse scenarios, exhibiting minimal performance degradation, as shown in Figure 1.

2 Related Works

2.1 Graph Learning for Data Sparsity Scenarios

For graph learning in sparse scenarios, existing research primarily focuses on addressing missing node representations, edge absences, or label deficiencies (Rossi et al., 2022; Guo et al., 2023; Zhang et al., 2022). Most of them employ vector completion based on graph propagation or attention to handle these issues. However, there is still a lack of targeted research on sparse scenarios of TAGs.

2.2 Shallow Embedding Methods for TAG Learning

TAG learning commonly uses shallow embeddings (e.g., skip-gram (Mikolov et al., 2013) or BoW (Harris, 1954)) as inputs for GCNs (Kipf and Welling, 2017). While simple and efficient, they fail to capture complex semantics and nuanced relationships with limited effectiveness.

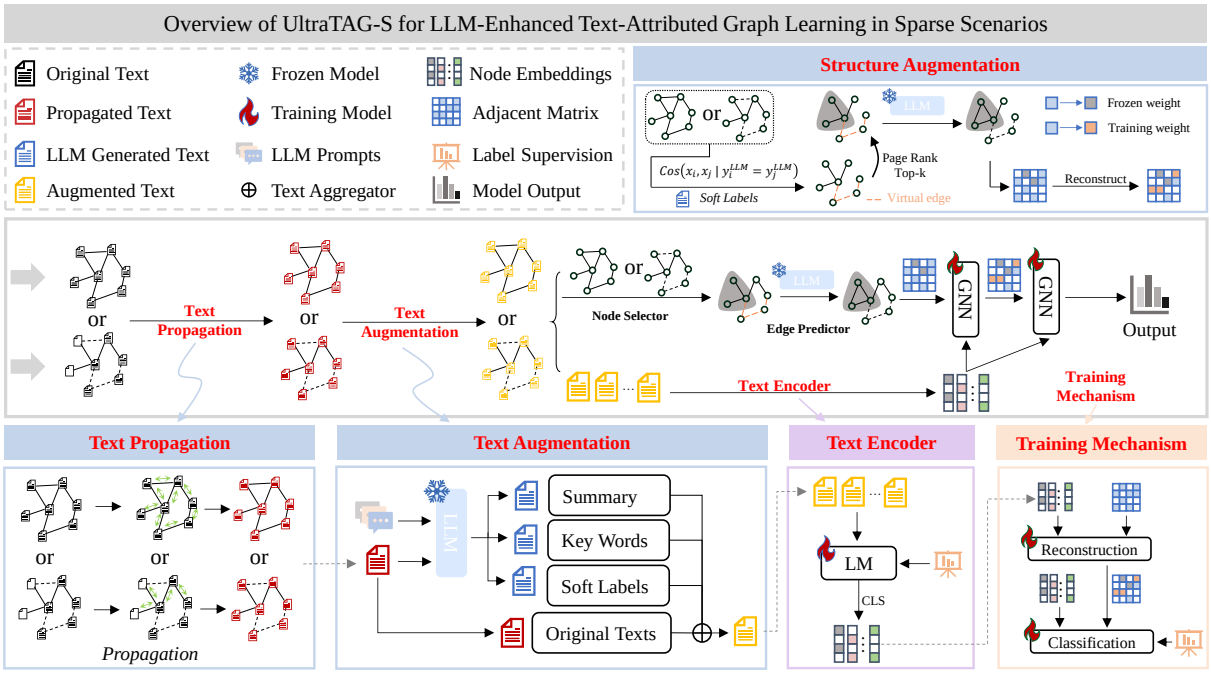

Figure 3: Overview of UltraTAG-S for LLM-Enhanced Text-Attributed Graph Learning in Sparse Scenarios.

2.3 LM/LLM-based Methods for TAG Learning

With the rise of LMs like BERT (Devlin et al., 2019), researchers encode textual information in TAGs by fine-tuning LMs on downstream tasks (Zhao et al., 2023; Duan et al., 2023) or aligning LM and GNN via custom loss functions (Wen and Fang, 2023). The emergence of LLMs like GPT-3 (Brown et al., 2020) has further advanced TAG learning, focusing on: (1) text enhancement (e.g., better node descriptions, labels) (He et al., 2024a; Wang et al., 2024; Pan et al., 2024; He et al., 2024b), and (2) superior text encoding for node representations (Zhu et al., 2024; Huang et al., 2024), collectively boosting performance.

3 UltraTAG

In this section, we provide details about three modules of UltraTAG shown in Figure 2: Data Augmentation, Text Encoder and Training Mechanism.

3.1 Notations

Given a TAG $\mathcal{G} = \{\mathcal{V}, \mathcal{T}, \mathcal{A}, \mathcal{Y}\}$, where $\mathcal{V}$ is the set containing N nodes, $\mathcal{T}$ is the set of texts, for $i \in \mathcal{V}$, $t_i \in \mathcal{T}$ is the text attribute of node i. $\mathcal{A} \in \mathbb{R}^{N \times N}$ is the adjacency matrix and $\mathcal{Y}$ is ground-truth labels.

This study focuses on the TAG node classification task. The dataset is split into training nodes $\mathcal{V}_{tr}$ with training labels $\mathcal{Y}_{tr}$ and testing nodes $\mathcal{V}_{te}$ with testing labels $\mathcal{Y}_{te}$. A model f_{θ^*} is trained on $\mathcal{V}_{tr}$ and tested on $\mathcal{V}_{te}$ to generate predictions. The optimization objective is formalized as:

$$f_{\theta^*} = \underset{\theta}{\operatorname{argmax}} \mathbb{E}_{n \in \mathcal{V}_{tr}} P_\theta(\hat{y}_n = y_n \mid n), \quad (1)$$

where y_n is ground-truth and $\hat{y}_n$ is prediction.

3.2 Data Augmentation

TAGs rely solely on node texts, not representations, making text preprocessing crucial. To enhance text representation, data augmentation from a natural language perspective is effective. Leveraging LLM's capabilities, we input $\mathcal{T}$ and generate augmented texts $\mathcal{T}'$ using varied prompts $\mathcal{P}$:

$$\mathcal{T}' = \{t_i' \mid t_i' = \text{LLM}(\mathcal{P}, t_i, \alpha), \forall t_i \in \mathcal{T}\}, \quad (2)$$

where α is frozen parameters of LLM.

Then, $\mathcal{T}'$ is typically aggregated with $\mathcal{T}$ to produce the final textual representation $\mathcal{T}^*$:

$$\mathcal{T}^* = \{t_i^* \mid t_i^* = \text{Agg}(t_i', \{t_j' \mid j \in \mathcal{N}_i\})\}, \quad (3)$$

where Agg is the text aggregator, which include selection and concatenation and so on, $\mathcal{N}_i$ is the set of neighbor nodes of node i.

3.3 Text Encoder

The given nodes' texts $\mathcal{T}_{tr}^*$ associated with the training nodes must be encoded into embeddings to facilitate subsequent model processing which can be efficiently accomplished using LMs or LLMs.

LMs as Encoder. Text encoding typically employs LMs like BERT (Devlin et al., 2019). Fine-tuning LMs on downstream tasks enhances their task-specific encoding capability. As for $t_i^* \in \mathcal{T}_{tr}^*$, this process can be described as:

$$h_i = \text{LM}(t_i^*, \theta_{\text{LM}}) \in \mathbb{R}^d, \forall t_i^* \in \mathcal{T}_{tr}^*, \quad (4)$$

where h_i is the output of the LM, we train the parameters by adding an MLP after the LM.

Meanwhile, various downstream tasks can be used to do it, such as node classification (Duan et al., 2023) or others (Chien et al., 2022).

LLMs as Encoder. Leveraging LLM's language understanding capabilities, their features from different layers capture varying abstraction levels with versatile representations (Zhu et al., 2024). Inspired by it, for each node's text $t_i^* \in \mathcal{T}_{tr}^*$, we can get $h_i^1, h_i^2, h_i^3, ..., h_i^l$ from different LLM layers:

$$h_i^1, h_i^2, h_i^3, ..., h_i^l = \text{LLM}(t_i^*, \theta_{\text{LLM}}) \in \mathbb{R}^d, \quad (5)$$

where $h_i^j, j \in [1, l]$ denotes the output vector representation of LLM layer j of node i.

