Bridging Local Details and Global Context in Text-Attributed Graphs

Anonymous ACL submission

Abstract

Representation learning on text-attributed graphs (TAGs) is vital for real-world applications, as they combine semantic textual and contextual structural information. Research in this field generally consist of two main perspectives: local-level encoding and global-level aggregating, respectively refer to textual node information unification (*e.g.*, using Language Models) and structure-augmented modeling (*e.g.*, using Graph Neural Networks). Most existing works focus on combining different information levels but overlook the interconnections, *i.e.*, the contextual textual information among nodes, which provides semantic insights to bridge local and global levels. In this paper, we propose GraphBridge, a *multi-granularity integration* framework that bridges local and global perspectives by leveraging contextual textual information, enhancing fine-grained understanding of TAGs. Besides, to tackle scalability and efficiency challenges, we introduce a graph-aware token reduction module. Extensive experiments across various models and datasets show that our method achieves state-of-the-art performance, while our graph-aware token reduction module significantly enhances efficiency and solves scalability issues. Codes are available at https://anonymous.4open.science/r/GraphBridge-13E0

1 Introduction

Text-**A**ttributed **G**raphs (TAGs), characterized by the association of nodes with text attributes (Yang et al., 2021), are prevalent in diverse real-world contexts. In TAGs, nodes represent entities with textual information and edges capture relationships between entities, *e.g.*, social graphs where each user is accompanied by a textual description and paper citation graphs where textual content is linked to each respective paper. These relationships yield specialized and crucial insights that are fundamental for our understanding, thereby facilitating the resolution of subsequent tasks. The utilization of TAGs empowers us to unlock new discoveries across various domains, including graph learning (Zhang et al., 2024) and information retrieval (Seo et al., 2024).

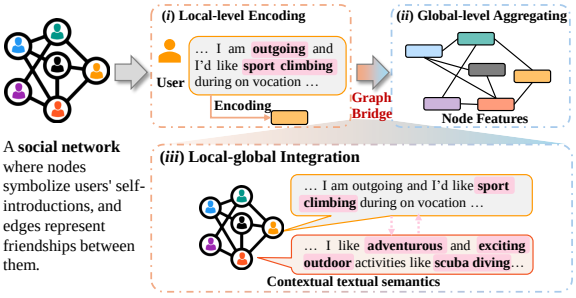

Figure 1: Illustration of the local-global integration in TAGs within a social network context. The words in pink emphasize the interconnection semantic relationship between them. (*i*) The local-level encoding module processes individual nodes' textual information into unified vectors; (*ii*) The global-level aggregating module enhances node features with structural information; (*iii*) Our method bridges these two perspectives through incorporating contextual textual information.

The nucleus of learning on TAGs lies in the effective integration of both the node attributes (textual semantics) and graph topology (structural connections) to facilitate the learning of node representations. Broadly, previous methods can be divided into two modules: (*i*) encoding module (local-level) and (*ii*) aggregating module (global-level). The encoding module transcribes the textual information (tokens) of each node into a unified vector employing static shallow embedding techniques such as Bag of Words (Zhang et al., 2010), or language models (LMs) like BERT (Devlin et al., 2018), serving as node attributes. The aggregating module enhances these features via structural information, procuring structure-augmented features through Graph Neural Networks (GNNs) like GCN (Kipf and Welling, 2016). These two mod-

ules can be integrated into cascading (Duan et al., 2023; Chien et al., 2021), joint (Zhao et al., 2022) or side structure (Zhu et al., 2024), as illustrated in Figure 1. Despite promising, the aforementioned local-global integration suffers from the discrete interconnection of encoding and aggregating modules, *i.e.*, the contextual textual semantics among nodes are overlooked (Figure 1(*iii*)). In real-life scenarios, *e.g.*, social networks, the closely connected individuals are more likely to share substantial semantically textual information in text, which serves as the common characteristics for constructing their relationships.

Based on the aforementioned insight, one optimizable TAG learning solution is to leverage such contextual textual information that could effectively bridge local and global perspectives, thereby boosting fine-grained understanding of TAGs, like Figure 1(*iii*). However, this method faces severe efficiency and scalability issues. Memory complexity increases with graph size, as neighborhood texts are also encoded. Using Large Language Models (LLMs) with densely connected nodes further exacerbates resource consumption, potentially impairing TAG's practicality. In summary, these shortcomings necessitate a thorough reevaluation of TAG learning and its corresponding solutions.

In this work, we introduce a novel *multi-granularity integration* framework for text-attributed graphs, named GraphBridge, which seamlessly bridges local and global perspectives by incorporating contextual textual information. This method enhances semantic analysis and provides deeper graph structure insights, significantly improving representation learning. Additionally, to address the efficiency and scalability issues mentioned above, we developed a graph-aware token reduction module. This module uses a learnable mechanism that considers both the graph structure and downstream task information to selectively retain the most crucial tokens, reducing information loss and allowing for the inclusion of more contextual text. Extensive experiments show that our method achieves state-of-the-art performance across various domains compared to previous methods, while solving the efficiency and scalability issues. Key contributions of this work include:

- We propose an innovative *multi-granularity integration* framework named GraphBridge to integrate both local and global perspectives through leveraging contextual textual information, thereby enhancing the fine-grained understanding of TAGs.

- A graph-aware token reduction module is designed to ensure efficiency and scalability while minimizing information loss.

- Extensive experiments conducted across various domains demonstrate that our proposed method achieves state-of-the-art performance compared to various baselines, demonstrating its effectiveness in bridging the gap between local and global information, while maintaining efficiency and scalability.

2 Related Work

2.1 Representation Learning on TAGs

Representation learning for text-attributed graphs has increasingly garnered attention in graph machine learning (Yang et al., 2021). Typically, previous methods in this field can be divided into two key components, as depicted in Figure 1: (*i*) an encoding module at the local level, which employs word embedding methods such as Bag of Words (Zhang et al., 2010) or advanced LMs like BERT (Devlin et al., 2018) and RoBERTa (Liu et al., 2019) to generate token representations from nodes' textual data. These representations are integrated using methods like mean pooling to derive nodes' attributes; (*ii*) an aggregating module at the global level, which utilizes GNNs (Kipf and Welling, 2016; Veličković et al., 2018) or graph transformers (Wu et al., 2022) to augment nodes' attributes with structural information. The encoding module concentrates on extracting fine-grained semantic details from textual attributes individually, whereas the aggregating module emphasizes structural relationships between nodes, neglecting the intricate local textual information.

