Simplifying Graph Transformers

Anonymous Authors[1]

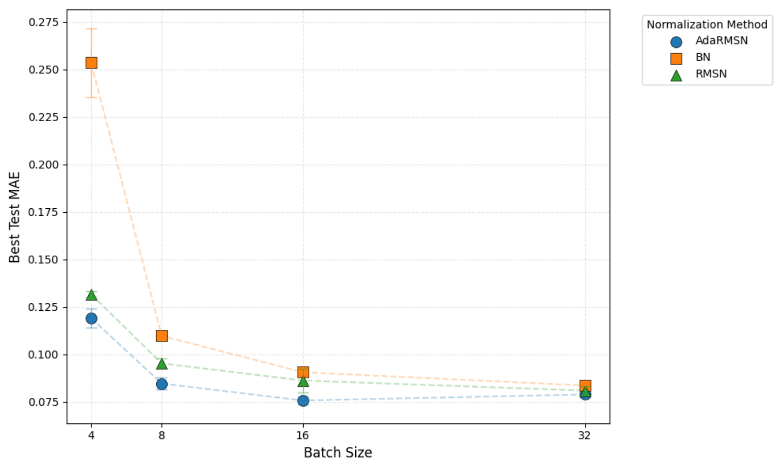

Figure 1. SGT on ZINC: Test MAE v.s. Batch Size (BS). # Training epochs are adjusted per batch-size for the same total update steps: $400 * \mathrm{BS}/32$. The first 10% epochs are in the warmup stage.

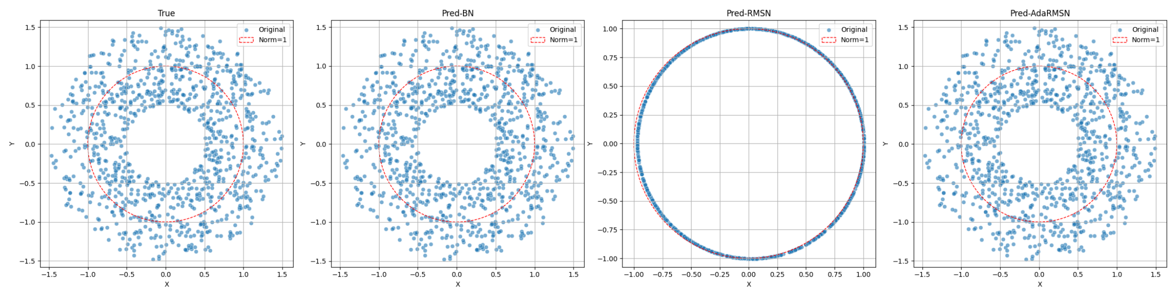

Figure 2. (Case Study of AdaRSMN) Visualization of Input and Pred data points. Ovefitting test on Auto-encoders of 2-dim (*Linear* → *BN/RMSN/AdaRMSN* → *Linear*): each model is trained 5000 epochs via AdamW without regularization. (together with Figure. 3)

Table 1. Performance on ZINC. GPS+sL_2: integrating sL_2 attention into GPS without changing other parts. (run 3 trials)

ZINC	GPS	GPS+sL_2	SGT
MAE (↓)	0.070 ± 0.004	0.0693 ± 0.0023	0.0566 ± 0.002

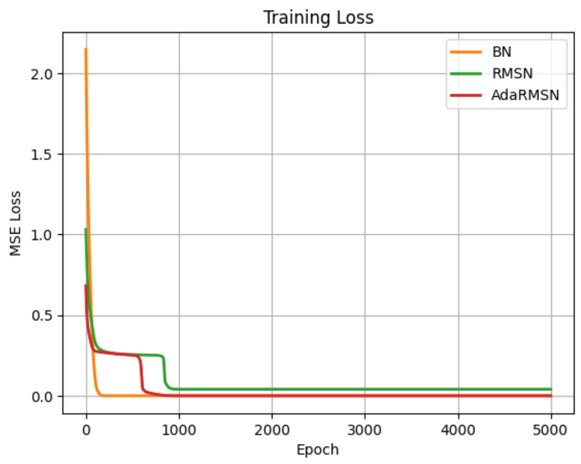

Figure 3. (Case Study of AdaRSMN) Training curves of overfitting test. (together with Figure. 2)

