When Graph Tokens Sink: A Mechanistic Analysis of Graph Language Models

Anonymous Author(s)
Affiliation
Address
email

Abstract

Graph Language Models (GLMs) have become a promising direction for adapting Large Language Models (LLMs) to graph learning tasks. By transforming graph topology and node information into graph tokens, GLMs allow LLMs to jointly process structured graph inputs and textual instructions. Yet, it remains unclear how LLMs internally interpret these graph tokens and whether graph tokens act as meaningful carriers of graph structure. In this work, we analyze how LLMs process graph information through graph-token behavior in representative GLM architectures.

Findings. We find that the internal saliency of graph tokens in GLMs is not equivalent to graph information utilization. Graph sink tokens consistently emerge as activation-level outliers: they can be identified by massive activation values along a small set of hidden-state dimensions and are biased toward early graph-token positions. However, this activation-level saliency does not imply that these tokens are the main carriers of graph information. Unlike classical attention sinks in language and vision-language models, graph sink tokens do not necessarily attract the largest attention weights from query tokens. Through pruning, repositioning, and swapping interventions, we show that graph sink tokens are not the most important semantic or structural tokens for downstream prediction.

Implications. Together, these results suggest that after current GLMs map graph structure into the LLM token space, the resulting graph-token representations do not naturally form a fully usable topology-aware internal representation; instead, they exhibit a decoupling between activation-level saliency and graph-semantic utility. This decoupling points to limitations in existing graph-token construction, placement, and alignment mechanisms.

1 Introduction

Graph Language Models (GLMs) have emerged as a promising framework for adapting Large Language Models (LLMs) to relational learning tasks [Wang et al., 2023, Liu et al., 2024, Chen et al., 2024a]. By transforming graph topology and node features into discrete or soft "*graph tokens*," GLMs allow transformers to jointly process structured graph inputs alongside textual instructions. While these architectures show strong empirical performance across standard benchmarks [Jin et al., 2024, Ren et al., 2024], their central representational assumption remains largely unverified: that inherently non-Euclidean graph topologies can be flattened into a sequential token stream and faithfully parsed by an LLM without triggering architectural pathologies [Tang et al., 2024].

This representational assumption is directly at odds with known transformer dynamics. Indeed, mechanistic interpretability research has shown that transformers are highly prone to structural pathologies when processing sequential data. For instance, in text [Kovaleva et al., 2021, Wei et al., 2022] and vision [Luo et al., 2025] modalities, models routinely develop '*attention sinks*', *i.e.,*

Submitted to 40th Conference on Neural Information Processing Systems (NeurIPS 2026). Do not distribute.

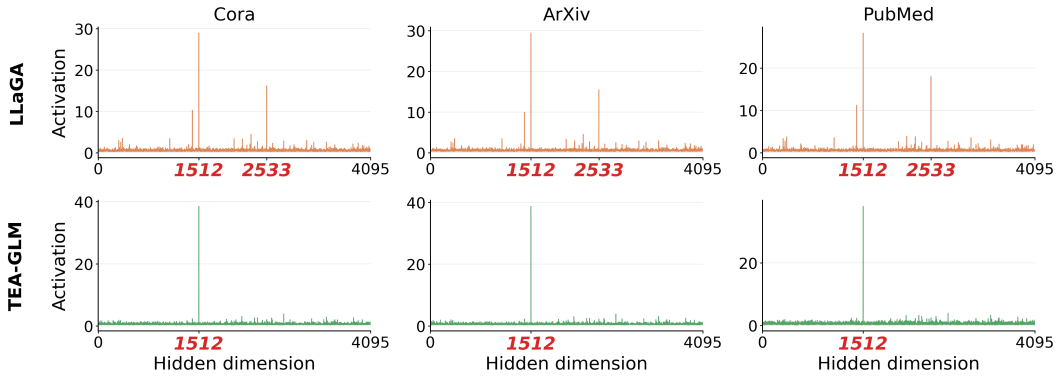

Figure 1: **Activation values across hidden dimensions for detected graph sink tokens on node classification.** We show average activation magnitudes over test samples for LLaGA and TEA-GLM on Cora, Arxiv, and PubMed. Graph sink tokens emerge as sparse activation-level outliers, with large spikes concentrated on a small set of hidden dimensions.

specific tokens that accumulate massive internal saliency and attention scores not to convey semantic meaning, but merely to stabilize the network's internal computations. This exposes a profound tension for GLMs. Graph tokens are explicitly injected to transmit vital structural information, yet they are subjected to the exact same transformer dynamics known to hijack tokens for non-semantic, architectural reasons. While initial studies have noted sink-like attention patterns in graph inputs [Guan et al., 2025], a much deeper vulnerability remains unresolved: when a graph token becomes internally 'loud' within the LLM, is the model actually utilizing it for topological reasoning, or is it merely exploiting the tokenized graph structure to build architectural artifacts?

Present work. In this work, we study how LLMs process graph tokens through the lens of graph sink behavior by auditing the underlying activation mechanics of two representative GLM designs: node-aligned graph tokens (LLaGA) and encoder-generated graph tokens (TEA-GLM). We demonstrate that in graph modalities, 'sink' behavior manifests primarily as activation outliers; consequently, we define *graph sink tokens* by their massive magnitude along specific hidden-state dimensions. Our mechanistic analysis reveals three key findings: First, graph sink tokens consistently emerge as sparse activation-level outliers and are heavily biased toward early graph-token positions, yet this internal prominence does not reliably translate into attention dominance or downstream utility: graph sink tokens do not necessarily receive the largest query-to-graph attention. Second, targeted intervention experiments show that these highly salient tokens contribute surprisingly little to downstream model performance. Finally, our interpretability analysis demonstrates—for the first time—that graph sink tokens predominantly capture weak, domain-level signals rather than the intended task-specific or topology-aware structural information. Together, these results reveal a severe decoupling between activation-level saliency and graph-semantic utility, suggesting that current GLMs do not automatically form usable topology-aware internal representations after mapping graph structure into the LLM token space.

2 Graph Sink Tokens Emerge as Activation-Level Outliers

Graph Sink Definition. Prior works on attention sinks in language and vision-language models show that sink tokens are often linked to massive activation patterns: a small number of tokens produce unusually large values on specific hidden dimensions, and these tokens may later receive disproportionate attention from other tokens [Xiao et al., 2023, Darcet et al., 2023, Sun et al., 2024, Kang et al., 2025, Luo et al., 2025]. We call these dimensions *sink dimensions*. In GLMs, we define graph sink tokens from the same activation-based view: a graph token is considered a sink token if its hidden state has a large response on the identified sink dimensions.