Then, we train a multi-layer GNN to simulate the propagation process of multi-layer representations in the LLM with cross-entropy loss:

$$\theta^* = \arg\min_{\theta} \sum_{i=1}^{N} \sum_{j=1}^{L} \text{CE}(\mathcal{M}(h_i^j, \mathcal{A}; \theta), y_i), \quad (6)$$

where $\mathcal{M}$ denotes the message passing module of GNN, $\mathcal{A}$ is the adjacent matrix.

3.4 Training Mechanism

After obtaining the nodes' textual representations $\mathcal{H} = \{h_1, h_2, h_3, ..., h_N\}$ and adjacency matrix $\mathcal{A}$, input of them into a GNN will yield the final prediction. We can use a simple GNN module, or combine GNN with LM for joint training.

Simple GNN. A simple GNN produces final predictions through downstream task training:

$$\theta^* = \arg\min_{\theta} \sum_{i=1}^{N} \text{CE}(\text{GNN}(h_i, \mathcal{A}; \theta), y_i), \quad (7)$$

GNN with LM. For the combination of GNN and LM training, the pseudo-labels $\mathcal{Y}_{\text{G}}$ generated by GNN guide LM training, and the pseudo-labels generated by LM $\mathcal{Y}_{\text{L}}$ guide GNN training, and the cycle repeats with the same downstream task:

$$\mathcal{Y}_{\text{L}} = \text{GNN}(\mathcal{H}, \mathcal{A}; \theta_{\text{G}}), \mathcal{Y}_{\text{G}} = \text{LM}(\mathcal{T}; \theta_{\text{L}}), \quad (8)$$

4 UltraTAG-S

To address the challenge of data sparsity of TAGs in real-word applications, we creatively propose UltraTAG-S as shown in Figure 3, which is composed of three modules: LLM-based Robustness Enhancement, LM-based Resilient Representation Learning and Graph-Enhanced Robust Classifier.

4.1 LLM-based Robustness Enhancement

Data Augmentation module can be divided into Text Propagation, Text Augmentation and Structure Augmentation.

Text Propagation. Leveraging the homophily principle in graph theory, we posit that adjacent nodes exhibit textual similarity. Inspired by the message-passing mechanism in GNNs, we propagate textual information from neighboring nodes to reconstruct missing text attributes.

Specifically, for node $v_i \in \mathcal{V}$, $t_i \in \mathcal{T}$ and its neighbors $\mathcal{N}_i$, propagated texts $\mathcal{T}^{'}$ are obtained by:

$$\mathcal{T}^{'} = \{t_i^{'} \mid t_i^{'} = t_i \oplus \{t_j \mid j \in \mathcal{N}_i\}, \forall t_i \in \mathcal{T}\}, \quad (9)$$

where $\oplus$ denotes concatenation of neighbors' texts.

Text Augmentation. Leveraging the advanced language comprehension capabilities of LLM, we utilize prompt engineering to extract critical textual information and enrich data representations.

Specifically, for the propagated text $t_i^{'} \in \mathcal{T}^{'}$, we get the augmented text $\mathcal{T}^*$ with different prompts:

$$\mathcal{T}_{\text{Su}} = \{t_i^{''} \mid t_i^{''} = \text{LLM}(\mathcal{P}_{\text{Su}}, t_i^{'}, \theta_{\text{LLM}}), \quad (10)$$

$$\mathcal{T}_{\text{KW}} = \{t_i^{''} \mid t_i^{''} = \text{LLM}(\mathcal{P}_{\text{KW}}, t_i^{'}, \theta_{\text{LLM}}), \quad (11)$$

$$\mathcal{Y}_{\text{SL}} = \{t_i^{''} \mid t_i^{''} = \text{LLM}(\mathcal{P}_{\text{SL}}, t_i^{'}, \theta_{\text{LLM}}), \quad (12)$$

$$\mathcal{T}^* = \text{AGG}(\mathcal{T}^{'}, \mathcal{T}_{\text{Su}}, \mathcal{T}_{\text{KW}}, \mathcal{Y}_{\text{SL}}), \quad (13)$$

where AGG denotes the text aggregation module of concatenation, $\mathcal{T}_{\text{Su}}$, $\mathcal{T}_{\text{KW}}$, $\mathcal{Y}_{\text{SL}}$ are the Summary, Key Words and Soft Labels generated by LLMs respectively. $\mathcal{P}_{\text{Su}}$, $\mathcal{P}_{\text{KW}}$, $\mathcal{P}_{\text{SL}}$ are the prompts.

Structure Augmentation. To mitigate edge sparsity, we introduce a Structure Augmentation module composed of Virtual Edge Generator, Node Selector and Edge Reconfigurator. This module leverages LLMs to re-identify edges for selected nodes, thereby optimizing the graph structure.

a. Virtual Edge Generator. In order to ensure the integrity of the graph structure before node selection, we use the soft labels $\mathcal{T}_{\text{SL}}$ generated by

LLMs and calculate the similarity with the same soft label, which can be described as:

$$h_i = \text{LM}(t_i^*, \alpha_{\text{LM}}), h_j = \text{LM}(t_j^*, \alpha_{\text{LM}}), \quad (14)$$

$$\mathcal{S}_{ij} = cos(h_i, h_j), \forall y_i = y_j \ \ \& \ \ y_i, y_j \in \mathcal{Y}_{\text{SL}}. \quad (15)$$

The adjacency matrix is updated to $\mathcal{A}^{'}$:

$$\mathcal{A}_{ij}^{'} = \begin{cases} 1, & \text{if } \mathcal{A}_{ij} = 1 | \mathcal{S}_{ij} > \tau_1, \\ 0, & \text{else}, \end{cases} \quad (16)$$

where $\mathcal{A}$ denotes the adjacency matrix with virtual edges after sparse process, τ_1 denotes the similarity threshold for edges to add.

b. Node Selector. Considering the impracticality of re-judging all edges, we design a node selector to select important nodes set $\mathcal{V}_c$. We calculate the pagerank score for each node in $\mathcal{V}$ and use these scores as importance score by following:

$$\text{Score}(v_i) = \text{PageRank}(v_i, \mathcal{A}^{'}), \quad (17)$$

$$\mathcal{V}_c = \{v_i \mid \text{Score}(v_i) > \text{Score}(v_k)\}, \quad (18)$$

where v_k denotes the node with k-th largest node importance score calculated by PageRank algorithm with original edges and virtual edges.

c. Edge Reconfigurator. For each edge in the complete graph of $\mathcal{V}_c$, we use LLM with prompt detailed in Appendix F to re-determines its existence with the confidence score $\mathcal{C}_{ij}$ of edge e_{ij}:

$$\mathcal{C}_{ij} = \text{LLM}(\mathcal{P}_{\text{edge}}, t_i^*, t_j^*), \forall v_i, v_j \in \mathcal{V}_c, \quad (19)$$

The updated adjacency matrix $\mathcal{A}^*$ is expressed as:

$$\mathcal{A}_{ij}^* = \begin{cases} \mathcal{A}_{ij}, & \text{if } v_i \notin \mathcal{V}_c \mid v_j \notin \mathcal{V}_c; \\ 1, & \text{if } \mathcal{C}_{ij} > \tau_2; \quad 0, \text{ else}; \end{cases} \quad (20)$$

where τ_2 is the confidence threshold for LLM edge reconfiguration in Equation 19.

4.2 LM-based Resilient Representation Learning

After augmenting the graph $\mathcal{G}^* = \{\mathcal{V}, \mathcal{T}^*, \mathcal{A}^*, \mathcal{Y}\}$, we fine-tune the language model on downstream tasks of node classification. Specifically, node i's text t_i^* is passed through the fine-tuned language model LM_θ to output the feature representation for downstream node classification. The process can be described as following equation:

$$\hat{y}_i = \text{softmax}(W \cdot \text{LM}(t_i, \theta) + b), \forall\, t_i^* \in \mathcal{T}^*, \quad (21)$$

where W is the weight matrix, b is the bias term.

After fine-tuning with the following negative log-likelihood Loss $\mathcal{L}_{ft}$, the node representations $\mathcal{H}$ are calculated by following equations:

$$h_i = \text{LM}(t_i^*, \theta^*), \quad (22)$$

$$\mathcal{L}_{ft} = -\frac{1}{N} \sum_{i=1}^{N} \sum_{k=1}^{K} y_{i,k} \log \hat{y}_{i,k}, \quad (23)$$

where N, K are the number of training nodes and classes, $y_{i,k}$ is the ground-truth, $\hat{y}_{i,k}$ is the output.