Table 2. Comparison of peak GPU memory usage and per-epoch training time for GRIT and SGT. Dataset: Peptides-Structure (15K graphs); Model config.: 5 transformer layers, 96 channels, batch size 32. Hardware: a single Nvidia V100 GPU with 32GB memory, supported by 80 Intel Xeon Gold 6140 CPUs running at 2.30GHz

Model	**GPU Memory (GB)**	**Training Time (Sec/Epoch)**
GRIT	29.16	141.60
SGT	25.07	100.68
Improv.	∼14.03%	∼28.9%

Table 3. Performance on PCMQM4Mv2 (over 3.7M graphs). The eval. pipeline follows Rampášek et al. (2022); no 3D-info included.

PCMQM4Mv2	**Val MAE (↓)**	**# Param.**
Graphormer	0.0864	48.3M
GPS	0.0858	19.4M
GRIT	0.0859	16.6M
SGT	0.0856	17.6M

Table 4. Performance comparison across different models on various datasets. Best results are highlighted in bold. * indicates the difference to the best is not statistically significant (by two-tail T-test)

Model	ZINC	SP-CIFAR	SP-MNIST	PATTERN	CLUSTER	Peptides-Struct	Peptides-Func
	MAE (↓)	Acc. (↑)	Acc. (↑)	W.Acc. (↑)	W.Acc. (↑)	MAE (↓)	AP (↑)
Exphormer	-	74.69±0.125	98.55±0.037	86.74±0.015	78.07±0.037	0.2481±0.0007	0.6527±0.0043
GEAET	-	76.634±0.427	98.513±0.086	86.993±0.026	-	**0.2445±0.0013**	-
GEANet	0.193±0.001	73.857±0.306	98.315±0.097	85.607±0.038	77.013±0.224	0.2512±0.0003	0.6722±0.0065
SGT	**0.0566±0.002**	**78.560±0.700**	**98.614±0.096**	**89.752±0.030**	**80.027±0.114**	0.2450±0.0017*	**0.6961±0.0062**

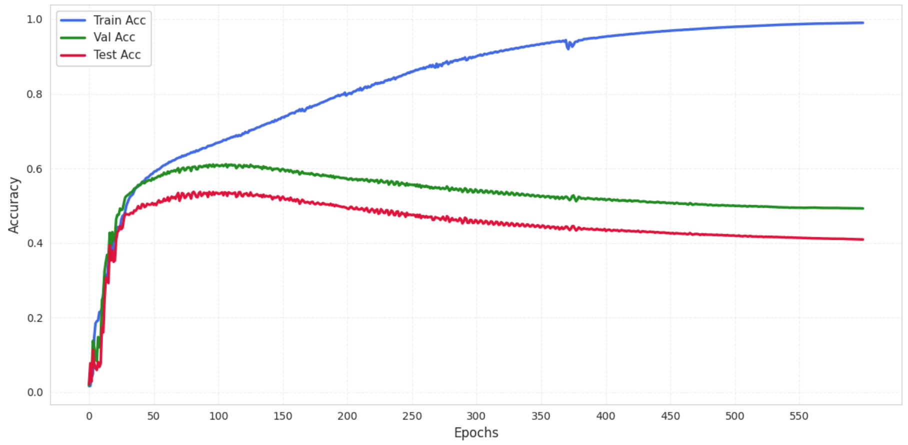

Figure 4. Sanity check of Exp-SGT on a large-scale graph in OGBN-ArXiv (169,343 nodes). Use the same configuration as Exphormer and remove all regularizations to validate the trainability via an overfitting test.

Red color is highlighting ; “Purple is the experiments to add”.

[Mark : I don’t really understand what this is responding to?] [liheng: Move to here from Reviwer pJA6; to rearrange the placement] (General response):

We thank the reviwer for the acknowledgement of our work.