Recent advancements aim to effectively integrate these two modules. Integration strategies include joint frameworks (Yang et al., 2021; Zhao et al., 2022) and side structures (Zhu et al., 2024), as well as cascading approaches (Duan et al., 2023; He et al., 2023). However, these integration strategies fail to explicitly capture the interconnection between the encoding and aggregating modules, *i.e.*, the contextual textual semantics among nodes are frequently overlooked, potentially compromising the efficacy of the results.

2.2 Token Reduction for LMs

Sequence length has become a significant factor limiting the scalability of transformer models (Vaswani et al., 2017). Token reduction has gained considerable research interest because it can reduce computational costs by decreasing sequence length. The general idea of token reduction is to drop some tokens based on their importance (Xu and McAuley, 2023). Specifically, DynSAN (Zhuang and Wang, 2019) applies a gate mechanism to measure the importance of tokens for selection, dropping less important tokens in higher layers to enhance efficiency. TR-BERT (Ye et al., 2021) introduces a dynamic reinforcement learning mechanism for making decisions of reducing tokens. LTP (Kim et al., 2022) learns a threshold for each Transformer layer, dropping tokens with a saliency score below this threshold instead of adhering to a preset token reduction schedule.

Although token reduction has proven successful in streamlining individual sentences, its application to interrelated texts within TAGs has yet to be fully explored. In this work, we propose a graph-aware token reduction module that leverages the graph structure along with the information from downstream tasks to perform token reduction.

3 Method

In this section, we will introduce the notations used in this paper. Subsequently, we will present the proposed graph-aware token reduction method in Section 3.2. Finally, the multi-granularity integration framework will be discussed in Section 3.3.

3.1 Notations

When dealing with node classification tasks of TAGs, we formally consider a text-attributed graph $\mathcal{G} = \{\mathcal{V}, \mathcal{T}, \mathcal{A}, \mathcal{Y}\}$, where $\mathcal{V}$ is a set of nodes, $\mathcal{T} \in \mathbb{R}^{|\mathcal{V}|\times k}$ denotes the textual features associated with each node $i \in \mathcal{V}$, and k is the sequence length. $\mathcal{A} \in \{0,1\}^{|\mathcal{V}|\times|\mathcal{V}|}$ is the adjacency matrix where each entry $\mathcal{A}_{i,j}$ indicates the link between nodes $i, j \in \mathcal{V}$, and $\mathcal{Y}$ represents labels for each node. Given a set of labeled nodes $\mathcal{V}_L \subset \mathcal{V}$, our goal is to predict the remaining unlabeled nodes $\mathcal{V}_U = \mathcal{V} \setminus \mathcal{V}_L$.

3.2 Graph-Aware Token Reduction

To bridge local and global perspectives for TAGs, it is essential to consider both the text of the current node and its neighboring nodes. This approach, however, leads to extremely long sequences that can result in prohibitive computational costs for LM processing. To mitigate this, we first implement a graph-aware token reduction module. Formally, for an input graph $\mathcal{G}$, the text of each node $\mathcal{T}_i$ is initially tokenized, resulting in $s_i \in \mathbb{R}^k$, where k is the number of tokens. Our objective is to reduce the number of tokens to k' (where $k' \ll k$), focusing on retaining the most pivotal tokens while omitting the lesser ones. Specifically, we assess the importance of each token for node i based on its textual and structural information:

$$P(\text{Score}_i \mid \mathcal{T}_i, \mathcal{T}_{\mathcal{N}_i}, \mathcal{A}), \tag{1}$$

where $\text{Score}_i \in \mathbb{R}^{1\times k}$ is the importance score for each token in node i, and $\mathcal{N}_i$ denotes the neighboring nodes of i. The importance score is calculated by a trainable graph-enhanced attention module, as depicted in Figure 2.

For better evaluating the importance of each token, we first use Pre-trained LMs like BERT and RoBERTa to obtain fine-grained representations for each token. For node i, these representations are represented as $E_i \in \mathbb{R}^{k\times d}$, consisting of vectors $[e_0, e_1, \ldots, e_k]$, where each e_j is a d-dimensional token representation from the PLM. We subsequently employ mean pooling $\mathcal{P}_{\text{mean}}$ on these token representations to extract the sentence-level textual feature, which serves as the node attribute:

$$z_i = \mathcal{P}_{\text{mean}}(E_i) = \frac{1}{k}\sum_{j}^{k} e_j. \tag{2}$$

Then, a parameter-free message-passing mechanism is employed to aggregate text features from neighboring nodes, excluding self-loops to avoid reinforcing a node's own information. This ensures the integration of contextual information from neighbors, enhancing node features with structural insights from graph. This process is described as:

$$z_i^{(l)} = \frac{1}{|\mathcal{N}_i|}\sum_{j\in\mathcal{N}_i} z_j^{(l-1)}, \tag{3}$$

where l means l-hop message passing, and z_i^0 is z_i.