4.3 Graph-Enhanced Robust Classifier

We employ a dual-GNN framework to tackle edge sparsity: one GNN learns enhanced graph structures, and the other focuses on node classification.

Specifically, with the nodes' representations$\mathcal{H} \in \mathbb{R}^{N \times d}$ and the structure representation $\mathcal{A}^* \in \mathbb{R}^{N \times N}$, we first compute the similarity matrix of node vector representations:

$$\mathbf{S} = \text{Norm}(\mathcal{H}^{(1)} \cdot \mathcal{H}^{(1)^\top}), \mathcal{H}^{(1)} = \text{GNN}_1(\mathcal{H}, \mathcal{A}^*). \quad (24)$$

Then, we update the original adjacency matrix $\mathcal{A}^*$ with preserving the judgment of the LLM without alteration:

$$\tilde{\mathcal{A}}_{ij}^* = \begin{cases} \mathcal{A}_{ij}^* + \mathbf{S}_{ij}, & \text{if } v_i \notin \mathcal{V}_c \mid v_j \notin \mathcal{V}_c, \\ \mathcal{A}_{ij}^*, & \text{else}. \end{cases} \quad (25)$$

We use the updated matrix as the input of $\text{GNN}_2(\cdot)$ and jointly optimize $\text{GNN}_1(\cdot)$ and $\text{GNN}_2(\cdot)$ using cross entropy loss $\mathcal{L}_{GNN}$:

$$\mathcal{H}^{(2)} = \text{GNN}_2(\mathcal{H}, \tilde{\mathcal{A}}^*), \quad (26)$$

$$\mathcal{L}_{GNN} = -\frac{1}{N} \sum_{i=1}^{N} \sum_{k=1}^{K} y_{i,k} \log \hat{y}_{i,k}, \quad (27)$$

where N is nodes' number, and K is classes' one.

5 Experiments

In this section, we analyze the effectiveness of UltraTAG-S through experimental evaluation. To comprehensively assess the performance of our approach, we address the following questions: ***Q1:*** What are the differences between existing TAGs learning methods under UltraTAG? ***Q2:*** What is the performance of UltraTAG-S as a general and robust TAGs learning paradigm in ideal and sparse scenarios? ***Q3:*** What factors contribute to the performance and robustness of UltraTAG-S? ***Q4:*** What is the training time complexity of UltraTAG-S? Details of the datasets and baselines are in Appendix A and B, respectively.

Table 1: The Comparison of Different LLM-enhanced TAG Learning Methods under UltraTAG. The top four methods use LMs, while the bottom five use LLMs. 'XMC' is eXtream Multi-label Classification, 'Iteration' is iterative training with LM and GNN using pseudo labels, 'Joint' means joint training with multiple GNNs.

Method	Data Augmentation	Text Encoder	Encoder Supervision	Traing Mechanism
GLEM	✗	DeBERTa	Node Classification	Iteration
GIANT	✗	BERT	XMC	Only GNN
G2P2	✗	RoBERTa	Node Classification	Combined Loss
SimTeG	✗	e5-large/RoBERTa	Node Classification	Only GNN
TAPE	✓	DeBERTa	Node Classification	Only GNN
ENGINE	✗	LLaMA2-7B	/	Joint
LLMGNN	✓	BoW	/	Only GNN
GraphAdapter	✗	LLaMA2-13B	Token Prediction	Only GNN
UltraTAG-S	✓	BERT	Node Classification	Joint

Table 2: The Comparison of Different LLM-enhanced TAG Learning Methods for Sparse Scenarios Robustness from Four Dimensions of Robustness.

Robustness	Input	Node	Edge	Training
GLEM	✗	✗	✗	✗
GIANT	✗	✗	✗	✗
G2P2	✓	✓	✗	✗
SimTeG	✗	✗	✗	✗
TAPE	✓	✗	✗	✗
ENGINE	✗	✗	✗	✓
LLMGNN	✓	✗	✓	✗
GraphAdapter	✓	✗	✗	✓
UltraTAG-S	✓	✓	✓	✓

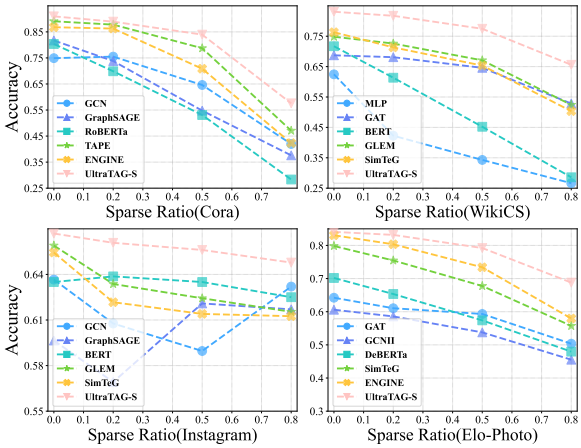

Figure 4: Robustness Comparison in Sparse Scenarios.

5.1 Paradigm Comparision

In this section, we compare the similarities and differences of the current LLM-enhanced TAG learning methods under the framework of UltraTAG from four aspects, namely Data Augmentation, Text Encoder, Encoder Supervision, and Training Mechanism, as shown in Table 1. LM-based methods (Zhao et al., 2023; Chien et al., 2022; Wen and Fang, 2023; Duan et al., 2023) utilize distinct language models and fine-tuning tasks, while LLM-based methods (He et al., 2024a; Chen et al., 2024) focus on data augmentation. Meanwhile, anyone can optimize one small module among the four modules of UltraTAG to form a new baseline.

We also compare the sparse scenarios robustness of Different LLM-enhanced TAG Learning Methods, which means whether these methods consider the robustness of input data, missing texts, missing edges and training. The comparison is shown as Table 2, we find that only UltraTAG-S(Ours) takes into account the robustness across all dimensions.

5.2 Performance and Robustness Analysis

We conduct a comprehensive evaluation of UltraTAG-S by comparing with GNN-only, LM-only and LLM-GNN methods, as the results in Table 3. Since GNN-only methods cannot accept texts as input, in order to make a fair comparison, we encode these methods using a unified BERT (Devlin et al., 2019) to get unified representation as input. As can be seen from Table 3, the performance of UltraTAG-S on all datasets is better than that of the current existing methods, and the improvement of the effect is up to 2.21%.

In order to simulate the challenges of the sparse scene, we randomly delete the texts and edges of nodes in a ratio of 20%, 50%, and 80% without considering data noise or additional constraints. As illustrated in Figure 4, our proposed method, UltraTAG-S, demonstrates the best robustness compared with current TAG learning baselines. Specifically, our method maintains the smallest decline in classification accuracy under sparse scenarios.

Table 3: Experimental results of node classification, optimal performance in **bold** and sub-optimal in underlined.

Methods	Cora	CiteSeer	PubMed	WikiCS	Instagram	Reddit	Elo-Photo
MLP	54.94±3.68	61.91±1.67	52.13±7.35	62.46±0.70	51.85±10.78	52.91±1.81	47.45±9.18
GCN	74.91±8.71	69.00±2.83	72.54±6.95	73.27±4.62	63.66±1.11	55.00±4.34	69.49±3.93
GAT	71.70±2.75	70.31±1.01	75.86±1.08	68.72±0.90	64.80±0.22	60.36±0.25	64.22±2.58
GCNII	77.23±0.66	71.91±1.05	73.28±1.67	70.12±1.81	65.07±0.59	62.78±0.49	60.60±1.36
GraphSAGE	81.70±1.00	66.68±0.80	68.41±9.59	75.16±0.33	59.65±5.78	53.59±2.24	70.48±6.03
BERT	79.70±0.32	76.88±0.41	90.95±0.11	71.70±1.09	63.50±0.09	58.78±0.05	70.01±0.08
DeBERTa	73.39±4.54	75.16±1.08	90.81±0.20	68.18±4.10	62.40±0.59	59.92±0.45	70.18±0.18
RoBERTa	80.35±0.48	77.04±1.49	91.13±0.11	72.12±0.70	64.67±0.34	59.23±0.06	70.25±0.34
GLEM	87.07±1.01	76.30±2.45	89.56±1.65	74.83±0.95	65.90±0.36	60.88±0.03	77.74±0.27
SimTeG	88.75±0.42	77.37±0.64	88.31±0.75	76.32±0.53	64.29±0.19	61.60±0.88	79.82±0.21
TAPE	89.07±0.56	77.02±0.71	90.38±0.99	80.17±0.18	65.44±0.35	63.01±0.82	82.26±0.64
ENGINE	86.79±0.58	78.03±0.48	91.43±0.13	81.38±0.38	66.27±0.41	62.57±0.13	83.06±0.22
UltraTAG-S	**90.96±0.45**	**78.68±0.21**	**92.41±0.30**	**83.05±0.16**	**66.69±0.14**	**63.78±0.30**	**84.70±0.03**