We would like to emphasize that our primary contribution is to demonstrate that our proposed graph Transformers, adhering to plain Transformer architectures with our proposed minor modifications, can effectively learn from graph data. This contrasts with the findings in previous studies that plain Transformers are ineffective for graph learning (Dwivedi & Bresson, 2021), while good-performing graph Transformers often depend on complex modifications, such as additional MPNN layers (Rampasek et al., 2022) and intricate attention mechanisms (Ma et al., 2023).

This approach enables us to leverage the extensive training and architectural design advances explored for Transformers across various domains. Furthermore, it positions our method to seamlessly integrate with existing vision and language foundation models, offering significant potential for cross-domain applications and knowledge transfer.

[Mark : I don’t think it’s a great idea to quote other reviewers in a response to this reviewer. If this is a general response to all reviewers, then that would be OK.] This contribution is kindly acknowledged by Reviwer bTCe (“I really like the core motivation of the paper, which rethinks how these models are built while ensuring they remain close to vanilla Transformers, allowing them to leverage computational efficiency advances made in standard Transformers.”), Reviwer sPMh (“The paper addresses the issue of overly complex existing GT architectures by proposing a simplified GT architecture, with a solid motivation, as current GT models are indeed too complex.”), and Reviwer pJA6 (“The simplified design of graph transformer will significantly impact the community on the fundamental usage of GT...”).

1. Reviews

1.1. Reviewer bTCe – 2

Strengths:

- The paper is well written, easy to follow.
- I really like the core motivation of the paper, which rethinks how these models are built while ensuring they remain close to vanilla Transformers, allowing them to leverage computational efficiency advances made in standard Transformers.

Q1: The authors claim their modifications fundamentally improve transformer blocks for graph transformers. However, these seem plug-and-play yet are only tested on a vanilla transformer? The goal of our paper is not to propose plugins to improve transformer blocks for graph Transformers (GTs). Instead, our main contribution is to demonstrate that plain/vanilla Transformer architectures can work effectively for learning graph representations.

As discussed in our Introduction (Section 1) and Related Work (Section 4), initial attempts using plain/vanilla transformer architectures for graph learning tasks (e.g., Dwivedi & Bresson, 2021) proved ineffective. Consequently, previous graph transformers introduced complex modifications to address these limitations, such as incorporating additional MPNN layers (Rampasek et al., 2022) and implementing intricate attention mechanisms (Ma et al., 2023). However, these modifications have led to the architectures diverging significantly from vanilla Transformers, hindering the adoption of training advances developed for Transformers in other domains and preventing potential integration with other foundation models. This undermines the original motivation for using Transformers in graph learning.

Unlike previous works, our work aims to uncover the previous misunderstanding about plain Transformers and demonstrate that plain Transformers can effectively learn graph-structured data with minor modifications that are also adaptable to other domains.

Importantly, we have demonstrated that SGT, as a plain/vanilla Transformer, outperforms (or at least performs as well as) previous more complicated graph Transformers in graph learning tasks, as evidenced by our experiments on real-world graph benchmarks and the empirical expressivity benchmark.

Q2 Would they enhance other architectures? For example, does replacing SDP attention with sL_2 in GPS improve performance? Can the authors evaluate this?

Our proposed designs could potentially enhance other architectures.
““To add 1. Exphormer@arxiv and GPS@zinc experiment””.

However, the performance improvement may vary depending on the choice of MPNNs and attention mechanisms in GPS, as well as the datasets. This variation occurs because, in many cases, the MPNN component plays a more significant role than the attention mechanisms in GPS, as evidenced by their ablation studies (Tables B.1-B.4 in Appendix B).

It is important to note that GPS does not effectively incorporate graph positional encodings into its attention mechanisms and heavily relies on its MPNN component. Consequently, its expressivity does not conform to the GD-WL framework (Zhang et al., 2023), and there is no guarantee that it can exceed the 1-WL test. This expressivity limitation corresponds with GPS's inferior empirical performance compared to SGT.

- Zhang B, Luo S, Wang L, He D. Rethinking the Expressive Power of GNNs via Graph Biconnectivity. In ICLR 2023.

[Mark : I think it would be a good idea to explain why the presence of the MPNN may not be desirable. After reading this response, a reviewer might say: "well if the MPNN is doing the heavy-lifting and the architecture performs well, then why try to remove it?"]