Following [Kang et al., 2025, Luo et al., 2025], let a GLM take a mixed-token sequence containing both text tokens and graph tokens. We denote the set of graph-token indices by $\mathcal{I}_g$, and let $\mathbf{x}_j^l \in \mathbb{R}^d$ be the hidden state of token j at layer l, where d is the LLM hidden dimension. The set of graph sink tokens at layer l is defined as

$$\widehat{\mathcal{I}}_g^l = \left\{ j \in \mathcal{I}_g \;\middle|\; \phi\left(\mathbf{x}_j^{l-1}\right) \geq \tau \right\}, \tag{1}$$

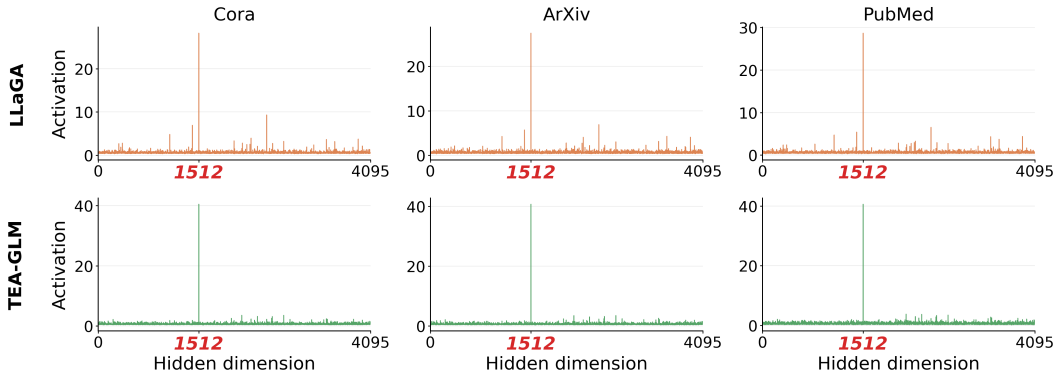

Figure 2: **Activation values across hidden dimensions for detected graph sink tokens on link prediction.** The same sparse activation pattern emerges across datasets and architectures, suggesting that the identified sink dimensions are stable across graph learning tasks.

where $\phi(\cdot)$ is a sink characteristic function and τ is a predefined threshold. In this work, we use activation magnitude as the sink characteristic. Given a set of sink dimensions $\mathcal{D}_{\text{sink}}$, we define

$$\phi\left(\mathbf{x}_j^{l-1}\right) = \max_{\bar{d} \in \mathcal{D}_{\text{sink}}} \left| \text{RMSNorm}\left(\mathbf{x}_j^{l-1}\right)_{\bar{d}} \right|. \tag{2}$$

Here, $\bar{d}$ denotes one sink dimension, and RMSNorm denotes Root Mean Square Normalization. Prior work has shown that sink dimensions can be language-model specific [Sun et al., 2024, Kang et al., 2025, Luo et al., 2025, Sun et al., 2026]. For example, LLaMA2-7B [Touvron et al., 2023] has been shown to contain sink dimensions such as $\{1415, 2533\}$. Since the GLMs studied here are built on LLaMA-family backbones, we ask whether their graph tokens simply inherit these dimensions, or whether graph-token processing induces new sink dimensions.

GLM Choices. We study two representative GLMs (LLaGA [Chen et al., 2024b] and TEA-GLM [Wang et al., 2024]) as they represent two common ways of passing graph information to an LLM. LLaGA uses a node-aligned graph-token sequence, where we use its Neighborhood Detail (ND) template: the center node is placed at graph sequence index 0 and its k-hop neighborhood is organized into a fixed-shape tree by sampling neighbors at each hop. If the sampled neighborhood is smaller than the budget, `[PAD]` tokens are inserted to maintain graph-token length. TEA-GLM follows a different design. It first uses a GNN encoder to aggregate neighborhood structure into a central-node representation and then projects this representation into a fixed number of learnable graph-token embeddings through a linear projector. These graph tokens are aligned with the LLM hidden space through contrastive learning against principal components of LLM token embeddings. We choose these models for two reasons: i) they represent two broad GLM families: node-aligned graph tokens and encoder-embedded graph tokens; and ii) both models use fixed-length graph-token sequences, which allows us to compare graph sink positions across samples, where K is the number of graph tokens.

Sink Activation Magnitudes. We run experiments on Cora [Yang et al., 2016], Arxiv [Hu et al., 2020], and PubMed [Sen et al., 2008], covering both node classification and link prediction. We set $\tau = 15.0$ in all settings. Figures 1 and 2 show the average hidden activation magnitude of detected graph sink tokens. For LLaGA, we average over 500 test samples per dataset; for TEA-GLM, we average over 300 test samples per dataset. The pattern is sharp and consistent. Across datasets and tasks, most hidden dimensions stay near a low activation range, while only a few dimensions form large spikes. In LLaGA, dimensions 1512 and 2533 repeatedly appear as dominant sink dimensions. In TEA-GLM, dimension 1512 is the dominant sink dimension across all datasets and both tasks. This suggests that graph sink behavior is not only inherited from known LLM sink dimensions, such as 2533 in LLaMA-based models. The repeated appearance of dimension 1512 across both GLM designs points to sink behavior that emerges after graph information is mapped into the LLM token space.

Sink Token Distributions. We next ask where these graph sink tokens appear in the graph-token sequence. For LLaGA, the graph-token sequence length is $K = 111$ for node classification and $K = 222$ for link prediction. For TEA-GLM, the graph-token sequence length is fixed as $K = 5$ for both tasks. Figs. 3-4 show the frequency of detected graph sink tokens at each graph-token index.

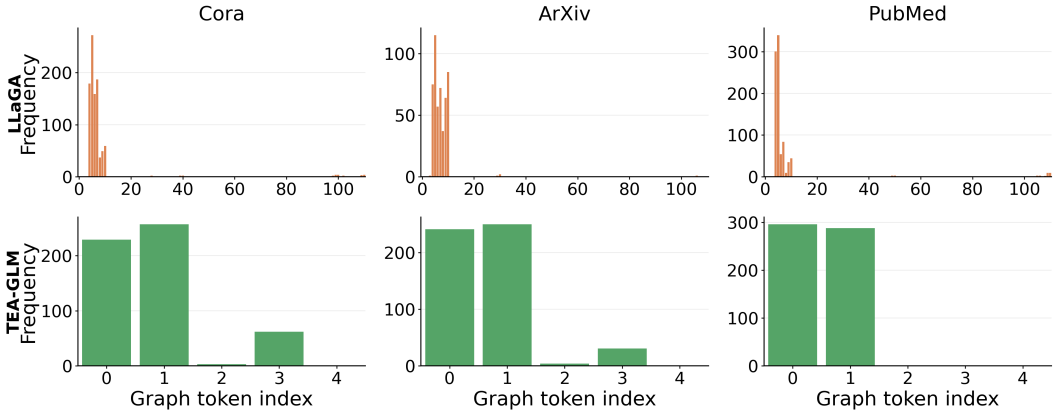

Figure 3: **Distribution of detected graph sink token positions on node classification.** For LLaGA, $K = 111$; for TEA-GLM, $K = 5$. Graph sink tokens are biased toward early graph-token positions. In LLaGA, the strongest sink tokens are consistently `[PAD]` tokens.

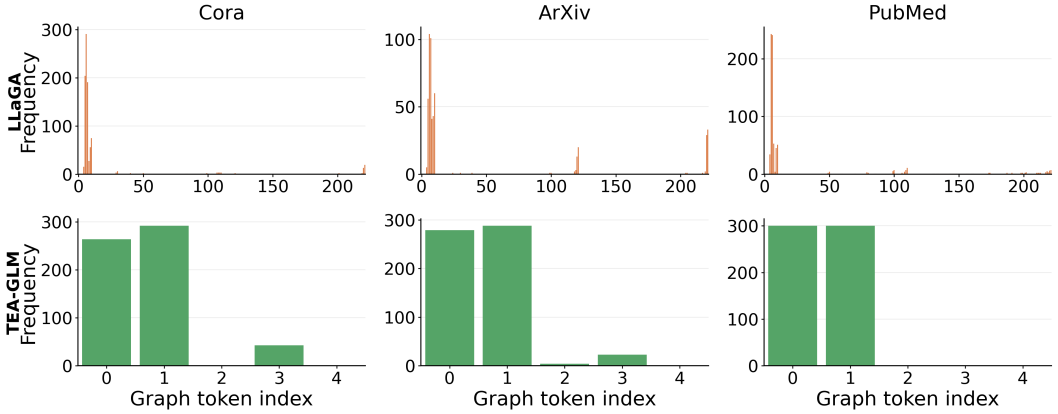

Figure 4: **Distribution of detected graph sink token positions on link prediction.** For LLaGA, $K = 222$; for TEA-GLM, $K = 5$. The position pattern is consistent with node classification: graph sink tokens mainly appear near the beginning of the graph-token sequence.