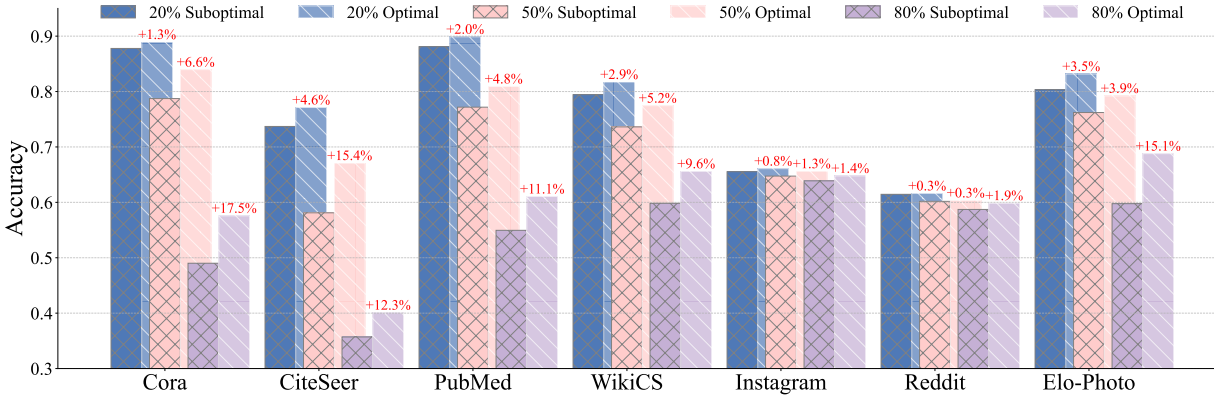

Figure 5: Robustness Comparison among All Datasets in Sparse Ratio of 20%, 50% and 80%.

The details of performance in sparse ratio of 80% is shown in Table 4. As can be seen from the results in sparse scenarios, our proposed UltraTAG-S can also achieve SOTA node classification accuracy in extremely sparse scenarios 80%, and the performance enhancement of UltraTAG-S is up to 17.5% in sparse ratio of 80%. The details with sparse ratio of 20% and 50% are shown in Appendix E Table 7, 8. As shown in Figure 5, UltraTAG-S consistently achieves the highest accuracy in all data sets at varying sparsity levels, demonstrating optimal robustness. The robustness improves significantly as data sparsity increases, highlighting the effectiveness in extreme data sparsity.

5.3 Ablation Study

In this part, we perform an ablation study on the CiteSeer, and PubMed datasets to verify the effectiveness and robustness of UltraTAG-S, particularly in sparse scenarios. The results of PubMed and CiteSeer are in Figure 6, more results of ablation and backbones are in Appendix G.

Specifically, the Text Augmentation module enhances the model's ability to generalize by introducing diverse textual variations, leading to improvement of up to 16.89% on CiteSeer and 55.45% on PubMed. This module is particularly effective in scenarios where textual diversity is limited, as it enriches the input data and reduces overfitting. The Structure Augmentation module further contributes to the model's robustness by optimizing the graph structure, achieving improvements of 3.07% on CiteSeer and 5.90% on PubMed. As for the Structure Learning module, it demonstrates even more substantial gains, with improvements of 32.49% on CiteSeer and 40.09% on PubMed, highlighting its ability to capture complex relationships in graph. It is evident that the Structure Learning module plays the most significant role in enhancing both the effectiveness and robustness, as it not only improves accuracy but also ensures stable performance across varying data conditions. These results underscore the importance of combining these modules to achieve optimal performance.

Table 4: Robustness Comparison in Sparse Scenarios with Ratio of 80%, which means nodes' texts and edges with proportion of 80% are removed randomly to simulate real-world scenario. Optimal performance is in **bold** and sub-optimal performance is underlined.

Methods	Cora	CiteSeer	PubMed	WikiCS	Instagram	Reddit	Elo-Photo
MLP	30.41±0.59	27.74±0.59	44.25±1.68	26.64±1.17	62.54±1.02	50.93±0.41	46.21±0.61
GCN	41.96±0.73	30.53±0.68	49.68±1.31	54.24±2.29	63.20±0.31	54.38±1.88	51.27±0.33
GAT	38.86±0.59	30.50±0.27	52.03±0.30	52.85±0.71	63.17±0.86	56.41±0.25	50.35±0.85
GCNII	36.79±0.28	31.07±0.97	51.33±0.37	50.83±1.32	62.27±0.84	57.62±1.10	45.53±0.15
GraphSAGE	37.60±0.67	31.69±0.44	50.59±0.71	53.03±0.71	61.70±0.93	58.72±0.81	52.17±0.65
BERT	37.59±0.08	31.50±0.54	49.95±0.04	28.58±1.24	62.50±0.43	51.40±0.15	49.59±0.04
DeBERTa	29.98±1.09	30.80±0.55	42.34±4.11	21.83±1.06	63.59±0.27	50.24±0.28	47.96±1.47
RoBERTa	28.23±0.00	23.32±4.55	47.24±4.20	20.36±0.40	63.68±0.00	50.32±0.21	49.52±0.12
GLEM	49.01±0.58	36.64±1.46	51.48±0.54	52.41±0.76	61.54±0.56	50.82±1.04	56.25±2.14
SimTeG	45.78±0.22	30.40±0.66	54.95±0.61	50.35±0.72	60.61±0.16	58.08±0.12	55.73±0.84
TAPE	47.08±0.20	29.77±0.28	54.87±0.50	59.83±0.77	61.25±0.59	58.10±0.72	59.76±0.12
ENGINE	42.32±0.66	35.70±0.19	54.74±0.09	49.42±0.45	63.88±0.20	57.54±0.77	57.96±0.13
UltraTAG-S	**57.57±1.38**	**40.08±0.45**	**61.05±0.49**	**65.60±0.34**	**64.78±0.67**	**59.85±0.01**	**68.79±0.07**

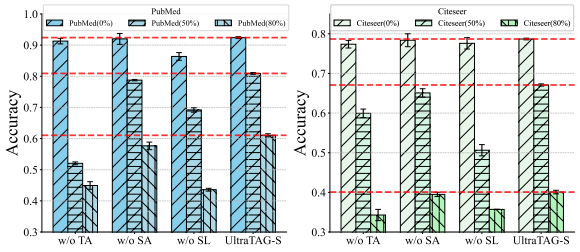

Figure 6: Ablation Study on PubMed and CiteSeer. The x-axis represents the modules in the ablation study, where 'w/o TA', 'w/o SA', 'w/o SL' denote the removal of Text Augmentation module, Structure Augmentation module and Structure Learning module, respectively. The y-axis represents accuracy in different ratios.

5.4 Complexity Analysis

Table 5: The comparison of different methods in downstream GNN training time per epoch.

	Cora	PubMed	WikiCS
SimTeG	0.142s	1.564s	2.109s
GLEM	0.131s	1.034s	1.645s
ENGINE	0.290s	2.165s	2.730s
UltraTAG-S	**0.015s**	**0.170s**	**0.845s**

The computational complexity of our proposed method is primarily determined by two GNN operations. The first GNN calculates the similarity matrix $\mathbf{S} \in \mathbb{R}^{N \times N}$ and updates the adjacency matrix $\mathcal{A}^*_{ij}$. This step involves pairwise computations between nodes, leading to a complexity of $O(N^2)$. The second GNN performs node classification using the updated adjacency matrix $\tilde{\mathcal{A}}^*_{ij}$ and node features $\mathcal{H}$. With m layers and $\mathcal{E} = O(N^2)$ edges in the graph, the complexity for this operation scales as $O(m \cdot N^2)$. Therefore, the total computational cost per epoch is dominated by these two steps, resulting in an overall complexity of $O(m \cdot N^2)$.