[liheng: added explanation]

Q3 The proposed design changes appear highly modular, as hinted in the paper. I strongly recommend testing these modifications in another graph Transformer model, such as GPS, to evaluate whether similar improvements can be observed. This experiment would significantly strengthen the paper's claims and impact.

Same as Q2.

Q4: The use of the ZINC dataset—primarily categorical for the ablation study is not convincing enough to demonstrate benefits from the sL_2 attention and the adaptive normalization.
The choice of ZINC for the ablation study may not be ideal, as its node and edge features are categorical (atom and bond types). Since the authors propose methods related to normalization and magnitude sensitivity, it would be more informative to conduct these analyses on datasets containing continuous-valued node features (e.g., QM9, PPI), where magnitude information plays a significant role.

First, it is important to note that both AdaRMSN and sL_2 attention operate on continuous-valued token representations rather than directly on the raw categorical node and edge attributes. Therefore, the categorical nature of ZINC's attributes does not diminish the validity of our ablation study.

Second, due to the usage of RRWP as continuous positional encoding (PE) concatenated to node/edge attributes, we *de facto* use a mixture of categorical and continuous attributes, which is actually the same case as the QM9 dataset (categorical attributes plus 3D coordinates).

The PPI dataset, though with continuous attributes, is a very small-scale dataset with only 24 graphs in total, and thus is not sufficient for training a powerful and complicated model like Transformers. Therefore, considering these factors, we argue that the ablation study on ZINC is sufficiently representative, compared to using QM9 and PPI.

[Mark : Is it possible to do the test on QM9? If possible, it's always better to satisfy a reviewer's request rather than arguing that it's unnecessary.] [liheng: The current pipeline does not support QM9; and QM9 has 129,433 graphs. Might take too long to run ablation study compared to ZINC (12K nodes).]

Q5: The benchmarks used for evaluation are limited to small- to medium-scale graphs. This does not align well with the core motivation of developing scalable Graph Transformers that remain close to vanilla Transformer architectures . Without testing on large-scale datasets (e.g., OGB), it is unclear whether the proposed modifications hold up in more realistic graph settings.
The benchmarks chosen for evaluation are appropriate, covering a variety of small- to medium-scale datasets. However, while the authors do not explicitly claim scalability to large graphs, they motivate their work in the context of building Graph Foundation Models, which implies applicability to larger datasets .
Given that much of Graph Transformer research prioritizes efficiency on large-scale graphs , the authors should evaluate their method on larger benchmarks (e.g., OGB) to convincingly demonstrate its scalability.
My primary concerns lie in the experimental validation, which lacks critical elements needed to convincingly support the

paper's claims.
Performance on Larger Graphs: Given that a lot of prior graph Transformer approaches often introduce complex mechanisms specifically to handle larger-scale graphs, can the authors test their approach on larger-scale datasets, such as datasets from OGB?

We thank the reviewer for the suggestions.

We respectfully clarify that the reviewer may have misinterpreted our core motivation. Our primary goal is to develop **expressive** Graph Transformers that remain close to the vanilla Transformer architecture, rather than "developing **scalable** Graph Transformer that ...". These represent two distinct research directions that are often mutually exclusive in practice.

We contend that our focus on expressive power better aligns with the motivation of Graph Foundation Models, which require strong capacity to learn from large-scale datasets containing numerous graphs, rather than the scalability to large-scale-graph datasets that typically contain a single graph per dataset (i.e., small scale in terms of the amount of examples). This setting also better aligns with the **realistic graph setting – molecular property prediction (MPP)**, which plays a crucial role in advancing critical scientific fields such as drug discovery, materials science, and environmental chemistry. The benchmarks selected for our evaluation are also representative, encompassing both graph-level and node-level tasks.

Therefore, we respectfully argue that the criticism on our choice of evaluation benchmarks is inappropriate.