The distribution is not uniform. Across tasks, datasets, and models, graph sink tokens are biased toward early graph-token positions. For TEA-GLM, most sink tokens appear at positions 0 and 1, indicating that the first few encoder-generated graph-token slots are most likely to become activation outliers. LLaGA shows the same early-position bias, but the distribution is less compressed because its graph-token sequence is much longer. LLaGA also lets us inspect whether these sink positions correspond to meaningful graph roles. Under the ND template, index 0 is the center node, while later positions correspond to sampled neighbors or `[PAD]` tokens. We find that the top-2 graph sink tokens with the largest activation scores are always `[PAD]` tokens across all datasets and both tasks. By contrast, the center-node token at index 0 is never identified as a graph sink token. This already suggests a gap between activation-level saliency and graph-semantic importance: the most salient graph tokens by activation are not necessarily the most meaningful graph tokens by structure.

Key Takeaway. Graph sink tokens consistently emerge as activation-level outliers, identified by large values on a small set of hidden dimensions and biased toward early graph-token positions.

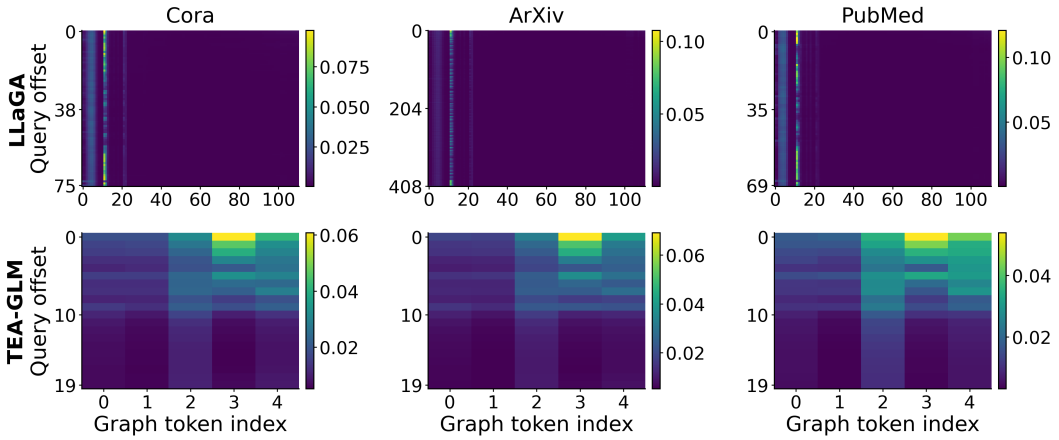

Figure 5: **Query-to-graph attention maps for node classification for both models across three datasets.** Attention weights are averaged over heads and test samples. In TEA-GLM, high-attention regions often appear on later graph-token indices rather than the main sink-token positions $\{0, 1\}$. In LLaGA, several graph-token positions receive stable attention across query offsets, but sink tokens are not necessarily the highest-attended tokens.

3 Activation-Level Saliency Does Not Imply Attention Dominance

Sec. 2 shows that graph sink tokens emerge as activation-level outliers: they have large responses on a small set of hidden dimensions and are biased toward early graph-token positions. This raises a critical question: *do these exceptionally "loud" tokens also dictate the routing of information by attracting the majority of attention weights?*

Query-to-Graph Attention. We first analyze attention from query to graph tokens during the pre-fill stage. For each model and dataset, we average attention weights over heads and test samples, and visualize how query-token positions attend to each graph-token index. Fig. 5 shows node classification results on Cora, Arxiv, and PubMed (link prediction results are provided in Appendix A.2). The TEA-GLM results show a clear separation between sink-token and high-attention positions. As shown in Sec. 2, TEA-GLM graph sink tokens mostly appear at indices 0 and 1. However, the average attention is often stronger on later graph tokens, especially indices 2–4. Conversely, the attention maps of LLaGA contain narrow vertical bands over several graph-token positions, indicating that these positions receive stable attention across query offsets. Some of these bands overlap with the graph sink region identified in Sec. 2. However, this attention is not clearly query-specific, and the average attention on sink positions is not necessarily higher than on non-sink positions. Since the top LLaGA sink tokens with the highest magnitudes are often `[PAD]` tokens, this stable attention is unlikely to correspond directly to useful node or neighborhood information.

Layer-wise Attention Patterns. Next, we check whether this separation also appears across transformer layers. Fig. 6 shows graph-token attention as a function of layer. For TEA-GLM, later graph-token indices again receive stronger attention than the main sink-token positions 0 and 1, especially in lower and middle layers. This shows that the mismatch between activation saliency and attention dominance is not an artifact of averaging over query positions; it is visible throughout the model. For LLaGA, the layer-wise plots again show thin vertical bands at fixed graph-token positions. These bands persist across layers, suggesting that LLaGA repeatedly routes some attention to the same graph-token positions. Yet the attention is spread across both sink and non-sink regions, rather than being concentrated only on detected graph sink tokens. This supports a more cautious interpretation: LLaGA graph sink tokens can participate in stable attention patterns, but they are not the sole or strongest pathway through which query tokens access graph information. This observation is consistent with recent findings on sink behavior in text-only LLMs. Sun et al. [2026] show that activation spikes and attention sinks often co-occur, but they are not strictly coupled. Our results suggest that the same distinction is important for graph tokens: a graph token can be highly salient in the hidden activation space without becoming the dominant target of query-to-graph attention.

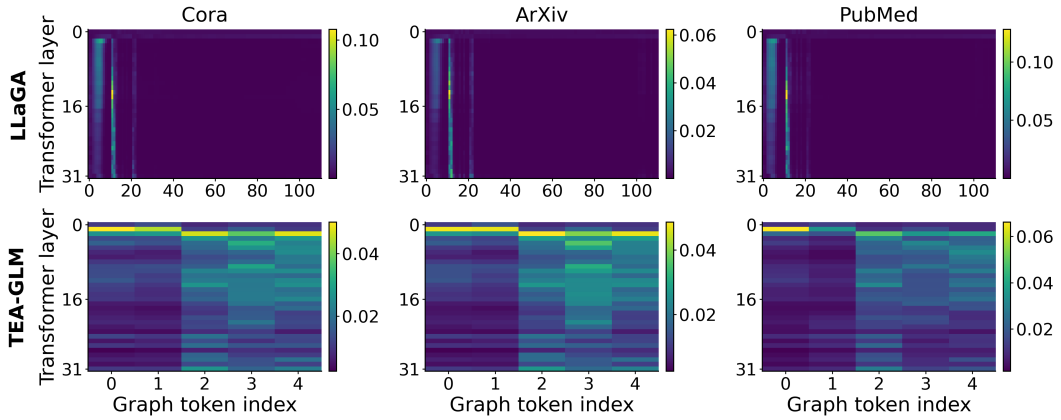

Figure 6: **Layer-wise query-to-graph attention maps for node classification for both GLMs.** Attention weights are averaged over heads and test samples, and the y-axis denotes transformer layers. TEA-GLM assigns stronger attention to later graph-token indices than to the main sink-token positions, while LLaGA shows stable vertical attention bands that are not exclusive to graph sink tokens.