Experimentally, compared to other TAG learning methods, UltraTAG-S demonstrates a significant advantage in training time for downstream tasks. As shown in Table 5, under non-sparse experimental settings, it achieves up to a 19× speedup per training epoch over the suboptimal method.

6 Conclusion and Future Work

In response to the current LLM-enhanced TAG Learning methods, we first propose UltraTAG as a unified and domain-adaptive pipeline learning framework. Simultaneously, to address the challenges faced by existing LLM-enhanced TAG learning methods in real-world sparse scenarios, such as nodes' texts missing or edges missing, we introduce UltraTAG-S, a TAG learning paradigm specifically tailored for sparse scenarios. UltraTAG-S effectively resolves the issues of nodes' texts sparsity and edge sparsity in real-world settings through LLM-based text propagation strategy and text augmentation strategy, as well as PageRank and LLM-based graph structure learning strategies, achieving state-of-the-art performance in both ideal and sparse scenarios. In the future, we will further explore the pivotal role of text propagation strategies in TAG representation learning.

Limitations

While UltraTAG-S effectively handles sparse TAG learning, its performance depends on LLM quality, incurs higher computational costs. Meanwhile, our consideration of real-world scenarios has been limited to sparsity, while more complex real-world data scenarios remain to be explored.

References

AI@Meta. 2024. Llama 3 model card.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, and 12 others. 2020. Language models are few-shot learners. Preprint, arXiv:2005.14165.

Ming Chen, Zhewei Wei, Zengfeng Huang, Bolin Ding, and Yaliang Li. 2020. Simple and deep graph convolutional networks. In International Conference on Machine Learning, ICML.

Zhikai Chen, Haitao Mao, Hongzhi Wen, Haoyu Han, Wei Jin, Haiyang Zhang, Hui Liu, and Jiliang Tang. 2024. Label-free node classification on graphs with large language models (llms). Preprint, arXiv:2310.04668.

Eli Chien, Wei-Cheng Chang, Cho-Jui Hsieh, Hsiang-Fu Yu, Jiong Zhang, Olgica Milenkovic, and Inderjit S Dhillon. 2022. Node feature extraction by self-supervised multi-scale neighborhood prediction. Preprint, arXiv:2111.00064.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. Preprint, arXiv:1810.04805.

Keyu Duan, Qian Liu, Tat-Seng Chua, Shuicheng Yan, Wei Tsang Ooi, Qizhe Xie, and Junxian He. 2023. Simteg: A frustratingly simple approach improves textual graph learning. Preprint, arXiv:2308.02565.

C Lee Giles, Kurt D Bollacker, and Steve Lawrence. 1998. Citeseer: An automatic citation indexing system. In Proceedings of the third ACM conference on Digital libraries.

Dongliang Guo, Zhixuan Chu, and Sheng Li. 2023. Fair attribute completion on graph with missing attributes. Preprint, arXiv:2302.12977.

Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. Advances in Neural Information Processing Systems, NeurIPS.

Zellig S. Harris. 1954. Distributional structure.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. Deberta: Decoding-enhanced bert with disentangled attention. Preprint, arXiv:2006.03654.

Xiaoxin He, Xavier Bresson, Thomas Laurent, Adam Perold, Yann LeCun, and Bryan Hooi. 2024a. Harnessing explanations: Llm-to-lm interpreter for enhanced text-attributed graph representation learning. Preprint, arXiv:2305.19523.

Yufei He, Yuan Sui, Xiaoxin He, and Bryan Hooi. 2024b. Unigraph: Learning a unified cross-domain foundation model for text-attributed graphs. Preprint, arXiv:2402.13630.

Xuanwen Huang, Kaiqiao Han, Yang Yang, Dezheng Bao, Quanjin Tao, Ziwei Chai, and Qi Zhu. 2024. Can gnn be good adapter for llms? Preprint, arXiv:2402.12984.

Thomas N Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In International Conference on Learning Representations, ICLR.

Xunkai Li, Meihao Liao, Zhengyu Wu, Daohan Su, Wentao Zhang, Rong-Hua Li, and Guoren Wang. 2024a. Lightdic: A simple yet effective approach for large-scale digraph representation learning. arXiv preprint arXiv:2401.11772.

Zening Li, Rong-Hua Li, Meihao Liao, Fusheng Jin, and Guoren Wang. 2024b. Privacy-preserving graph embedding based on local differential privacy. Preprint, arXiv:2310.11060.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. Preprint, arXiv:1907.11692.

Péter Mernyei and Cătălina Cangea. 2020. Wiki-cs: A wikipedia-based benchmark for graph neural networks. arXiv preprint arXiv:2007.02901.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. Preprint, arXiv:1310.4546.

Jianmo Ni, Jiacheng Li, and Julian McAuley. 2019. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In Proc. of EMNLP.

Bo Pan, Zheng Zhang, Yifei Zhang, Yuntong Hu, and Liang Zhao. 2024. Distilling large language models for text-attributed graph learning. Preprint, arXiv:2402.12022.

Emanuele Rossi, Henry Kenlay, Maria I. Gorinova, Benjamin Paul Chamberlain, Xiaowen Dong, and Michael Bronstein. 2022. On the unreasonable

effectiveness of feature propagation in learning on graphs with missing node features. Preprint, arXiv:2111.12128.

Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. 2008. Collective classification in network data. AI magazine.

Gurpreet Singh and Manoj Sachan. 2014. Multi-layer perceptron (mlp) neural network technique for offline handwritten gurmukhi character recognition. In 2014 IEEE International Conference on Computational Intelligence and Computing Research, pages 1–5.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2018. Graph attention networks. In International Conference on Learning Representations, ICLR.

Yaoke Wang, Yun Zhu, Wenqiao Zhang, Yueting Zhuang, Yunfei Li, and Siliang Tang. 2024. Bridging local details and global context in text-attributed graphs. Preprint, arXiv:2406.12608.

Zhihao Wen and Yuan Fang. 2023. Augmenting low-resource text classification with graph-grounded pre-training and prompting. In Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '23, page 506–516. ACM.

Hao Yan, Chaozhuo Li, Ruosong Long, Chao Yan, Jianan Zhao, Wenwen Zhuang, Jun Yin, Peiyan Zhang, Weihao Han, Hao Sun, and 1 others. 2023. A comprehensive study on text-attributed graphs: Benchmarking and rethinking. In Proc. of NeurIPS.

Wentao Zhang, Ziqi Yin, Zeang Sheng, Yang Li, Wen Ouyang, Xiaosen Li, Yangyu Tao, Zhi Yang, and Bin Cui. 2022. Graph attention multi-layer perceptron. Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD.

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. Preprint, arXiv:2210.14709.

Yun Zhu, Yaoke Wang, Haizhou Shi, and Siliang Tang. 2024. Efficient tuning and inference for large language models on textual graphs. Preprint, arXiv:2401.15569.

A Datasets

This section provides a detailed introduction to the datasets used in the main content. The statistics of the TAG datasets we use is as shown in Table 6. The details of each dataset are as follows:

Table 6: Statistics of the TAG datasets. The datasets are partitioned in Train-Val-Test-Out mode, 'Out' is data that not involved in the partitioning of training, validation, or test sets. All datasets are evaluated by node classification accuracy.

Dataset	#Nodes	#Edges	#Classes	#Split Ratio(%)
Cora	2,708	5,278	7	60-20-20-0
CiteSeer	3,186	4,277	6	60-20-20-0
PubMed	19,717	44,324	3	60-20-20-0
WikiCS	11,701	215,863	10	5-15-50-30
Instagram	11,339	144,010	2	10-10-80-0
Reddit	33,434	198,448	2	10-10-80-0
Elo-Photo	48,362	873,793	12	40-15-45-0

Cora (Sen et al., 2008) dataset comprises 2,708 scientific publications, which are classified into seven categories: Case-based, Genetic Algorithms, Neural Networks, Probabilistic Methods, Reinforcement Learning, Rule Learning, and Theory. Each publication in this citation network either cites or is cited by at least one other publication, forming a total of 5,278 edges. For our study, we utilize the dataset with raw texts provided by TAPE (He et al., 2024a), available at the following repository[1].

CiteSeer (Giles et al., 1998) dataset contains 3,186 scientific publications, categorized into six classes: Agents, Machine Learning, Information Retrieval, Databases, Human-Computer Interaction, and Artificial Intelligence. The objective is to predict the category of each publication using its title and abstract.