To better distinguish between the two distinct notions of "large scale" in graph learning, we present a further comparison:

1. **Large-scale (graph) datasets** (a large number of examples): This aligns with the conventional understanding of scale in machine learning across domains like language and vision, where model capacity is the crucial factor of learning from the vast volume of training data.
2. **Large-scale-graph dataset** (a large number of nodes in a graph): This actually corresponds to the challenges in long-context learning for language models and high-resolution image processing in vision models, where individual instances require significant computational resources. It is worth mentioning that large-scale-graph datasets are usually challenging in terms of **efficiency** but relatively easy w.r.t. **model capacity**.

Last but not least, graph Transformers (Wu et al., 2022; Wu et al., 2023; Chen et al., 2023) that prioritize applicability to large-scale graphs do not target graph-level tasks due to their compromise in reduced model capacity. Therefore, by analogy, SGT cannot and does not try to compete with these models in terms of applicability to large-scale graphs, as we focus on stronger capacity and generality, motivated by the potential construction of a graph foundation model.

- Wu Q, Zhao W, Li Z, Wipf D, Yan J. Nodeformer: A scalable graph structure learning transformer for node classification. In NeurIPS 2022. - Wu Q, Yang C, Zhao W, He Y, Wipf D, Yan J. DIFFormer: Scalable (Graph) Transformers Induced by Energy Constrained Diffusion. In ICLR 2023. - Chen J, Gao K, Li G, He K. NAGphormer: A Tokenized Graph Transformer for Node Classification in Large Graphs. In ICLR 2023.

"Running experiments on PCQM4Mv2 datasets, which is a large-scale graph datasets (not large-scale-graph). Potentially to include it. "

"not working well, overfitting, to rethink about it." (However, we would like to showcase the potential of SGT to integrate with efficiency designs for plain Transformers. Thus, "we construct an sparse version of SGT based on Exphormer (Shirzad et al., 2023) (i.e., the graph version of BigBird (Zaheer et al., 2020)) and evaluate it on the ogbn-arxiv dataset."

"attempt Exphormer-like SGT; overfitting severely leading to close to 99% train accuracy but bad test accuracy (55% compared to 72% in Exphormer)"

To reformulate the following.

Q6: Stability of Adaptive RMS Normalization: Did the authors encounter any stability or training sensitivity issues when introducing adaptive normalization (AdaRMSN) in comparison to LN/RMSN? Given that BatchNorm inherently preserves magnitude information but is known to be unstable, it would be valuable if the authors could provide a sensitivity analysis comparing AdaRMSN against these methods to highlight its relative stability and robustness.

We did not experience any stability or sensitivity issues during training with AdaRMSN in our experiments. Our initialization strategy ensures that AdaRMSN behaves similarly to regular RMSN during the initial training phase, thus guaranteeing stability. Although we observed more significant fluctuations in training loss when initializing AdaRMSN as an identity mapping (i.e., setting $\boldsymbol{\alpha} = \mathbf{1}$ and $\boldsymbol{\beta} = \mathbf{0}$), the overall training process maintained stability. Therefore, we propose our current initialization of AdaRMSN for further stability.

As mentioned by the reviewer, although BatchNorm (BN) inherently preserves magnitude information and has additional regularization ability, it is known for instability and sensitivity to the change of batch sizes (Wu & He, 2018), limiting its usage for training larger models and transferring features across tasks.

As suggested by the reviewer, we conduct a sensitivity study on ZINC comparing AdaRMSN, RMSN and BN. Due to the limit of time and long training time with small batch sizes, we shortened the training epochs from 2000 to 100 and removed all dropouts and droppaths. Following Wu & He (2018), we adjusted the learning rate $\mathrm{lr}' = \mathrm{lr} \cdot N/32$ for a batch size of N. Although this procedure means that the model is not optimally trained, the experiment does still allow us to gauge the stability and sensitivity w.r.t. batch sizes.

- Wu Y, He K. Group normalization. In ECCV 2018.

“To discuss whether we shall also adjust the tranining epochs instead of lr to balance the total training trials: to balance the total number of update instead of learning rate.”

“Attempted the variance across trials, not significant difference across BN, RMSN, AdaRMSn.”

“Attempted the large learning rates, all fail on proper training but not NaN occurs on all of BN, RMSN, AdaRMSn.”

Q7 “to add extra response to say our goal is not scalable graph Transformers. Rather it is powerful graph Transformers.”