Graph-Enhanced Importance Score. To measure the importance of each token, we define a graph-enhanced importance score calculated through a well designed cross-attention module,

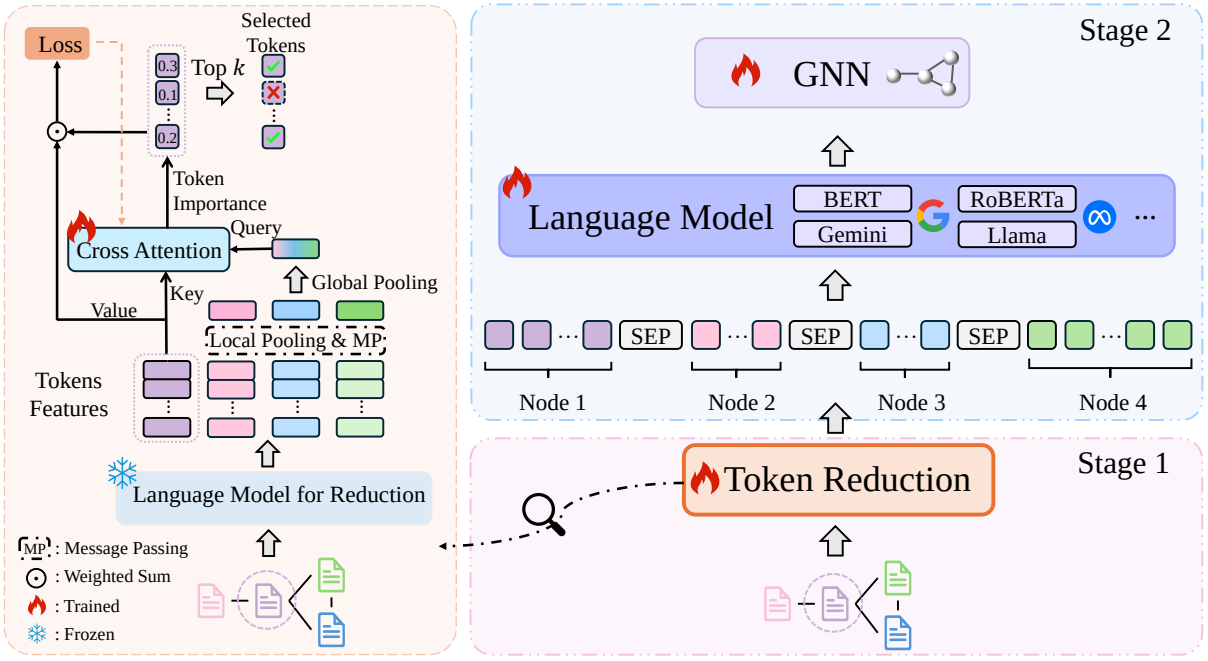

Figure 2: Overview of the **GraphBridge** framework. **Left**: The Graph-Aware Token Reduction module which selectively retains crucial tokens, enhancing efficiency and scalability. **Right**: A detailed pipeline illustrates the integration process, where selected tokens undergo a cascaded structure that bridges local and global perspectives, leveraging contextual textual information to effectively refine node representations.

that quantifies the significance of each token utilizing both textual and structural information. Specifically, for each node i, the query is derived from the message passing output of neighboring nodes, while the key and value come from the node's own textual token embedding. The importance score is calculated as follows:

$$\text{Score}_i = \sigma\left(\frac{\left(z_i^{(l)} W_q\right)\left(E_i W_k\right)^T}{\sqrt{d}}\right), \quad (4)$$

where $W_q, W_k \in \mathbb{R}^{d \times d'}$ are the parameter matrices for the query and key, and σ denotes the softmax function. A top-k function is then used to select the k' most crucial tokens.

Optimizing Importance Score. The supervisory signals derived from downstream tasks provide valuable guidance, allowing the attention module to select informative tokens more effectively. Specifically, during the training phase, we aggregate the token representations using Score_i, which can be formulated as:

$$s_i = \text{Score}_i E_i, \quad (5)$$

where $s_i \in \mathbb{R}^{1 \times d}$ denotes the weighted summation of text features using attention score. Observe that E_i can be directly regarded as the value matrix, because we set the value parameter matrix as the identity matrix I here for efficiency. Finally, s_i is fed into a linear classifier $\mathcal{C}$ for prediction. The training loss $\mathcal{L}_{\text{down}}$ is computed using the cross-entropy loss $\text{CE}(\cdot,\cdot)$ between the prediction and true label for the target node i:

$$\mathcal{L}_{\text{down}} = \mathbb{E}_{i \in \mathcal{V}_L} \text{CE}(\hat{y}_i | \mathcal{C}_i(s_i), y_i). \quad (6)$$

Note that, only the attention module and the classifier are trained while keeping the PLM frozen for efficiency.

Regularization. Through our empirical study, we discovered that the importance score for the majority of nodes deteriorates when solely optimizing $\mathcal{L}_{\text{down}}$, as depicted in Figure 3. We hypothesize that this phenomenon is due to overfitting on the limited set of training nodes. Consequently, most nodes tend to converge on a single token with an excessively high importance score (*e.g.*, 0.99), thereby hindering the selection of multiple informative tokens and inhibiting exploration. To mitigate this phenomenon, we introduce a regularization term. This term penalizes the network when certain tokens receive disproportionately high importance scores, achieved with a KL-divergence loss:

$$\mathcal{L}_{\text{reg}} = \mathbb{E}_{i \in \mathcal{V}_L} D_{\text{KL}}(U || \text{Score}_i), \quad (7)$$

where U is an uniform distribution.

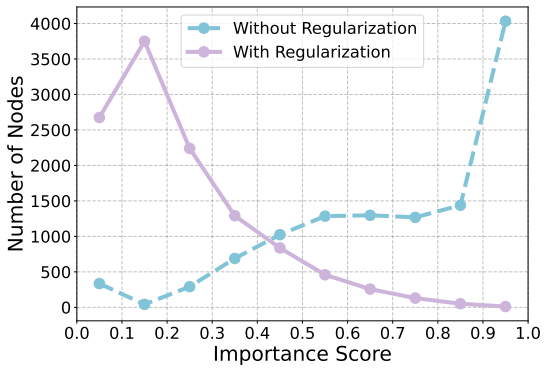

Figure 3: Selecting the highest score token for each node in WikiCS dataset, with and without regularization. The x-axis means the highest importance score of token for each node, the y-axis indicates the number of nodes corresponding to each importance score.

The overall training loss of the reduction model is as follows:

$$\mathcal{L}_{\text{train}} = \mathcal{L}_{\text{down}} + \beta \mathcal{L}_{\text{reg}}, \tag{8}$$

where β is the regularization parameter for controlling the distribution of importance score. A larger β results in a more uniform score distribution, and vice visa.