PubMed (Sen et al., 2008) dataset comprises 19,717 scientific publications from the PubMed database related to diabetes. These publications are categorized into three classes: Experimentally Induced Diabetes, Type 1 Diabetes, and Type 2 Diabetes. The associated citation network contains a total of 44,324 links.

WikiCS (Mernyei and Cangea, 2020) dataset is a Wikipedia-based resource developed for benchmarking Graph Neural Networks. It is derived from Wikipedia categories and includes 10 classes representing various branches of computer science, characterized by a high degree of connectivity. The node features are extracted from the text of the associated articles. The raw text for each node is obtained from the following repository [2].

Instagram (Huang et al., 2024) dataset serves as a social network where nodes represent users and edges correspond to following relationships. The classification task involves distinguishing between commercial and normal users within this network.

Reddit (Huang et al., 2024) dataset is a social network where nodes represent users, and node features are derived from the content of users' historically published subreddits. Edges indicate whether two users have replied to each other. The classification task involves determining whether a user belongs to the top 50% in popularity, based on the average score of all their subreddits. This dataset is built on a public resource[3], which collected replies and scores from Reddit users. The node text features are generated from each user's historical post content, limited to their last three posts. Users are categorized as popular or normal based on the median of average historical post scores, with those exceeding the median classified as popular and the rest as normal.

Ele-Photo (Yan et al., 2023) dataset is derived from the Amazon-Electronics dataset (Ni et al., 2019). In this dataset, nodes represent electronics-related products, and edges signify frequent co-purchases or co-views between products. Each node is labeled based on a three-level classification scheme for electronics products. User reviews serve as the textual attributes for the nodes; when multiple reviews are available for a product, the review with the highest number of votes is selected. If no such review exists, a random review is used. The task is to classify electronics products into 12 predefined categories.

B Baselines

This section contains detailed information about baselines:

MLP (Singh and Sachan, 2014) is a simple feed-forward neural network model, commonly used for baseline classification tasks. It consists of multiple layers of neurons, where each layer is fully connected to the previous one. The model is trained via backpropagation, with the final output layer producing predictions.

GCN (Kipf and Welling, 2017)is a graph-based

[1] Cora Dataset

[2] WikiCS Dataset

[3] Reddit Dataset

neural network model that performs node classification tasks by aggregating information from neighboring nodes. The model is built on graph convolutional layers, where each node's embedding is updated by combining the features of its neighbors, enabling it to capture the graph structure.

GAT (Veličković et al., 2018)introduces attention mechanisms to graph convolutional networks, allowing nodes to weigh their neighbors differently when aggregating features. This attention mechanism helps GAT to focus on the most informative neighbors, making it particularly effective in graphs with heterogeneous relationships between nodes.

GCNII (Chen et al., 2020)is an improved version of the GCN model, which integrates higher-order graph convolutions and a skip connection strategy. This enhancement enables GCNII to better capture deep graph structures and mitigate the over-smoothing problem that arises in deep GCN architectures.

GraphSAGE (Hamilton et al., 2017)is an inductive framework for graph representation learning, where node embeddings are learned by sampling and aggregating features from neighbors. This model can be applied to large-scale graphs by utilizing different aggregation functions, such as mean, pooling, or LSTM-based aggregation.

BERT (Devlin et al., 2019)is a pre-trained transformer-based model that learns contextualized word embeddings by predicting missing words in a sentence. BERT's bidirectional attention mechanism allows it to capture contextual information.

DeBERTa (He et al., 2021)improves upon BERT by introducing disentangled attention and enhanced decoding strategies. These innovations allow DeBERTa to better capture the relationships between different parts of the input text, leading to improved performance on multiple natural language understanding tasks.

RoBERTa (Liu et al., 2019)is an optimized version of BERT that increases training data size and model capacity, removes the Next Sentence Prediction (NSP) objective, and fine-tunes hyperparameters. These modifications lead to improved performance over BERT on many benchmark tasks, especially in natural language understanding.

GLEM (Zhao et al., 2023)is a method for learning on large TAGs. It uses a variational EM framework to alternately update LMs and GNNs, improving scalability and performance in classification.

SimTeG (Duan et al., 2023)is a straightforward yet effective approach for textual graph learning. It first conducts parameter-efficient fine - tuning (PEFT) on LM using downstream task labels. Then, it generates node embeddings from the fine-tuned LM. These embeddings are further used by a GNN for training on the same task.

TAPE (He et al., 2024a)is an approach for TAGs representation learning. It uses LLMs to generate predictions and explanations, which are then transformed into node features by fine-tuning a smaller LM. These features are used to train a GNN.

ENGINE (Zhu et al., 2024)is an efficient tuning method for integrating LLMs and GNNs in TAGs. It attaches a G-Ladder to each LLM layer to capture structural information, freezing LLM parameters to reduce training complexity. ENGINE with caching can speed up training by 12x. ENGINE (Early) uses dynamic early exit, achieving up to 5x faster inference with minimal performance loss.

G2P2 (Wen and Fang, 2023)is a model for low-resource text classification. It has two main stages. During pre-training, it jointly trains a text encoder and a graph encoder using three graph interaction-based contrastive strategies, including text-node, text-summary, and node-summary interactions, to learn a dual-modal embedding space. In downstream classification, it uses prompting. For zero-shot classification, it uses handcrafted discrete prompts, and for few-shot classification, it uses continuous prompts with graph context-based initialization.

LLMGNN (Chen et al., 2024)is a pipeline for label-free node classification on graphs. It uses LLMs to annotate nodes and GNNs for prediction. It selects nodes considering annotation difficulty, gets confidence - aware annotations, and post - filters to improve annotation quality, achieving good results at low cost.

GIANT (Chien et al., 2022)is a self-supervised learning framework for graph-guided numerical node feature extraction. It addresses the graph-agnostic feature extraction issue in standard GNN pipelines. By formulating neighborhood prediction as an XMC problem and using XR-Transformers, it fine-tunes LMs with graph information.

GraphAdapter (Huang et al., 2024)is an approach that uses GNN as an efficient adapter for LLMs to model TAGs. It conducts language-structure pre-training to jointly learn with frozen LLMs, integrating structural and textual information. After pre-training, it can be fine-tuned with prompts for downstream tasks. Experiments show it outperforms baselines on multiple datasets.

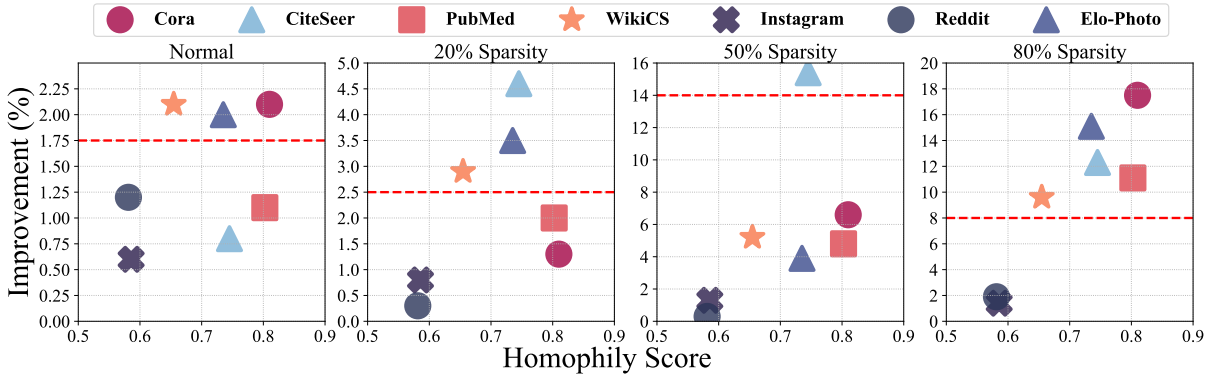

Figure 7: Homophily Analysis of the Datasets. The horizontal coordinate is the homophily score of the datasets, and the vertical coordinate is the improvement of UltraTAG-S compared to the suboptimal method.