Summary. Together, the two attention views point to the same conclusion. Graph sink tokens in GLMs are activation-salient, but this saliency does not reliably translate into attention dominance. TEA-GLM provides the clearest case, where sink tokens are located at early graph-token slots but attention often shifts to later graph tokens. LLaGA shows partial overlap between sink positions and stable attention bands, but these sink positions are often `[PAD]` tokens and are not consistently the highest-attended tokens. Therefore, graph sink tokens should not be treated as direct graph-domain versions of classical attention sinks.

> **Key Takeaway.** Graph sink tokens are activation-salient, but they do not necessarily become the dominant attention targets of query tokens.

4 Graph Sink Tokens Are Not the Main Semantic or Structural Carriers

In Sec. 3, we show that graph sink tokens are activation-level outliers, yet they do not necessarily receive dominant query-to-graph attention. This raises a functional question: *do these highly activated graph tokens actually carry the graph information used for prediction?* To answer this question, we move from observational analysis to direct graph-token interventions. We test whether removing, swapping, or repositioning graph sink tokens causes meaningful performance degradation.

Intervention setup. For each test sample, we first run the original model and identify graph sink token positions using Eq. 1. These positions are then fixed for all intervention runs on the same sample. We consider three interventions. TOP-2 sink removes the two graph sink tokens with the largest activation magnitudes. NON-SINK removes two randomly selected non-sink graph tokens, serving as a control for the effect of shortening the graph-token sequence; we average this setting over 15 random seeds. SWAP exchanges two sink-token positions with two randomly selected non-sink positions while keeping all token embeddings in the prompt; we average this setting over five random seeds.

Sink-token interventions have limited impact. Table 1 shows that pruning the top-2 graph sink tokens has little effect on node classification for both LLaGA and TEA-GLM. In contrast, pruning random non-sink tokens leads to larger drops in several settings, especially for LLaGA. We observe a related trend in link prediction, reported in Appendix A.1, Table 2: although the contrast is clearer in node classification, both tasks show that graph sink tokens are not the main graph-information carriers. Across node classification and link prediction, the most activation-salient graph tokens are often less important than ordinary non-sink tokens for prediction.

In LLaGA, graph tokens are designed to preserve neighborhood structure through the ND template, yet the sink tokens with top-2 magnitudes are not the ones whose removal degrades the performance the most. In TEA-GLM, graph tokens are learned as high-dimensional projections of GNN represen-

Table 1: Node classification performance under graph-token interventions. We compare the baseline model with pruning the top-2 graph sink tokens, randomly pruning two non-sink graph tokens, and swapping sink and non-sink token positions. Non-sink pruning is averaged over 15 random seeds, and swapping is averaged over five random seeds. Red values denote cases where non-sink pruning causes a larger performance drop than top-2 sink pruning.

Model	Intervention	Arxiv	Cora	PubMed
LLaGA	Baseline	77.00	88.40	94.60
	Top-2 Sink	77.00	88.00	95.00
	Non-sink	74.36±1.44	80.48±2.47	89.16±4.58
	Swap	76.64±0.27	87.74±0.39	94.12±0.27
TEA-GLM	Baseline	56.67	13.33	83.67
	Top-2 Sink	56.67	13.00	83.67
	Non-sink	44.40±1.13	12.80±1.72	83.40±1.09
	Swap	56.40±0.43	12.87±0.18	83.27±0.43

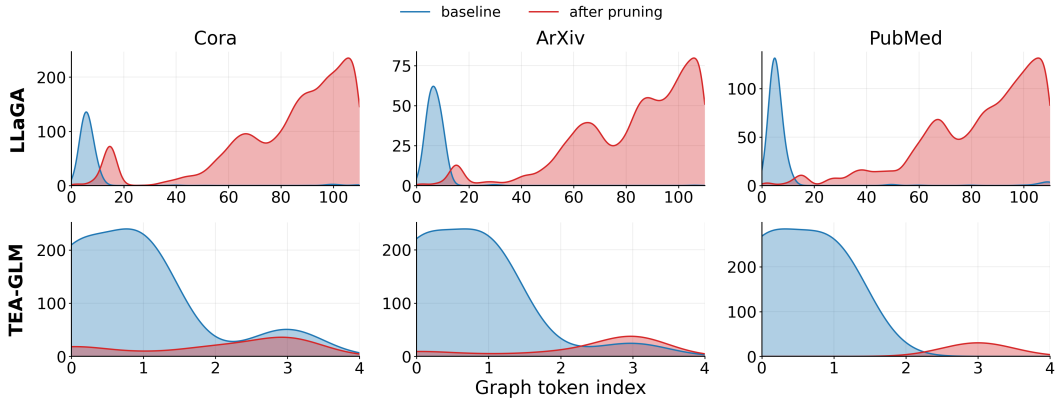

Figure 7: **Sink-position distribution shifts before and after pruning all identified graph sink tokens.** For LLaGA, sink tokens appear again after pruning but become more broadly distributed across graph-token positions. For TEA-GLM, activation magnitudes decrease after pruning, and the remaining graph tokens rarely satisfy the sink criterion. Note that the subplots for TEA-GLM are being smoothed for visualization purposes.

tations, but pruning sink tokens still has little impact. Moreover, on some TEA-GLM datasets, even pruning non-sink tokens causes only small changes, suggesting that the model may not consistently use graph-token semantics in zero-shot prediction.

The swap intervention leads to the same conclusion. Exchanging sink and non-sink positions changes performance only mildly, meaning that sink tokens are not strongly tied to sample-specific graph content or to a uniquely important position. For LLaGA, we further test moving sink tokens to the front of the graph-token sequence. Since LLaGA's ND template encodes graph topology through fixed token positions, such repositioning disrupts the original correspondence between sequence position and neighborhood structure. However, the results in Appendix A.1, Table 3, show little consistent impact on performance, suggesting that these sink tokens are not reliable graph summary or routing tokens. Together, pruning, swapping, and repositioning all suggest that graph sink tokens are not the main carriers of graph-structural information.

Do new sink tokens appear after pruning? A natural follow-up question is whether removing the original sink tokens simply causes other graph tokens to take over the sink role. To test this, after pruning *all* detected sink tokens, we run sink detection again on the pruned graph-token sequence and compare the sink-position distributions before and after pruning in Figure 7.

For LLaGA, sink tokens appear again after pruning, but their positions become much more spread out across the graph-token sequence rather than remaining concentrated near the earlier biased sink positions. This indicates that LLaGA maintains high-activation behavior even after the original sink

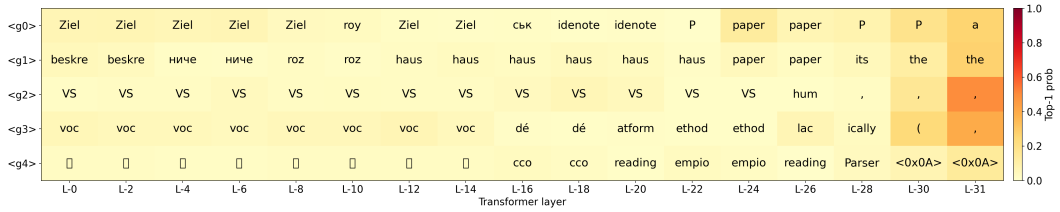

Figure 8: **Logit lens analysis for TEA-GLM graph tokens on PubMed node classification.** We select samples where graph sink tokens occur at positions g0 and g1. Each cell shows the most frequent top-1 decoded vocabulary token at a given graph-token position and transformer layer, while the color indicates its average probability across samples. Sink-token positions frequently decode to generic domain terms such as `paper` in later layers, but the overall probabilities remain low.

tokens are removed, but the sink role is redistributed across many graph-token positions. In other words, pruning does not eliminate the activation-sink pattern in LLaGA; it changes where it occurs and increases the spread of high-activation tokens.