3.3 Multi-Granularity Integration

As discussed in Section 1, the integration of local-global perspectives is essential for representation learning in TAGs (Yan et al., 2023). However, previous methods (Duan et al., 2023; He et al., 2023) often fail to adequately address the interconnection between these two perspectives. In this work, we propose an innovative multi-granularity integration framework that bridges local and global perspectives by considering contextual textual semantics among nodes, as illustrated in Figure 2. Intuitively, the contextual textual semantics among nodes can offer supplementary information for a given node. For example, within a citation network, the textual content of neighboring nodes might include key terms or concepts pertinent to the target node.

To incorporate contextual information, we consider both the text of each node and its neighbors. For each node i, we concatenate its own text with the text of its neighboring nodes into a single sequence Q_i:

$$Q_i = (t_i^1, \cdots, t_i^{k'_i}, [\text{SEP}], t_{j_1}^1, \cdots, t_{j_n}^{k'_n}), \tag{9}$$

where $t_i^{k'_i}$ represents the tokens in the node i, [SEP] is a separator token for separating different nodes, and $\{j_1, \cdots, j_n\} \in \mathcal{N}_i$. Note that concatenating the text of multiple nodes results in an excessively long sequence, which results in efficiency and scalability issues. Therefore, we utilize the token reduction module described in Section 3.2 to select the most crucial k' tokens for forming the sequence.

After constructing the sequence Q_i for each target node i, we train the language model $\text{LM}(\cdot)$, which serves as the encoding module, on this sequence to obtain embeddings enriched with both textual and contextual information:

$$\mathcal{L}_{\text{LM}} = \mathbb{E}_{i \in \mathcal{V}_L} \text{CE}(\hat{y}_i | \mathcal{C}\left(\text{LM}\left(Q_i\right)\right), y_i). \tag{10}$$

It is noteworthy that sampling fewer neighboring nodes focuses the model more on the fine-grained semantic information from a local perspective, whereas sampling more neighboring nodes shifts the model's emphasis towards capturing the structural semantic information from a global perspective.

After completing the training of the LM, the model is used to produce node representations H. Subsequently, we train the aggregating module $\text{GNN}(\cdot)$ as follows:

$$\mathcal{L}_{\text{GNN}} = \mathbb{E}_{i \in \mathcal{V}_L} \text{CE}(\hat{y}_i | \mathcal{C}(\text{GNN}(A, H)_i), y_i), \tag{11}$$

where the aggregating module GNN will generate node features that further reflect the structural semantics from a global perspective.

Additionally, training the GNN and LM is fully decoupled, allowing for the use of any existing GNN and LM models. This cascading structure enables the independent optimization of each model, enhancing flexibility and facilitating integration with diverse architectures and applications.

4 Experiments

In this section, we first introduce the used datasets in Section 4.1. We then detail the baseline methods and experimental setup in Sections 4.2 and 4.3, respectively. Experiments are presented to evaluate our proposed method in Section 4.4. We further investigate the use of a causal large language model as the backbone in Section 4.5. Finally, we provide an analysis of hyper-parameters, assess scalability and efficiency, and conduct an ablation study.

4.1 Datasets

In this work, we adopt seven widely used textual graphs to evaluate our proposed GraphBridge: Cora (Sen et al., 2008), WikiCS (Mernyei and

Dataset	#Nodes	#Edges	#Avg.tokens	#Avg.degrees	#Classes
Cora	2,708	5,429	194	3.90	7
WikiCS	11,701	215,863	545	36.70	10
CiteSeer	3,186	4,277	196	1.34	6
ArXiv-2023	46,198	78,543	253	1.70	40
Ele-Photo	48,362	500,928	185	18.07	12
OGBN-Products (subset)	54,025	74,420	163	2.68	47
OGBN-ArXiv	169,343	1,166,243	231	13.67	40

Table 1: **Data statistics. #Nodes**, **#Edges**, **#Classes** and **#Avg.degrees** mean the number of nodes, edges, classes and average degrees for each dataset, respectively. **#Avg.tokens** represents the average number of tokens per node in each dataset when using the RoBERTa-base's tokenizer.

Cangea, 2020), CiteSeer (Giles et al., 1998), ArXiv-2023 (He et al., 2023), Ele-Photo (Yan et al., 2023), OGBN-Products (Hu et al., 2020) and OGBN-ArXiv (Hu et al., 2020). The raw text of these datasets are collected by previous works (Chen et al., 2023; Yan et al., 2023; He et al., 2023). Details of these datasets can be found in Appendix A.

4.2 Baselines

To verify the effectiveness of our proposed method, we select several baseline models for comparison, categorized into three types:

Traditional GNN-based methods: primarily focus on the global level but utilize static shallow embeddings, which neglect fine-grained textual information, *e.g.*, MLP, GCN (Kipf and Welling, 2016), SAGE (Hamilton et al., 2017), GAT (Veličković et al., 2018), NodeFormer (Wu et al., 2022).

LM-based methods: primarily focus on the local textual level and do not consider global structural information, *e.g.*, BERT (Devlin et al., 2018), RoBERTa (Liu et al., 2019).

Recent works designed for TAGs: integrate both local and global levels, *e.g.*, GLEM (Zhao et al., 2022), TAPE (He et al., 2023), SimTeG (Duan et al., 2023), ENGINE (Zhu et al., 2024).

4.3 Experimental Setup

For traditional GNN-based methods, we utilize the raw features of each dataset, which are derived using bag of words or one-hot vectors. For LM-based methods, we fine-tune LMs with raw texts of each node on downstream tasks. For recent TAGs methods, we select RoBERTa-base and RoBERTa-large as the LM backbones, and a two-layer SAGE with 64 hidden size as the GNN backbone. Regarding to our method, we select the same LM and GNN backbones with recent TAGs methods for a fair comparison. Additionally, we utilize RoBERTa-base as the LM encoder for our token reduction module. For alternative LMs used in token reduction, please refer to Appendix D.

In our experiments, we fine-tune all parameters for base language models such as RoBERTa-base. For larger models like RoBERTa-large, we employ LoRA (Hu et al., 2021) with a rank of 8 to ensure scalability and maintain consistency with the SimTeG approach (Duan et al., 2023).