C Hyperparameter Settings

For detailed parameter settings, we employ five random seeds {42, 43, 44, 45, 46}. The $\text{GNN}_1(\cdot)$ and $\text{GNN}_2(\cdot)$ use the Adam optimizer for joint optimizing with a learning rate of 1e-2, weight decay of 5e-4, dropout of 0.5, and 100 epochs. Each GNN consists of 2 layers, with a similarity calculation threshold of 0.8. The number of important nodes selected by PageRank is 10% of the total training nodes, and the acceptance threshold for LLM-based edge reconfiguration is 0.5. For fine-tuning the LM, the learning rate is set to 5e-5, with 3 epochs, a batch size of 8, and a dropout of 0.3. The LLM used is Meta-Llama-3-8B-Instruct (AI@Meta, 2024) in full compliance with its research license terms. All experiments were conducted on a system equipped with a single NVIDIA A100 80GB PCIe GPU and an Intel Xeon Gold 6240 CPU @ 2.60GHz (18-core), running CUDA 12.4 on Ubuntu 22.04 LTS with 256GB RAM and Python 3.12.

D Homophily Analysis

In this section, we examine the impact of dataset homogeneity and heterogeneity levels on the performance improvement of UltraTAG-S, with experimental results illustrated in Figure 7. We compute the homophily scores for all datasets and measure UltraTAG-S's performance gains over the suboptimal method under normal and varying sparsity conditions. The results demonstrate that our method consistently improves performance across all datasets under different scenarios. Notably, datasets with higher homogeneity scores exhibit greater performance gains. Furthermore, the optimal improvement is achieved when the homophily score is approximately 0.7.

E Robustness Comparison

This section contains the performance of various existing methods for node classification tasks on multiple datasets in multiple sparse scenarios. The results of 20% and 50% sparse ratio are shown in Table 7 and 8, respectively. As shown in the table, under 20% and 50% data sparsity conditions, UltraTAG-S still achieves the best node classification performance and robustness among all existing methods with up to 4.6% and 15.4% performance improvement.

Meanwhile, the LLM-GNN hybrid approach demonstrates superior robustness across varying sparsity levels, exhibiting significantly less accuracy degradation compared to baseline methods. Notably, GNN-only baselines achieve better robustness than LM-only baselines, indicating that GNNs are more effective than LMs at handling sparse graph data and possess stronger inherent robustness for graph-structured data.

F LLM Prompts

The LLM employed in our study is Meta-Llama-3-8B-Instruct, which is utilized for both text augmentation and structure augmentation tasks. This section provides a comprehensive overview of all the prompts we designed and implemented. Each prompt follows a consistent structure comprising "Dataset Description + Question," where the dataset description serves to contextualize the query and ensure clarity.

For different inference scenarios, our approach to querying the LLM is as follows. Due to the uniformity of the query format, the following demonstrations of Key Words and Soft Labels etc. use the Cora dataset as an example.

Table 7: Robustness Comparison in Sparse Scenarios with Ratio of 20%, which means nodes' texts and edges with proportion of 20% are removed randomly. Optimal performance is in **bold** and the sub-optimal is underlined.

Methods	Cora	CiteSeer	PubMed	WikiCS	Instagram	Reddit	Elo-Photo
MLP	40.70±3.10	48.37±5.08	59.59±4.25	42.27±4.84	62.18±3.75	54.68±2.05	48.51±2.31
GCN	75.50±3.22	64.86±1.31	75.57±5.08	72.33±3.94	60.77±4.89	52.61±2.05	70.85±4.15
GAT	68.67±2.06	65.64±1.01	72.93±0.92	68.10±0.68	64.90±0.35	57.43±1.24	61.05±0.85
GCNII	71.77±0.88	67.43±0.61	73.75±0.79	68.54±0.84	64.68±0.35	61.08±0.36	58.58±0.84
GraphSAGE	73.80±3.29	60.09±2.06	73.39±5.02	66.50±2.95	56.92±7.35	59.03±1.34	63.06±8.04
BERT	69.37±0.32	63.87±0.14	83.22±0.01	61.35±0.69	63.87±0.11	57.31±0.23	65.48±0.00
DeBERTa	58.21±8.94	53.53±1.96	82.78±0.35	45.93±3.86	61.65±1.07	56.20±3.48	65.34±0.45
RoBERTa	69.88±0.72	65.13±0.41	83.37±0.03	60.98±0.45	64.84±0.17	50.02±0.04	64.89±0.08
GLEM	85.71±2.01	68.71±1.54	82.36±0.37	72.59±3.01	63.37±0.18	55.82±0.21	72.75±1.94
SimTeG	82.34±0.74	70.10±0.60	85.65±0.41	71.33±0.13	62.46±0.62	60.28±0.35	75.46±0.28
TAPE	87.78±0.53	71.11±0.39	87.25±0.75	78.97±0.22	62.17±0.92	61.08±0.66	80.81±0.41
ENGINE	86.27±0.67	73.70±0.33	88.10±0.13	79.43±0.25	65.53±0.22	61.45±0.38	80.34±0.09
UltraTAG-S	**88.93±0.74**	**77.12±0.28**	**89.88±0.21**	**81.72±0.18**	**66.08±0.70**	**61.61±0.12**	**83.17±0.06**

Table 8: Robustness Comparison in Sparse Scenarios with Ratio of 50%, which means nodes' texts and edges with proportion of 50% are removed randomly. Optimal performance is in **bold** and the sub-optimal is underlined.

Methods	Cora	CiteSeer	PubMed	WikiCS	Instagram	Reddit	Elo-Photo
MLP	36.35±2.90	40.38±1.68	49.14±4.41	34.29±3.24	62.65±2.18	52.09±1.25	46.57±1.75
GCN	64.61±2.81	48.71±2.16	63.71±0.91	64.12±9.69	58.96±9.54	55.48±3.03	62.61±3.70
GAT	60.63±1.80	51.50±0.61	66.82±0.80	64.60±0.37	64.51±0.23	56.56±2.00	59.32±0.92
GCNII	62.36±0.87	49.66±0.73	67.01±0.67	63.62±0.41	64.08±0.23	59.27±1.15	53.77±1.58
GraphSAGE	54.76±3.89	47.68±0.84	64.74±5.26	64.56±1.17	62.09±3.04	60.14±0.83	62.61±2.51
BERT	55.58±0.24	48.08±0.07	66.28±0.26	45.21±0.81	63.50±0.09	54.35±0.06	57.41±0.04
DeBERTa	34.41±6.99	43.81±3.29	65.75±0.31	32.91±0.73	63.73±0.39	50.90±1.65	57.39±0.19
RoBERTa	53.09±0.40	44.32±1.02	66.28±0.04	42.73±2.54	63.07±0.18	50.97±1.77	57.25±0.00
GLEM	64.84±1.63	53.43±0.25	70.51±1.95	67.07±3.08	62.43±0.21	53.85±0.08	65.25±1.58
SimTeG	72.06±0.59	58.11±0.40	76.00±0.55	65.34±0.46	61.55±0.79	59.84±0.67	67.76±0.45
TAPE	78.73±0.34	54.31±0.78	77.18±0.46	73.62±0.63	61.40±0.26	60.18±0.19	76.21±0.69
ENGINE	70.85±0.48	56.30±0.12	75.42±0.15	71.72±0.47	64.74±0.05	60.18±0.18	73.40±0.23
UltraTAG-S	**83.95±0.80**	**67.08±0.28**	**80.91±0.29**	**77.45±0.33**	**65.61±0.12**	**60.34±0.21**	**79.21±0.06**

WikiCS:

Here is an article from the WikiCS dataset. This dataset is a Wikipedia-based resource developed for benchmarking Graph Neural Networks (GNNs). It is derived from Wikipedia categories and includes 10 classes representing various branches of computer science, characterized by a high degree of connectivity. The 10 classes are Computational Linguistics, Databases, Operating Systems, Computer Architecture, Computer Security, Internet Protocols, Computer File Systems, Distributed Computing Architectures, Web Technologies, and Programming Languages.

Edge Reconfigure:

You are provided with the text information of two nodes and their predicted category pseudo-label. Use this information to evaluate whether an edge should exist between the two nodes, and return a probability value between 0 and 1 representing the likelihood of the edge's existence. Only output the probability value, without any additional or irrelevant content. As for Node 1: <Title 1><Abstract 1>. Your prediction label is <SoftLabel 1>; As for Node 2: <Title 2><Abstract 2>. Your prediction label is <SoftLabel 2>.

Cora:

Now, here is a paper from the Cora dataset. This paper falls into one of seven categories: Case-based, Genetic Algorithms, Neural Networks, Probabilistic Methods, Reinforcement Learning, Rule Learning, and Theory.