TEA-GLM shows the opposite pattern. After pruning the original sink tokens, the overall activation magnitude decreases, and the remaining graph tokens no longer satisfy the sink criterion in most cases. This suggests that TEA-GLM sink behavior is more localized: once all sink tokens are pruned, the model does not create new sink tokens elsewhere. Combined with the limited performance change under sink pruning, this further suggests that TEA-GLM graph sink tokens are activation-level artifacts rather than reliable carriers of graph semantics.

Summary. Across pruning, swapping, repositioning, and post-pruning sink detection, graph sink tokens do not behave as the main semantic or structural carriers in graph language models. Removing the top-2 magnitude sink tokens usually causes little performance drop, while removing random non-sink tokens can be more harmful, especially for LLaGA. After pruning, LLaGA redistributes sink behavior across many graph-token positions, whereas TEA-GLM largely loses sink-qualified activations. These results connect back to our central observation: activation-level saliency does not imply graph-information utility.

Key Takeaway. Graph sink tokens are not the main graph information carriers.

5 Mechanistic Evidence for Limited Graph-Token Semantics

Sec. 4 shows that graph sink tokens are not prediction-critical, *i.e.,* pruning them causes little performance change, and post-pruning detection suggests that sink behavior is not tied to a fixed set of semantically essential graph tokens. We now ask a complementary mechanistic question: *what information is exposed by graph-token hidden states inside the LLM?* To answer this, we apply a logit lens analysis [nostalgebraist, 2020, Geva et al., 2021, Belrose et al., 2023, Agarwal, 2025] to TEA-GLM on node classification. We focus on TEA-GLM because it uses a compact fixed graph-token block, $K = 5$, making token positions comparable across samples, datasets, and layers. We select samples where graph sink tokens appear at positions g0 and g1, and decode each graph-token hidden state at each transformer layer into the LLM vocabulary space.

Logit lens setup. For each dataset, layer, and graph-token position, we aggregate selected test samples and record the vocabulary token that most frequently appears as the top-1 decoded token after projecting the graph-token hidden state through the LM head. We also report the average probability of this decoded token across selected samples. Thus, the text inside each cell denotes the most frequent decoded token, while the heatmap color denotes its average probability. This analysis tests whether graph-token hidden states expose task-relevant labels, node semantics, or topology-related concepts.

Decoded graph-token states are weakly topology-aware. Fig. 8 shows limited evidence that decoded graph-token states expose topology-aware or task-specific information. Additional results on Cora and Arxiv are reported in Appendix A.3. Across layers, many decoded outputs are fragmented subwords, punctuation marks, or generic terms, rather than class labels or structural concepts. The

decoded probabilities are also generally low, suggesting that most graph-token hidden states are not strongly aligned with stable vocabulary predictions.

A consistent pattern appears at the sink-token positions g0 and g1 across all three datasets. From approximately layer 20 onward, these positions frequently decode to `paper`. This is expected on citation-style datasets to some extent, but it is still informative: TEA-GLM is pretrained on Arxiv and then applied zero-shot to Cora and PubMed, so the repeated appearance of `paper` suggests that the early graph-token slots retain a broad citation-domain prior from pretraining. Rather than decoding dataset-specific labels or graph topology information, the sink-token states appear to expose generic domain-level signals.

Summary. This mechanistic evidence is consistent with the intervention results in Sec. 4. If graph sink tokens were the main carriers of task-critical graph structure, we would expect their decoded states to reflect labels, neighborhood information, or discriminative node semantics. Instead, they mostly decode to generic terms with low confidence. Together, these results suggest that activation-level saliency in graph tokens is not sufficient evidence of graph-semantic utility, and that TEA-GLM does not fully translate its graph-token representations into dataset-specific topology-aware information under zero-shot transfer.

> **Key Takeaway.** Logit lens analysis shows that graph sink tokens mainly expose generic domain-level terms rather than task-specific or topology-aware graph information.

6 Related Work

Graph Language Models. Recent work has increasingly explored how LLMs can be adapted to graph-structured data, especially for text-attributed graphs, citation networks, recommendation graphs, and molecular graphs [Jin et al., 2024, Li et al., 2023, Ren et al., 2024, Zhang et al., 2024]. Existing approaches can be roughly grouped by the role assigned to the LLM: some use LLMs as predictors over verbalized graph prompts [Ye et al., 2024, Chen et al., 2024b], some use LLMs as text encoders to enrich node features, and others align graph representations with the LLM embedding space for generative or instruction-following graph tasks [Wang et al., 2024, Zhao et al., 2023]. These methods show that LLMs can provide useful semantic priors for graph learning.

Attention Sinks. Attention sinks have been widely observed in Transformer-based models, where certain tokens receive disproportionate attention from other tokens despite limited semantic relevance [Xiao et al., 2024, Gu et al., 2024]. Related work on massive activations shows that sink behavior is often accompanied by extreme hidden-state values on a small number of feature dimensions [Dettmers et al., 2022, Sun et al., 2024, Oh et al., 2024]. Recent studies further show that attention sinks are not limited to text-only LLMs, but also appear in vision-language and multimodal models, where visual or perceptual tokens can form sink-like behavior with model- and task-dependent effects [Kang et al., 2025, Luo et al., 2025, Zhang et al., 2025, Cappellazzo et al., 2025].

7 Conclusion

We study how GLMs process graph tokens through the lens of graph sink behavior. Across LLaGA and TEA-GLM, we find that graph sink tokens consistently emerge as activation-level outliers, but their internal saliency does not translate into dominant query attention or functional importance for downstream prediction. Our intervention and logit lens analyses further show that graph sink tokens are not the main carriers of graph-semantic or structural information. These results suggest that mapping graph topology into the LLM token space does not by itself guarantee a fully usable topology-aware internal representation.

Limitations and Future Work. While we chose the two most representative GLM architectures used in current GLM research, other GLM architectures may exhibit different graph sink token behavior. Future work directions include how graph-token construction, placement, and graph-text alignment can be improved so that graph tokens better preserve topology-aware information inside LLMs.

References

Heng Wang, Shangbin Feng, Tianxing He, Zhaoxuan Tan, Xiaochuang Han, and Yulia Tsvetkov. Can language models solve graph problems in natural language? *Advances in Neural Information Processing Systems*, 36:30840–30861, 2023.

Zheyuan Liu, Xiaoxin He, Yijun Tian, and Nitesh V Chawla. Can we soft prompt llms for graph learning tasks? In *Companion proceedings of the ACM web conference 2024*, pages 481–484, 2024.

Zhikai Chen, Haitao Mao, Hang Li, Wei Jin, Hongzhi Wen, Xiaochi Wei, Shuaiqiang Wang, Dawei Yin, Wenqi Fan, Hui Liu, et al. Exploring the potential of large language models (llms) in learning on graphs. *ACM SIGKDD Explorations Newsletter*, 25(2):42–61, 2024a.