4.4 Main Results

From Table 2, we draw the following conclusions:

First, traditional GNN-based methods, which rely on global structural information using static shallow embeddings, underperform compared to current TAGs methods like GLEM that integrate both local and global levels information. For instance, on the ArXiv-2023 dataset, this integration results in a 12% higher absolute performance over GNN methods such as GCN, highlighting the crucial role of local textual information in enhancing model efficacy.

Second, LM-based methods primarily focusing on local textual information fall short on TAGs, as evidenced by integration methods which utilize the same LM backbones surpassing them by about 10% absolute performance on the Ele-Photo dataset, achieving over 80% accuracy. This highlights the essential role of global structural information in creating more semantically and structurally aware node embeddings.

Last, our method surpasses existing local and global integration approaches for TAGs. Specifically, GraphBridge achieves an absolute improvement of over 6% on the CiteSeer dataset and 4% on the ArXiv-2023 dataset, outperforming the previous state-of-the-art method, SimTeG, across various LM backbones. This demonstrates the effectiveness of our approach in seamlessly integrating local and global perspectives by incorporating contextual textual information among nodes, thereby

Methods	Cora	WikiCS	CiteSeer	ArXiv-2023	Ele-Photo	OGBN-Products	OGBN-ArXiv
MLP	76.12 ± 1.51	68.11 ± 0.76	70.28 ± 1.13	65.41 ± 0.16	62.21 ± 0.17	58.11 ± 0.23	62.57 ± 0.11
GCN	88.12 ± 1.13	76.82 ± 0.62	71.98 ± 1.32	66.99 ± 0.19	80.11 ± 0.09	69.84 ± 0.52	70.78 ± 0.10
SAGE	87.60 ± 1.40	76.65 ± 0.84	72.44 ± 1.11	68.76 ± 0.51	79.79 ± 0.23	70.64 ± 0.20	71.72 ± 0.21
GAT	85.13 ± 0.95	77.04 ± 0.55	72.73 ± 1.18	67.61 ± 0.24	80.38 ± 0.37	69.70 ± 0.25	70.85 ± 0.17
NodeFormer	88.48 ± 0.33	75.47 ± 0.46	75.74 ± 0.54	67.44 ± 0.42	77.30 ± 0.06	67.26 ± 0.71	69.60 ± 0.08
BERT	79.70 ± 1.70	78.13 ± 0.63	71.92 ± 1.07	77.15 ± 0.09	68.79 ± 0.11	76.23 ± 0.19	72.75 ± 0.09
RoBERTa-base	78.49 ± 1.36	76.91 ± 0.69	71.66 ± 1.18	77.33 ± 0.16	69.12 ± 0.15	76.01 ± 0.14	72.51 ± 0.03
RoBERTa-large	79.79 ± 1.31	77.79 ± 0.89	72.26 ± 1.80	77.70 ± 0.35	71.22 ± 0.09	76.29 ± 0.27	73.20 ± 0.13
GLEM$_{(base)}$	87.61 ± 0.19	78.11 ± 0.61	77.51 ± 0.63	79.18 ± 0.21	81.47 ± 0.52	76.15 ± 0.32	74.46 ± 0.27
TAPE$_{(base)}$	87.82 ± 0.91	–	–	80.11 ± 0.20	–	79.46 ± 0.11	74.66 ± 0.07
SimTeG$_{(base)}$	86.85 ± 1.81	79.77 ± 0.68	78.69 ± 1.12	79.31 ± 0.49	81.61 ± 0.18	76.46 ± 0.55	74.31 ± 0.14
ENGINE$_{(base)}$	87.56 ± 1.48	77.97 ± 0.94	76.79 ± 1.38	78.34 ± 0.15	80.50 ± 0.33	77.80 ± 1.20	73.59 ± 0.14
Ours$_{(base)}$	**92.14 ± 1.03**	**80.59 ± 0.47**	**85.32 ± 1.39**	**84.07 ± 0.34**	**83.84 ± 0.07**	**79.80 ± 0.19**	74.89 ± 0.23
GLEM$_{(large)}$	**89.11 ± 0.22**	77.99 ± 0.72	78.24 ± 0.31	78.91 ± 0.40	82.11 ± 0.66	78.59 ± 0.27	74.98 ± 0.45
TAPE$_{(large)}$	88.56 ± 0.88	–	–	80.21 ± 0.31	–	**79.76 ± 0.23**	**75.29 ± 0.11**
SimTeG$_{(large)}$	88.78 ± 1.05	80.13 ± 0.76	**79.59 ± 1.56**	**80.51 ± 0.33**	82.49 ± 0.17	78.55 ± 0.66	**75.16 ± 0.21**
ENGINE$_{(large)}$	88.49 ± 1.10	**80.21 ± 0.29**	78.02 ± 0.87	77.45 ± 0.46	**82.68 ± 0.09**	78.83 ± 0.80	74.62 ± 0.30
Ours$_{(large)}$	**92.73 ± 1.00**	**80.73 ± 0.41**	**86.81 ± 1.09**	**84.79 ± 0.29**	**84.18 ± 0.15**	**80.22 ± 0.47**	**75.90 ± 0.11**

Table 2: **Experimental results of node classification**: We report the mean accuracy with a standard deviation of 5 runs with different random seeds. Highlighted are the top **first**, **second**, and **third** results. 'base' and 'large' refer to RoBERTa-base and RoBERTa-large as LM backbones, respectively. '–' indicates that datasets do not support for this method.

Methods	Cora	WikiCS	Ele-Photo
LLaMA2	82.80 ± 1.37	80.82 ± 0.48	72.06 ± 0.10
SimTeG$_{(\text{LLaMA2})}$	92.84 ± 0.13	82.55 ± 0.51	82.05 ± 0.17
ENGINE$_{(\text{LLaMA2})}$	91.48 ± 0.32	81.56 ± 0.97	83.75 ± 0.08
Ours$_{(\text{LLaMA2})}$	**93.65 ± 0.44**	**84.18 ± 0.68**	**84.35 ± 0.12**

Table 3: Experimental results when utilizing LLaMA2-7B as the Large Language Model backbone. We employ LoRA with a rank of 4 to fine-tune the LLM and report the corresponding accuracy. We use **boldface** to denote the best performace.

enhancing the fine-grained understanding of TAGs.