Instagram:

This is a post from Instagram, a social network where edges represent following relationships and nodes represent users. The task is to classify users into two categories: commercial and normal.

CiteSeer:

Now, here is a paper from the Citeseer dataset. This paper falls into one of six categories: Agents, Machine Learning, Information Retrieval, Databases, Human-Computer Interaction, or Artificial Intelligence.

Reddit:

This is a post from the Reddit dataset, a social network where nodes represent users, and node features are derived from the content of users' historically published subreddits. Edges represent whether two users have replied to each other. The task is to classify users as belonging to the top 50 percent in popularity, based on the average score of all their subreddits. Node text features are generated from the content of each user's last three posts. Users are categorized as 'popular' or 'normal' based on the median of their average historical post scores, with those above the median classified as 'popular' and the rest as 'normal'.

PubMed:

The following is a paper from the PubMed dataset, which contains 19,717 scientific publications related to diabetes. These publications are categorized into three classes: Experimentally Induced Diabetes, Type 1 Diabetes, and Type 2 Diabetes.

Elo-Photo:

Here is a product review from the Elo-Potho dataset. The Elo-Potho dataset is derived from the Amazon-Electronics dataset. In this dataset, nodes represent electronics products, and edges indicate frequent co-purchases or co-views between products. Each node is labeled according to a three-level classification scheme for electronics products. User reviews serve as the textual attributes for the nodes; when multiple reviews are available for a product, the review with the highest number of votes is selected. If no such review exists, a random review is used. The task is to classify electronics products into 12 predefined categories. The categories are: Amazon Echo, Camera, Cell Phones, Clothing, Computers, Home and Kitchen, Laptops, Music, Office Supplies, Personal Care, Shoes, Sports and Outdoors.

Key Words:

Please help me identify the five keywords from its title and abstract that are most relevant for classification, and directly output the keywords. The title and abstract of the paper are as follows:<Title><Abstract>

Soft Labels:

Based on its title and abstract, please predict the most appropriate label for this paper and provide only the label as your response. The title and abstract of the paper are as follows:<Title><Abstract>

Summary:

Please summarize the title and abstract to improve their suitability for the classification task. Output only the summary text, without including any irrelevant content. The title and abstract of the paper are as follows:<Title><Abstract>

G Ablation Study and Backbones

This section presents more detailed ablation study results. We conducted ablation study under all sparse conditions on the Cora, CiteSeer, and PubMed datasets, as shown in Table 10. The results in the table demonstrate that each module of UltraTAG-S contributes to performance improvement, and the integrated combination of all modules achieves optimal experimental results.

Furthermore, we tested the impact of different language model backbones on node classification accuracy, performing comparative experiments under all sparse conditions on the Cora, CiteSeer and PubMed datasets as well, as shown in Table 9. From the table, it can be observed that under conditions of lower data sparsity, using the BERT model for text encoding yields superior classification accuracy. However, when the data sparsity reaches as high as 80%, employing the RoBERTa model for text encoding results in better classification accuracy.

Additionally, we briefly explored the impact of different text propagation strategies on the model's classification accuracy, with the experimental results presented in Table 11. We employ diverse LLM-generated texts as node text augmentation content. The table presents experimental results obtained by replacing original node texts with augmented versions under various permutations. The results demonstrate that all types of LLM-augmented texts generated using our prompts contribute to performance improvements. Moreover, optimal results can be achieved through appropriate connection combinations. From the table, it is evident that enhancing the original text data with more effective LLM-augmented texts can significantly improve the performance across all sparse conditions.

Table 9: Performance Comparison with Different Text Encoders in ideal scenarios and sparse scenarios. The optimal performance is in **bold**.

Encoder	Cora	CiteSeer	PubMed
Sparse Ratio	0%	0%	0%
BERT	**90.96±0.45**	**78.68±0.21**	91.99±1.21
DeBERTa	83.39±0.39	76.02±1.45	92.06±1.77
RoBERTa	88.38±1.82	77.74±0.76	**92.41±0.30**
Sparse Ratio	20%	20%	20%
BERT	**88.93±0.74**	**77.12±0.28**	89.38±0.29
DeBERTa	81.55±0.59	72.88±0.61	89.33±0.11
RoBERTa	88.75±0.22	74.92±0.46	**89.88±0.21**
Sparse Ratio	50%	50%	50%
BERT	**83.95±0.80**	**67.08±0.28**	80.76±0.26
DeBERTa	73.99±0.45	62.70±1.08	80.53±0.21
RoBERTa	80.81±0.54	65.05±0.14	**80.91±0.29**
Sparse Ratio	80%	80%	80%
BERT	**57.57±1.38**	**40.08±0.45**	60.70±0.74
DeBERTa	52.21±0.76	39.03±0.64	60.62±0.76
RoBERTa	54.61±1.06	38.87±0.88	**61.05±0.49**

Table 10: Detailed Performance Comparison of Ablation Study. 'TA', 'SA', 'SL' represent Text Augmentation, Structure Augmentation and Structure Learning, respectively.

Method	Cora	CiteSeer	PubMed
Sparse Ratio	0%	0%	0%
w/o TA	89.59±1.39	77.37±0.96	91.29±0.90
w/o SA	90.59±0.86	78.37±1.63	91.99±1.75
w/o SL	88.38±0.85	77.59±1.44	86.36±1.23
UltraTAG-S	**90.96±0.45**	**78.68±0.21**	**92.41±0.30**
Sparse Ratio	20%	20%	20%
w/o TA	86.52±0.21	75.12±0.45	88.38±1.50
w/o SA	87.93±1.72	77.01±0.31	89.18±0.58
w/o SL	86.72±1.33	68.34±0.90	85.42±1.34
UltraTAG-S	**88.93±0.74**	**77.12±0.28**	**89.88±0.21**
Sparse Ratio	50%	50%	50%
w/o TA	78.41±0.32	59.94±1.07	52.05±0.45
w/o SA	81.95±1.30	65.08±1.06	78.76±0.22
w/o SL	74.54±0.54	50.63±1.42	69.17±0.71
UltraTAG-S	**83.95±0.80**	**67.08±0.28**	**80.91±0.29**
Sparse Ratio	80%	80%	80%
w/o TA	48.60±1.83	34.29±1.42	44.93±1.22
w/o SA	56.83±1.45	39.50±0.54	57.65±1.21
w/o SL	38.01±1.08	35.71±0.04	43.58±0.45
UltraTAG-S	**57.57±1.38**	**40.08±0.45**	**61.05±0.49**

Table 11: Performance Comparison with Different Augmentation Texts Generated by LLM and Different Text Aggregator Strategies. 'OT' original texts, '+' means concatenate.

Texts	Cora	CiteSeer	PubMed
Sparse Ratio	0%	0%	0%
OT	89.48±0.56	75.24±1.66	90.62±0.62
OT+Su	90.22±1.23	77.59±0.24	91.89±1.19
OT+KW	90.41±0.89	77.59±1.35	91.84±0.94
OT+SL	89.48±1.78	77.74±0.31	91.99±0.29
OT+SKWSL	**90.96±0.45**	**78.68±0.21**	**92.41±0.30**
Sparse Ratio	20%	20%	20%
OT	88.12±0.34	74.92±0.71	87.78±0.74
OT+Su	88.76±0.67	77.12±1.88	89.12±1.08
OT+KW	88.43±1.12	77.27±0.42	89.48±0.95
OT+SL	88.53±0.98	75.86±1.53	89.38±0.27
OT+SKWSL	**88.93±0.74**	**77.12±0.28**	**89.88±0.21**
Sparse Ratio	50%	50%	50%
OT	82.47±0.21	64.89±1.21	79.54±0.39
OT+Su	82.66±0.76	65.52±1.65	80.78±0.61
OT+KW	83.58±0.43	65.83±0.25	80.65±0.90
OT+SL	83.21±0.65	66.46±0.41	80.78±0.28
OT+SKWSL	**83.95±0.80**	**67.08±0.28**	**80.91±0.29**
Sparse Ratio	80%	80%	80%
OT	54.98±0.91	39.34±0.96	60.40±0.77
OT+Su	57.01±1.77	39.66±0.26	60.83±1.07
OT+KW	57.43±0.32	39.97±0.73	60.78±0.60
OT+SL	57.38±0.68	39.81±0.40	60.70±0.82
OT+SKWSL	**57.57±1.38**	**40.08±0.45**	**61.05±0.49**