1.2. Reviewer rMSz – 2

Pro The authors evaluate the proposed SGT on five traditional datasets and two large-scale datasets for graph classification. In addition to widely used graph benchmarks, they further assess the model on the BREC benchmark, which comprehensively measures the empirical expressive power of GNNs concerning graph isomorphism. The evaluation also includes an analysis of the theoretical WL-class of the models and their performance on the real-world ZINC dataset. Furthermore, an ablation study on ZINC is conducted to examine the contribution of each architectural modification introduced in this work.

Q1: While experimental results demonstrate that SGT achieves comparable performance to other GT methods, its superiority is not immediately evident.

The results in Table 1 and Table 2 do not clearly demonstrate the superiority of the proposed method.

We would argue that Table 1 and Table 2 provide clear evidence of superiority. Perhaps the reviewer can explain more clearly how this adverse conclusion is reached?

Based on Table 1 and Table 2, SGT outperforms all previous graph transformer methods, achieving state-of-the-art performance on 6 out of 7 benchmarks and consistently ranking within the top-3 across all datasets. This superior performance, achieved by a single architecture adhering to the plain transformers, is demonstrated against a comprehensive set of competitive baselines, including MPNNs, non-MPNN GNNs, and other graph transformers (GTs). SGT shows substantial performance improvements over all previous GTs, with the exception of GRIT, which SGT still surpasses in overall performance. We stress that a major goal of the research was to remove architectural complexities and return to a vanilla transformer architecture. So even minor outperformance of GRIT, which incorporates a complex attention mechanism, is notable.

Furthermore, as demonstrated in Table 3, SGT achieves superior empirical expressivity compared to other graph Transformers, providing compelling evidence that plain Transformers can possess strong graph learning capabilities, without requiring the complex architectural modifications employed in previous GT methods. In Table 3, we see that SGT can distinguish 24 of the CFI graph pairs compared to only 8 distinguished by GRIT and 10 distinguished by Graphormer.

Q2: In the experiments, they evaluate the proposed method on graph classification tasks. However, to provide a more comprehensive assessment, it would be beneficial to also evaluate its performance on node classification tasks.

Our evaluation does include node classification tasks: the CLUSTER and PATTERN benchmarks. As demonstrated in Table

1, SGT achieves state-of-the-art performance across these tasks, outperforming a wide range of competitive baselines that include MPNNs, non-MPNN GNNs, and other graph transformer architectures.

For further concerns regarding the applicability to large-scale graphs, we kindly direct reviewer rMSz to our **response to Q5 of Reviewer bTCe**, which provides an in-depth discussion on the distinction between "large-scale (graph) datasets" and "large-scale-graph datasets", as well as insights into foundation models and realistic graph settings.

Q3 :

Additionally, an analysis of training time and memory usage would further clarify the efficiency and scalability of the proposed approach.
Since the goal is to simplify Graph Transformers (GT), a comparison of training time and memory usage is necessary to highlight the efficiency gains.

We thank the valuable suggestions on a comparison of training time and memory usage. Plese find "to add the stat table".

We would like to emphasize that our architectural simplification focuses on maintaining the core principles of plain Transformers, **without hurting the theoretical expressivity and empirical performance on learning graphs**. This approach enables us to leverage the extensive training and architectural design advances explored for Transformers across various domains. Furthermore, it positions our method to seamlessly integrate with existing vision and language foundation models, offering significant potential for cross-domain applications and knowledge transfer.

While efficiency is valuable, our primary focus remains on maintaining plain Transformer principles without compromising theoretical expressivity or empirical performance.

1.3. Reviewer pJA6 - 3

I don't have much background of this area, but I accept the story and emprical results from the author, so I give the weak accept to the authors for now. I am willing to discuss with authors and reviewers' opinions in the rebuttal period.

Pro The paper is easy to follow, typically the titles of each subsection.

Q1 : Though there are a lot equations in this paper, but I beileve the underlying motivation is quite simple, if it is possible, maybe some case study or tutorial in the Appendix might be more clear to see the improvements. For example, if you want to emphysis the limitations of Token-wise normalization, give some vectors example , and intuiatively show the difference between you and current ones.