4.5 Enhanced with Large Language Models

Our method, as demonstrated in Section 4.4, proves effective with small and medium discriminative LMs like RoBERTa-base and RoBERTa-large. Furthermore, we have expanded our method to incorporate causal Large Language Models (LLMs), which have shown significant capabilities across various natural language tasks (Achiam et al., 2023). Table 3 presents the results obtained using LLaMA2-7B (Touvron et al., 2023) as the LLM backbone. Our method outperforms both LM-based method (*i.e.*, LLaMA2) and integration methods (*i.e.*, SimTeG, ENGINE) that utilize LLM. This demonstrates the effectiveness of our method with the LLM backbone, highlighting the importance of bridging the local and global perspectives.

4.6 Sensitive Analysis

The number of walk steps. In the construction of the sequence Q as outlined in Equation 9, sampling neighboring nodes via a random walk with restart (Zhu et al., 2022) is essential for effectively incorporating contextual textual information. The number of walk steps, a key hyper-parameter, dictates the extent of neighboring node inclusion and thus influences the breadth of contextual information captured.

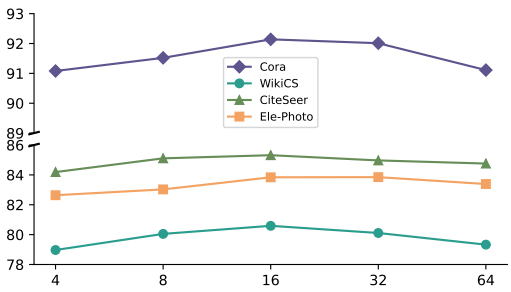

Figure 4: Sensitive analysis of the number of walk steps.

Empirically, we explore the impact of this number, choosing from {4, 8, 16, 32, 64}. The results in Figure 4 indicate that a low number of walk steps like 4, leads to insufficient contextual information, while a high number like 64, may blur fine-grained local information and introduce noise, negatively impacting performance. To balance local

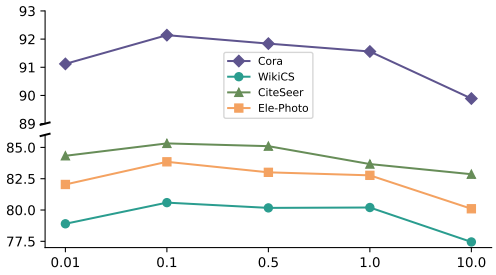

Figure 5: Sensitive analysis of regularization term β.

and global information effectively, an intermediate number of steps, *e.g.*, 16 or 32, is optimal.

The regularization term β. The regularization process penalizes the token reduction module when certain tokens receive extremely high importance scores. We analyze the regularization term β as described in Equation 8, testing various value {0.01, 0.1, 0.5, 1.0, 10.0}. Based on Figure 5, optimal results are observed with a β value of 0.1. A small or large β value can lead to a steep or excessively smooth score distribution, respectively.

4.7 Scalability and Efficiency Analysis

Walk Steps	Reduction	Memory (GB)	Total Time
8	✗	9.6	39h 14m
	✓	3.2	13h 55m
16	✗	20.1	68h 42m
	✓	4.6	18h 41m
32	✗	OOM	–
	✓	7.7	31h 14m
64	✗	OOM	–
	✓	15.2	61h 51m

Table 4: The scalability and efficiency analysis of training on the OGBN-ArXiv dataset with and without token reduction. The batch size was set to 1 for LM tuning, with total training time reported using a 48-core Intel(R) Xeon(R) CPU @ 2.50GHz and 8 NVIDIA GeForce RTX 3090 GPUs. 'OOM' refers to out of memory.

In this section, we assess the scalability and efficiency of our method by reducing the sequence length through token reduction module, detailed in Section 3.2. RoBERTa-base serves as our LM backbone, and for this experiment, we use the rotatory position embeddings (Su et al., 2024) to accommodate sequences of unlimited length. Results presented in Table 4, demonstrate that without token reduction, method which considers neighboring textual information suffers from significant computational costs in terms of training time and GPU memory usage, and it may even run out of memory with a large number of walk steps (*i.e.*, greater than 32). With our token reduction module, we only retain the most k' crucial tokens for each nodes, thereby enhancing efficiency and scalability.

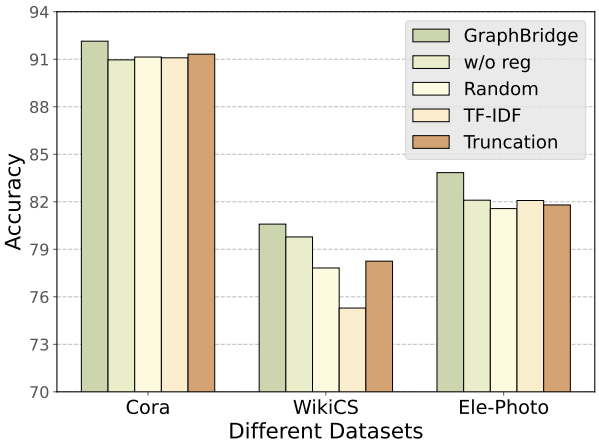

Figure 6: Experimental results of ablation study.

4.8 Ablation Study

In this section, we evaluate the effectiveness of our token reduction module. Specifically, 'w/o reg' refers to training the module without regularization. The methods 'Random', 'TF-IDF', and 'Truncation' represent three different alternative token reduction strategies: random selection, selection based on TF-IDF scores, and selecting the initial tokens from texts, respectively. Figure 6 shows that our token reduction module outperforms other reduction methods, highlighting its effectiveness in selecting crucial tokens. Additionally, regularization is essential as it helps prevent overfitting to specific tokens.