Bowen Jin, Gang Liu, Chi Han, Meng Jiang, Heng Ji, and Jiawei Han. Large language models on graphs: A comprehensive survey. *IEEE Transactions on Knowledge and Data Engineering*, 36 (12):8622–8642, 2024.

Xubin Ren, Jiabin Tang, Dawei Yin, Nitesh Chawla, and Chao Huang. A survey of large language models for graphs. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 6616–6626, 2024.

Jiabin Tang, Yuhao Yang, Wei Wei, Lei Shi, Lixin Su, Suqi Cheng, Dawei Yin, and Chao Huang. Graphgpt: Graph instruction tuning for large language models. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 491–500, 2024.

Olga Kovaleva, Saurabh Kulshreshtha, Anna Rogers, and Anna Rumshisky. Bert busters: Outlier dimensions that disrupt transformers. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3392–3405, 2021.

Xiuying Wei, Yunchen Zhang, Xiangguo Zhang, Ruihao Gong, Shanghang Zhang, Qi Zhang, Fengwei Yu, and Xianglong Liu. Outlier suppression: Pushing the limit of low-bit transformer language models. *Advances in Neural Information Processing Systems*, 35:17402–17414, 2022.

Jiayun Luo, Wan-Cyuan Fan, Lyuyang Wang, Xiangteng He, Tanzila Rahman, Purang Abolmaesumi, and Leonid Sigal. To sink or not to sink: Visual information pathways in large vision-language models. *arXiv preprint arXiv:2510.08510*, 2025.

Zhong Guan, Likang Wu, Hongke Zhao, Ming He, and Jianpin Fan. Attention mechanisms perspective: Exploring llm processing of graph-structured data. *arXiv preprint arXiv:2505.02130*, 2025.

Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. Smoothquant: Accurate and efficient post-training quantization for large language models. In *International conference on machine learning*, pages 38087–38099. PMLR, 2023.

Timothée Darcet, Maxime Oquab, Julien Mairal, and Piotr Bojanowski. Vision transformers need registers. *arXiv preprint arXiv:2309.16588*, 2023.

Mingjie Sun, Xinlei Chen, J Zico Kolter, and Zhuang Liu. Massive activations in large language models. *arXiv preprint arXiv:2402.17762*, 2024.

Seil Kang, Jinyeong Kim, Junhyeok Kim, and Seong Jae Hwang. See what you are told: Visual attention sink in large multimodal models. *arXiv preprint arXiv:2503.03321*, 2025.

Shangwen Sun, Alfredo Canziani, Yann LeCun, and Jiachen Zhu. The spike, the sparse and the sink: Anatomy of massive activations and attention sinks. *arXiv preprint arXiv:2603.05498*, 2026.

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

Runjin Chen, Tong Zhao, Ajay Jaiswal, Neil Shah, and Zhangyang Wang. Llaga: Large language and graph assistant. *arXiv preprint arXiv:2402.08170*, 2024b.

Duo Wang, Yuan Zuo, Fengzhi Li, and Junjie Wu. Llms as zero-shot graph learners: Alignment of gnn representations with llm token embeddings. *Advances in neural information processing systems*, 37:5950–5973, 2024.

Zhilin Yang, William Cohen, and Ruslan Salakhudinov. Revisiting semi-supervised learning with graph embeddings. In *International conference on machine learning*, pages 40–48. PMLR, 2016.

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

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

nostalgebraist. Interpreting gpt: The logit lens. `https://www.lesswrong.com/posts/AcKRB8wDpdaN6v6ru/interpreting-gpt-the-logit-lens`, 2020. LessWrong blog post.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. Transformer feed-forward layers are key-value memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5484–5495, 2021.

Nora Belrose, Igor Ostrovsky, Lev McKinney, Zach Furman, Logan Smith, Danny Halawi, Stella Biderman, and Jacob Steinhardt. Eliciting latent predictions from transformers with the tuned lens. *arXiv preprint arXiv:2303.08112*, 2023.

Chirag Agarwal. Rethinking explainability in the era of multimodal ai. *arXiv preprint arXiv:2506.13060*, 2025.

Yuhan Li, Zhixun Li, Peisong Wang, Jia Li, Xiangguo Sun, Hong Cheng, and Jeffrey Xu Yu. A survey of graph meets large language model: Progress and future directions. *arXiv preprint arXiv:2311.12399*, 2023.

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

Ruosong Ye, Caiqi Zhang, Runhui Wang, Shuyuan Xu, and Yongfeng Zhang. Language is all a graph needs. In *Findings of the association for computational linguistics: EACL 2024*, pages 1955–1973, 2024.

Haiteng Zhao, Shengchao Liu, Ma Chang, Hannan Xu, Jie Fu, Zhihong Deng, Lingpeng Kong, and Qi Liu. Gimlet: A unified graph-text model for instruction-based molecule zero-shot learning. *Advances in neural information processing systems*, 36:5850–5887, 2023.

Guangxuan Xiao, Jiaming Tang, Jingwei Zuo, Junxian Guo, Shang Yang, Haotian Tang, Yao Fu, and Song Han. Duoattention: Efficient long-context llm inference with retrieval and streaming heads. *arXiv preprint arXiv:2410.10819*, 2024.

Xiangming Gu, Tianyu Pang, Chao Du, Qian Liu, Fengzhuo Zhang, Cunxiao Du, Ye Wang, and Min Lin. When attention sink emerges in language models: An empirical view. *arXiv preprint arXiv:2410.10781*, 2024.

Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. Gpt3. int8 (): 8-bit matrix multiplication for transformers at scale. *Advances in neural information processing systems*, 35: 30318–30332, 2022.

Jaehoon Oh, Seungjun Shin, and Dokwan Oh. House of cards: Massive weights in llms. *arXiv preprint arXiv:2410.01866*, 2024.

Stephen Zhang, Mustafa Khan, and Vardan Papyan. Attention sinks: A'catch, tag, release'mechanism for embeddings. *arXiv preprint arXiv:2502.00919*, 2025.

Umberto Cappellazzo, Stavros Petridis, Maja Pantic, et al. Mitigating attention sinks and massive activations in audio-visual speech recognition with llms. *arXiv preprint arXiv:2510.22603*, 2025.

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

A Appendix

A.1 More Intervention Results

We provide additional intervention results that are omitted from the main paper due to space constraints. Table 2 reports the link prediction results under pruning and swapping interventions. Table 3 reports the LLaGA repositioning experiment for both node classification and link prediction tasks, where detected graph sink tokens are moved to the front of the graph-token sequence.

The swap and repositioning interventions both alter the original graph-token order and therefore disturb the topology pattern encoded by LLaGA's fixed graph-token template. However, repositioning is a stronger positional intervention than swapping for LLaGA: it moves the detected sink tokens to the front of the graph-token sequence, following the idea of sink-token repositioning as a mitigation technique in prior vision attention sink analysis [Luo et al., 2025], but here we use it as a diagnostic intervention. We do not apply repositioning to TEA-GLM because its graph sink tokens already appear at positions 0 and 1 in most cases, so moving them to the front would not create a meaningful additional intervention.

Table 2: Link prediction performance under graph-token interventions. We compare the baseline model with pruning the top-2 graph sink tokens, swapping graph sink and non-sink token positions, and randomly pruning two non-sink graph tokens. Red values denote cases where non-sink pruning causes a larger performance drop than top-2 sink pruning.