[Mark : This seems unnecessarily confrontational.] [liheng: Shall we delete it or tune it down?]

Our work's underlying motivation is straightforward. However, we contend that the analyses and findings on graph learning with plain Transformers are non-trivial, leading to our proposed simple yet effective modifications. SGT, adhering to the plain Transformer architecture, achieves outstanding performance on various real-world graph benchmarks (encompassing graph and node level tasks) as well as empirical expressivity evaluation, surpassing previous graph Transformers with more intricate designs.

Simple case studies/demonstration of sL_2 attention and sinusoidal-encoding have been provided in the Figure 2 and Figure 3, respectively.

"The case study of token-wise normalization is kindly provided as follows ..."

1.4. Reviwer sPMh - 2

Q1 Although the SGT structure is simplified, the complexity is still quadratic.

Indeed, the complexity remains quadratic. However, we contend that this does not undermine our work, as its primary focus is not on the efficiency and scalability in terms of input sizes. Rather, our work's primary emphasis lies in the capacity to model graph structures, with plain transformer architectures.

Our primary contribution is to demonstrate that, with our proposed designs, plain Transformer architectures can maintain strong capacity for learning graphs, as evidenced by our empirical experiments on various challenging graph benchmarks and empirical expressivity evaluations. The plain Transformer architectures enable leveraging the extensive training advances and architectural designs previously explored for Transformers in other domains. The architectural alignment also enables the potential integration with existing vision and language foundation models, unlocking substantial opportunities for cross-domain applications and facilitating effective knowledge transfer across different modalities.

Q2 In Table 2, the authors mistakenly write the name of SGT as GRITv2.

We sincerely thank the reviewer for catching this typo. We will correct this mistake in the final version of the paper.

Q3 This paper does not compare the Exphormer published in ICML 2023 and the GEAET published in ICML 2024.

Please refer to the following table comparing Exphormer and GEAET. We will include this comparison in the final version of the paper.

Q4 Could the authors provide a detailed complexity analysis of SGT?

“To include the complexity analysis”.

Please also find the runtime and memory usage comparison in **the response to Q3 of Reviewer rMSz**. It provides further information beyond the asymptotic complexity.

Q5 Can SGT be extended to large graphs? For example, could the authors provide results on datasets like CiteSeer, PubMed, or larger graphs such as ogbn-proteins, Amazon2m, etc.?

As highlighted in our response to Q1, our primary focus is not on improving efficiency or scalability for large-scale graphs. SGT is designed to maintain strong expressive power and learning capacity, which are essential for its performance across diverse graph learning tasks. It would be unreasonable to expect SGT to match the efficiency of models specifically optimized for scalability, as such models typically achieve their efficiency gains by compromising on model capacity and expressivity.

It is important to note that while these datasets contain large graphs, they are relatively small-scale in terms of the number of training examples (i.e., the number of graphs). The limited number of training samples, combined with the datasets’ inherent simplicity, makes them particularly susceptible to overfitting when using high-capacity models like Transformers.

Even though SGT will not be directly applicable to large-scale graphs, like most previous graph Transformers (Kreuzer et al., 2021; Ying et al., 2021; Rampasek et al., 2022; Zhang et al., 2023b; Ma et al., 2023). As a sanity check, we implement an efficient variant of SGT, adopting the expander-graph idea from Exphormer and evaluate it on ogbn-arxiv (with 169,343 nodes). “To double-check and think about the excuse Exp-SGT can easily reach 99% training accuracy and suffer from overfitting issues without extra regularization techniques. We will provide more details in the appendix in the final version of the paper.”

For a more in-depth discussion about ”large-scale graph datasets” and ”large-scale-graph datasets,” we refer the reviewer to the **response to Q5 of Reviewer bTCe**.

“Probably need to rearrange some answer to meet the 5000-word limits.”

References

Rampášek, L., Galkin, M., Dwivedi, V. P., Luu, A. T., Wolf, G., and Beaini, D. Recipe for a General, Powerful, Scalable Graph Transformer. In *Adv. Neural Inf. Process. Syst.*, 2022.