5 Conclusion

In this paper, we introduce GraphBridge, an innovative multi-granularity integration framework for text-attributed graphs. Our method emphasizes the importance of bridging local and global perspectives by incorporating contextual textual information, thereby enhancing the fine-grained understanding of TAGs. To tackle scalability and efficiency challenges associated with handling extensive textual data, we propose a graph-aware token reduction module. Empirical studies confirm that GraphBridge surpasses existing state-of-the-art methods on various datasets.

6 Limitations

This work introduces a framework that seamlessly integrates both local and global perspectives by leveraging contextual textual information for TAGs. However, it primarily focuses on high-level discriminative tasks such as node classification and cannot be directly applied to generative tasks like graph description. Leveraging this framework to construct a graph foundation model presents a challenging yet valuable area for exploration.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Zhikai Chen, Haitao Mao, Hongzhi Wen, Haoyu Han, Wei Jin, Haiyang Zhang, Hui Liu, and Jiliang Tang. 2023. Label-free node classification on graphs with large language models (llms). *arXiv preprint arXiv:2310.04668*.

Eli Chien, Wei-Cheng Chang, Cho-Jui Hsieh, Hsiang-Fu Yu, Jiong Zhang, Olgica Milenkovic, and Inderjit S Dhillon. 2021. Node feature extraction by self-supervised multi-scale neighborhood prediction. In *Proc. of ICLR*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv 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. *arXiv 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*, pages 89–98.

Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30.

Xiaoxin He, Xavier Bresson, Thomas Laurent, Adam Perold, Yann LeCun, and Bryan Hooi. 2023. Harnessing explanations: Llm-to-lm interpreter for enhanced text-attributed graph representation learning. In *The Twelfth International Conference on Learning Representations*.

Edward J Hu, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. 2021. Lora: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. 2020. Open graph benchmark: Datasets for machine learning on graphs. *Advances in neural information processing systems*, 33:22118–22133.

Sehoon Kim, Sheng Shen, David Thorsley, Amir Gholami, Woosuk Kwon, Joseph Hassoun, and Kurt Keutzer. 2022. Learned token pruning for transformers. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 784–794.

Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*.

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. *arXiv 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*.

Jianmo Ni, Jiacheng Li, and Julian McAuley. 2019. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*, pages 188–197.

Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. 2008. Collective classification in network data. *AI magazine*, 29(3):93–93.

Hyunjin Seo, Taewon Kim, June Yong Yang, and Eunho Yang. 2024. Unleashing the potential of text-attributed graphs: Automatic relation decomposition via large language models. *arXiv preprint arXiv:2405.18581*.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. 2024. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

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

Kuansan Wang, Zhihong Shen, Chiyuan Huang, Chieh-Han Wu, Yuxiao Dong, and Anshul Kanakia. 2020. Microsoft academic graph: When experts are not enough. *Quantitative Science Studies*, 1(1):396–413.

Qitian Wu, Wentao Zhao, Zenan Li, David P Wipf, and Junchi Yan. 2022. Nodeformer: A scalable graph structure learning transformer for node classification. *Advances in Neural Information Processing Systems*, 35:27387–27401.

Canwen Xu and Julian McAuley. 2023. A survey on model compression and acceleration for pretrained language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 10566–10575.

Hao Yan, Chaozhuo Li, Ruosong Long, Chao Yan, Jianan Zhao, Wenwen Zhuang, Jun Yin, Peiyan Zhang, Weihao Han, Hao Sun, et al. 2023. A comprehensive study on text-attributed graphs: Benchmarking and rethinking. *Advances in Neural Information Processing Systems*, 36:17238–17264.

Junhan Yang, Zheng Liu, Shitao Xiao, Chaozhuo Li, Defu Lian, Sanjay Agrawal, Amit Singh, Guangzhong Sun, and Xing Xie. 2021. Graphformers: Gnn-nested transformers for representation learning on textual graph. *Advances in Neural Information Processing Systems*, 34:28798–28810.

Deming Ye, Yankai Lin, Yufei Huang, and Maosong Sun. 2021. Tr-bert: Dynamic token reduction for accelerating bert inference. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5798–5809.

Delvin Ce Zhang, Menglin Yang, Rex Ying, and Hady W Lauw. 2024. Text-attributed graph representation learning: Methods, applications, and challenges. In *Companion Proceedings of the ACM on Web Conference 2024*, pages 1298–1301.

Yin Zhang, Rong Jin, and Zhi-Hua Zhou. 2010. Understanding bag-of-words model: a statistical framework. *International journal of machine learning and cybernetics*.

Jianan Zhao, Meng Qu, Chaozhuo Li, Hao Yan, Qian Liu, Rui Li, Xing Xie, and Jian Tang. 2022. Learning on large-scale text-attributed graphs via variational inference. In *The Eleventh International Conference on Learning Representations*.

Yun Zhu, Jianhao Guo, Fei Wu, and Siliang Tang. 2022. Rosa: A robust self-aligned framework for node-node graph contrastive learning. *arXiv preprint arXiv:2204.13846*.

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

Yimeng Zhuang and Huadong Wang. 2019. Token-level dynamic self-attention network for multi-passage reading comprehension. In *Proceedings of the 57th annual meeting of the association for computational linguistics*, pages 2252–2262.

A Datasets

We evaluated our method using seven widely recognized text-attributed graph datasets. The details of these datasets are as follows:

Cora (Sen et al., 2008) dataset contains 2,708 scientific publications classified into seven classes: case-based, genetic algorithms, neural networks, probabilistic methods, reinforcement learning, rule learning, and theory. Each paper in the citation network cites or is cited by at least one other paper, resulting in a total of 5,429 edges.

WikiCS (Mernyei and Cangea, 2020) dataset is a Wikipedia-based dataset designed for benchmarking Graph Neural Networks, consisting of 10 computer science branches as classes with high connectivity. Node features are derived from the corresponding article texts[1].

CiteSeer (Giles et al., 1998) dataset comprises 3,186 scientific publications categorized into six areas: Agents, Machine Learning, Information Retrieval, Database, Human Computer Interaction, and Artificial Intelligence, with the task of classifying each paper based on its title and abstract.