Model	Intervention	Arxiv	Cora	PubMed
LLaGA	Baseline	91.40	83.60	87.00
	Top-2 Sink	90.80	83.60	87.60
	Non-sink	90.76$\pm$0.39	82.04$\pm$0.39	87.08$\pm$0.37
	Swap	90.60$\pm$0.33	83.40$\pm$0.81	88.04$\pm$0.27
TEA-GLM	Baseline	57.33	63.67	62.33
	Top-2 Sink	52.67	60.33	60.67
	Non-sink	54.04$\pm$3.22	59.20$\pm$2.35	61.42$\pm$2.62
	Swap	56.67$\pm$0.62	63.40$\pm$1.12	59.40$\pm$0.72

Table 3: Performance comparison for LLaGA under graph sink token repositioning intervention. We move the detected graph sink tokens to the front of the graph-token sequence and evaluate the effect on downstream performance.

Task	Reposition Setting	Arxiv	Cora	Pubmed
Node	Top-2 Sink to Front	77.00	87.60	95.00
	All Sink to Front	76.40	88.20	94.60
Link	Top-2 Sink to Front	91.60	82.80	87.40
	All Sink to Front	89.80	82.00	85.20

A.2 Attention Patterns of Graph Sink Tokens on Link Prediction

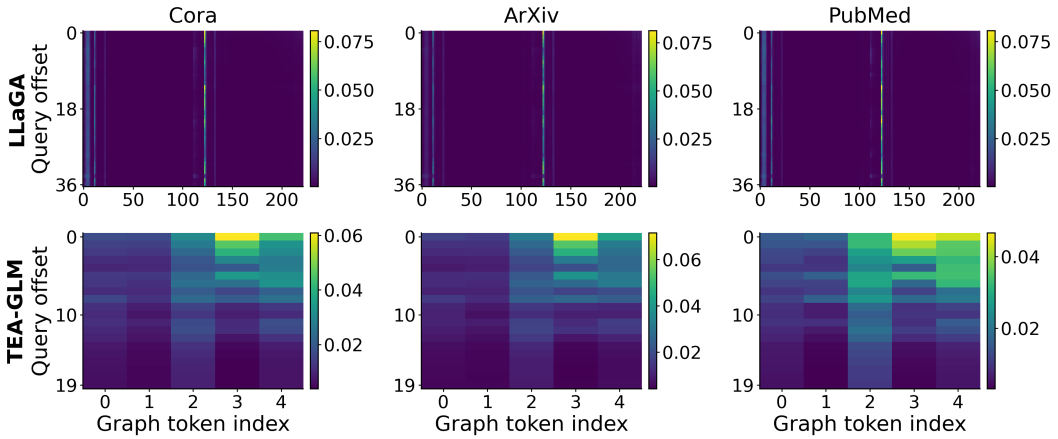

Figure 9: **Query-to-graph attention maps for link prediction.** Attention weights are averaged over heads and test samples.

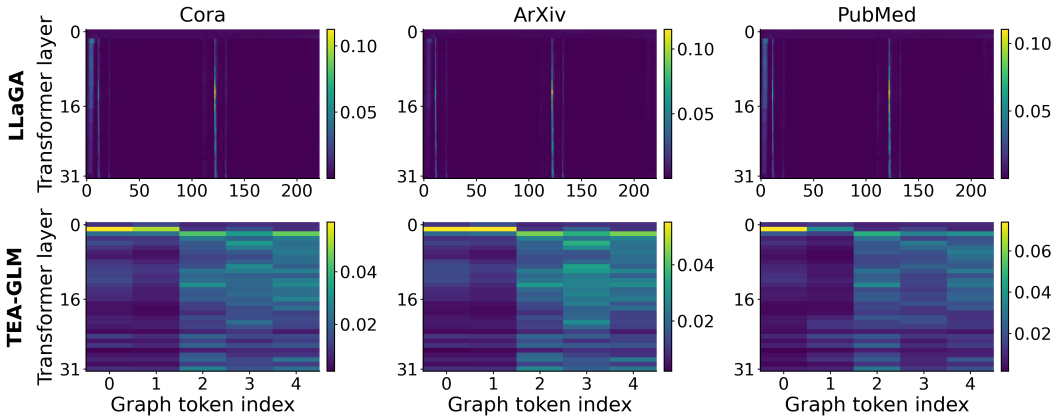

Figure 10: **Layer-wise query-to-graph attention maps for link prediction.** Attention weights are averaged over heads and test samples, and the y-axis denotes transformer layers.

A.3 More Logit Lens Results

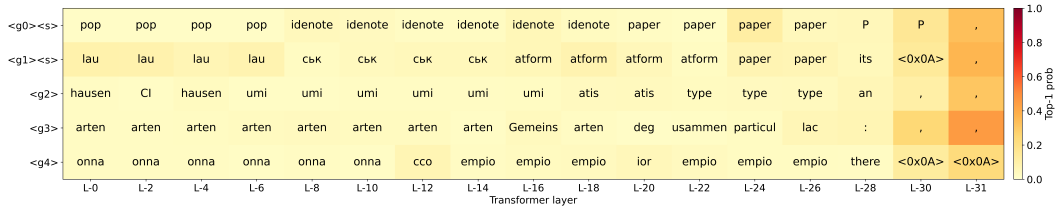

Figure 11: **Logit lens analysis for TEA-GLM graph tokens on Arxiv node classification.** Each cell shows the most frequent top-1 decoded vocabulary token for a graph-token position and transformer layer; color denotes the averaged probability across selected samples. Sink-token positions g0 and g1 frequently decode to generic citation-domain terms such as `paper` in later layers.

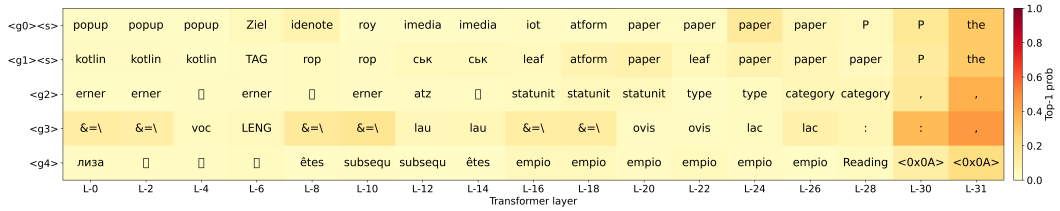

Figure 12: **Logit lens analysis for TEA-GLM graph tokens on Cora node classification.** Following the same setup as Fig. 11, decoded graph-token states mostly correspond to fragmented subwords or generic terms, with later layers showing citation-domain tokens such as `paper`.

A.4 Graph Sink Analysis on InstructGLM

In addition to LLaGA and TEA-GLM, we conduct a prototyping experiment on sink dimension activation analysis for InstructGLM [Ye et al., 2024] on the same three datasets, with $\tau = 10$ for all datasets. Figures 13 and 14 show that InstructGLM follows the same broad sink pattern observed in the main results. Its detected graph sink tokens are sparse activation-level outliers: most hidden dimensions remain at low magnitude, while only a small number of dimensions produce large spikes. The dominant sink dimension differs across datasets, with dimension 3840 appearing on Cora and dimension 3968 appearing on Arxiv and PubMed.

The position distribution also shows a strong early-token bias. Most graph sink tokens occur near the beginning of the graph-token sequence across all three datasets. These appendix results support the main finding that graph sink tokens emerge as activation-level outliers and tend to concentrate near early graph-token positions across GLM architectures.