ArXiv-2023 dataset, introduced in TAPE (He et al., 2023), is a directed graph representing the citation network of computer science arXiv papers published in 2023 or later. Similar to OGBN-ArXiv, it features nodes representing arXiv papers and directed edges for citations. The goal is to classify each paper into one of 40 subject areas such as cs.AI, cs.LG, and cs.OS, with classifications provided by the authors and arXiv moderators.

Ele-Photo (Yan et al., 2023) dataset, derived from the AmazonElectronics dataset (Ni et al., 2019), consists of nodes representing electronics products, with edges indicating frequent co-purchases or co-views. Each node is labeled according to a three-level classification of electronics products. The text attribute for each node is the user review with the most votes, or a randomly selected review if no highly-voted reviews are available. The task is to classify these products into 12 categories.

OGBN-Products (Hu et al., 2020) dataset, comprising 2 million nodes and 61 million edges, is reduced using a node sampling strategy from TAPE (He et al., 2023) to create the OGBN-Products (subset) with 54k nodes and 74k edges. Each node represents an Amazon product, with edges denoting co-purchases. The classification task involves categorizing products into one of 47 top-level categories.

OGBN-ArXiv dataset is a directed graph depicting the citation network among computer science arXiv papers indexed by MAG (Wang et al., 2020). Each node represents an arXiv paper with directed edges indicating citations. The goal is to classify papers into one of 40 subject areas like cs.AI, cs.LG, and cs.OS, with labels manually assigned by the authors and arXiv moderators.

B Baselines

The details of the baseline methods we compared GraphBridge to are as follows:

- Traditional GNNs: In this work, we adopted three simple yet widely used GNN models: GCN (Kipf and Welling, 2016), SAGE (Hamilton et al., 2017), and GAT (Veličković et al., 2018). Additionally, We include a graph transformer as the GNNs-based methods baseline, *i.e.*, NodeFormer (Wu et al., 2022).

- Fine-tuned Language Models: We adopt three commonly used pre-trained language models in our study: BERT (Devlin et al., 2018), two versions of RoBERTa (Liu et al., 2019), specifically RoBERTa-base and RoBERTa-large.

- GLEM (Zhao et al., 2022) is an effective framework that fuses language models and GNNs in the training phase through a variational EM framework. We use the official source code[2] to reproduce its results.

- TAPE (He et al., 2023) utilizes Large Language Models like ChatGPT (Achiam et al., 2023) to generate pseudo labels and explanations for textual nodes, which are then used to fine-tune Pre-trained Language Models alongside the original texts. We reproduced its results using the official source code[3].

- SimTeG (Duan et al., 2023) employs a cascading structure specifically designed for textual graphs, utilizing a two-stage training paradigm. Initially, it fine-tunes language models and subsequently trains GNNs.

[1] We obtain the raw texts of each node from https://github.com/pmernyei/wiki-cs-dataset.

[2] https://github.com/AndyJZhao/GLEM

[3] https://github.com/XiaoxinHe/TAPE

	Cora	WiKiCS	CiteSeer	Ele-Photo
Ours$_{(\text{base})}$	92.03 ± 0.94	80.13 ± 0.31	84.52 ± 1.17	83.14 ± 0.10
Ours$_{(\text{large})}$	91.96 ± 0.77	80.55 ± 0.24	85.91 ± 0.99	84.34 ± 0.14

Table 5: Results using BERT as an alternative language model backbone for token reduction.

We conducted experiments using the official source code[4].

- ENGINE (Zhu et al., 2024) is an efficient fine-tuning and inference framework for text-attributed graphs. It co-trains large language models and GNNs using a ladder-side approach, optimizing both memory and time efficiency. For inference, ENGINE utilizes an early exit strategy to further accelerate. We reproduce its results using the official source code[5].

C Implementation Details

In this section, we give the implementations details about our method GraphBridge.

Training of Graph-Aware Token Reduction Module. We train only the cross-attention module and the classifier, keeping the encoding language models frozen. Each dataset undergoes 100 training epochs, with an early stopping patience of 10 epochs. The learning rate is explored within {1e-3, 5e-4, 1e-4}, and the regularization term β is set to 0.1.

Training of Language Models. Initially, we construct the sequence Q by sampling adjacent nodes using a random walk with restart sampler Γ (Zhu et al., 2022, 2024). The number of walk steps for sampling varies among {8, 16, 32}. Training epochs for the language models are adapted according to the dataset sizes: {4, 6, 8} for small datasets (*e.g.*, Cora, WikiCS, CiteSeer), {4, 6} for medium datasets (*e.g.*, ArXiv-2023, Ele-Photo, OGBN-Products), and {4} for the large-scale dataset (OGBN-ArXiv). We employ AdamW optimizers. The learning rate is explored within {1e-4, 5e-5, 1e-5} for full parameter fine-tuning of RoBERTa-base[6], and {1e-3, 5e-4, 1e-4} for tuning RoBERTa-large[7] using LoRA (Hu et al., 2021) with a rank of 8. For the large language model LLaMA2-7B[8], as outlined in Table 3, we use LoRA with a rank of 4 and a learning rate within {5e-4, 1e-4, 5e-5}.

Training of Graph Neural Networks. We train the GNNs models (*i.e.*, SAGE) subsequent to acquiring node representations from the language models. Specifically, The number of training epochs is designated within the range of {100, 200, 500}, complemented by an early stopping mechanism set at 20 epochs for each dataset. We utilize the Adam optimizers, and the learning rate is chosen from the set {1e-2, 5e-3, 1e-3}.

D Alternative LMs as Backbones for Token Reduction

Our graph-aware token reduction module is compatible with any language model as a backbone. In this section, we demonstrate the effectiveness of our token reduction module using BERT[9] (Devlin et al., 2018) as an alternative backbone. Table 5 presents the results when employing BERT for token reduction.

[4] https://github.com/vermouthdky/SimTeG
[5] https://github.com/ZhuYun97/ENGINE
[6] https://huggingface.co/FacebookAI/roberta-base
[7] https://huggingface.co/FacebookAI/roberta-large
[8] https://huggingface.co/meta-llama/Llama-2-7b
[9] https://huggingface.co/google-bert/bert-base-uncased