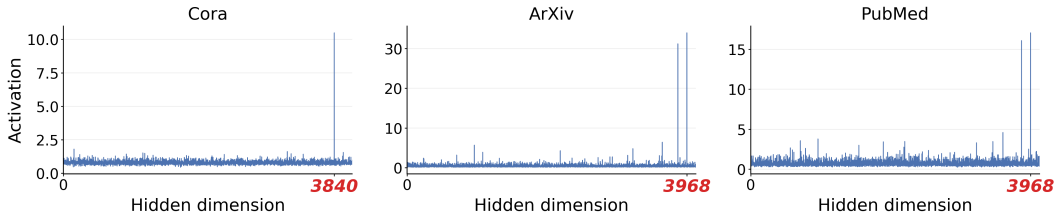

Figure 13: **Activation values across hidden dimensions for detected graph sink tokens in InstructGLM on node classification.** Results are averaged over 300 test samples for Cora, Arxiv, and PubMed. InstructGLM also exhibits sparse activation-level outliers, with dominant spikes appearing at a small number of hidden dimensions.

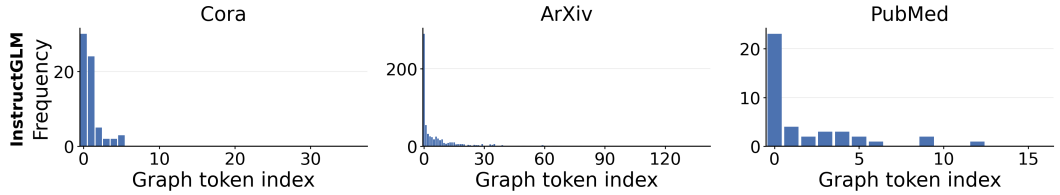

Figure 14: **Distribution of detected graph sink token positions in InstructGLM on node classification.** Results are averaged over 300 test samples for Cora, Arxiv, and PubMed. Graph sink tokens are strongly biased toward early graph-token positions across datasets.

A.5 Effect of Non-padded Graph Tokens on Sink Attention

We provide an analysis on whether LLaGA's attention to graph sink tokens depends on the amount of real graph content in the prompt. For each test sample, we compute the percentage of non-padded graph tokens in the graph-token sequence and measure the average query-to-graph attention assigned to the top-2 graph sink tokens. Fig. 15 reports this relationship across Arxiv, Cora, and PubMed.

Across all three datasets, we observe a clear decreasing trend: samples with fewer non-padded graph tokens assign higher attention to the top-2 sink tokens, while samples with more non-padded graph tokens assign less attention to these sink positions. Since the top-2 LLaGA sink tokens are consistently `[PAD]` tokens, this suggests that sink-token attention is partly tied to prompt sparsity. When the graph-token block contains less real neighborhood information, the model allocates more attention to padded sink positions; when more real graph tokens are present, attention shifts away from these padding-based sinks. This further supports our main claim that LLaGA graph sink tokens are not reliable carriers of graph-semantic content, but are strongly shaped by the graph-token construction and padding pattern.

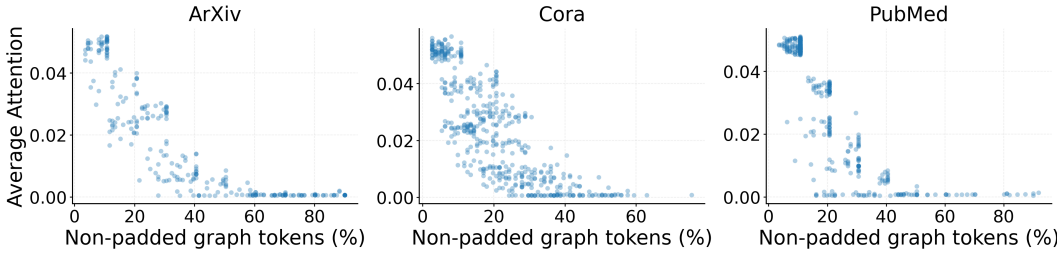

Figure 15: **Relationship between graph-token sparsity and attention to top-2 graph sink tokens in LLaGA.** The x-axis denotes the percentage of non-padded graph tokens in the graph-token sequence, and the y-axis denotes the average attention assigned to the top-2 graph sink tokens. Across Arxiv, Cora, and PubMed, attention to the top-2 sink tokens decreases as the proportion of non-padded graph tokens increases.

A.6 Implementation Details

Inference hyperparameters. For both LLaGA and TEA-GLM, all reported results are obtained using deterministic greedy decoding with a fixed random seed of 42 for reproducibility. Unless otherwise stated, we disable sampling and beam search during generation. All experiments are conducted on two NVIDIA A100 GPUs.

LLaGA. For LLaGA, we use the publicly released checkpoint with `vicuna-7b-v1.5-16k` as the language-model backbone. We follow the Vicuna v1 conversation template. Graph nodes are encoded with the SimTeG embedding stack [Duan et al., 2023] with a 4096-dimensional node representation. We use both the Neighborhood Detail (ND) template for node classification and for link prediction, with a sampling fan-out of 10 neighbors per hop and a maximum hop of 2. During inference, we call `model.generate` with `do_sample=False`, `temperature=0.0`, `top_p=None`, `num_beams=1`, and `max_new_tokens=1024`. We enable KV caching and return hidden states and attention weights for our analyses.

TEA-GLM. For TEA-GLM, we train a citation-domain checkpoint on ogbn-Arxiv using `vicuna-7b-v1.5` as the frozen LLM backbone. During training, only the graph encoder, cross-modal projector, and learned graph-token embeddings are updated. The graph branch is initialized from a pretrained two-layer GraphSAGE encoder with hidden size 2048 and output dimension 4096, matching the projector configuration. Each prompt is augmented with $K = 5$ learned graph tokens and uses `max_text_length`$= 700$ for ogbn-Arxiv. Training is launched with `accelerate` on two GPUs using a per-device batch size of 8 and gradient accumulation steps of 2. We use AdamW with learning rate 1×10^{-4}, a cosine learning-rate schedule, and a warmup ratio of 0.03. The model is trained for 50 epochs, and we use the final checkpoint for all TEA-GLM transfer evaluations reported in the paper.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Yes, the claims made in the abstract and introduction are reflected and elaborated throughout the paper and appendix.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The paper explicitly includes a "Limitations and Future Work" in Section 7.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: Please refer to Section 2 for more details.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: We provide experimental setups in Section 2 through Section 5, and more results in the Appendix. Code will be released following the review process.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We use datasets and models that are publicly available. Detailed implementations in Appendix A.6.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

Answer: [Yes]

Justification: Please refer to Sections 2- 5, and Appendix A.6 for more details.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: We report error bars and distributional information where applicable.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.

- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: Please refer to Appendix A.6.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

Answer: [Yes]

Justification: We followed the NeurIPS Code of Ethics.

Guidelines:

- The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [N/A]

Justification: While positive results and impacts are discussed in Sections 2- 5, there are no negative societal impacts in our work that need to be discussed.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A]

Justification: The paper does not release models or datasets that pose a high risk of misuse, and thus safeguards are not applicable.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: The creators or original asset owners have been credited in appropriate ways.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: We do not introduce any new assets in this work.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve human subjects or crowdsourcing.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: No human subjects are involved, so IRB approval is not required.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: The core methodology, scientific rigorousness, or originality of the research does not involve the usage of LLMs.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.