AnchorGT: Efficient and Flexible Attention Architecture for Scalable Graph Transformers

Wenhao Zhu[1] , **Guojie Song**[*1] , **Liang Wang**[2] and **Shaoguo Liu**[2]

[1]National Key Laboratory of General Artificial Intelligence, School of Intelligence Science and Technology, Peking University

[2]Alibaba Group

{wenhaozhu, gjsong}@pku.edu.cn, {liangbo.wl, shaoguo.lsg}@alibaba-inc.com

Abstract

Graph Transformers (GTs) have significantly advanced the field of graph representation learning by overcoming the limitations of message-passing graph neural networks (GNNs) and demonstrating promising performance and expressive power. However, the quadratic complexity of self-attention mechanism in GTs has limited their scalability, and previous approaches to address this issue often suffer from expressiveness degradation or lack of versatility. To address this issue, we propose AnchorGT, a novel attention architecture for GTs with global receptive field and almost linear complexity, which serves as a flexible building block to improve the scalability of a wide range of GT models. Inspired by anchor-based GNNs, we employ structurally important k-dominating node set as anchors and design an attention mechanism that focuses on the relationship between individual nodes and anchors, while retaining the global receptive field for all nodes. With its intuitive design, AnchorGT can easily replace the attention module in various GT models with different network architectures and structural encodings, resulting in reduced computational overhead without sacrificing performance. In addition, we theoretically prove that AnchorGT attention can be strictly more expressive than Weisfeiler-Lehman test, showing its superiority in representing graph structures. Our experiments on three state-of-the-art GT models demonstrate that their AnchorGT variants can achieve better results while being faster and significantly more memory efficient.

1 Introduction

Transformer[1] [Vaswani *et al.*, 2017] has become the dominant universal neural architecture for natural language processing and computer vision due to its powerful self-attention mechanism. Its success has sparked interest in adapting Transformer for use in graph machine learning [Ying *et al.*, 2021;

---

*Corresponding Author

[1]Please refer to https://arxiv.org/abs/2405.03481 for an extended version of this paper.

Rampášek *et al.*, 2022]. While Graph Neural Networks (GNNs) have limitations such as over-smoothing and neighbor explosion in the message-passing paradigm, the promising performance of Transformer-based approaches in graph machine learning has led researchers to investigate their potential use in a wider range of scenarios.

However, the application of Transformer in large-scale graph machine learning is often limited by the $O(N^2)$ complexity of the self-attention mechanism, where N is the number of nodes in the input graph. To address this computational bottleneck, previous approaches have typically employed two strategies: restricting the receptive field of nodes through techniques such as sampling [Zhang *et al.*, 2022], or applying linear attention methods directly to graph Transformer [Rampášek *et al.*, 2022]. However, both of these approaches have inherent flaws. The first strategy sacrifices the key element of self-attention - global receptive field - which reduces the model's ability to capture global graph structure and may ignore graph structural information with simple sampling methods. The second approach, while maintaining the global attention mechanism, is incompatible with the common relative structural encoding in graph Transformer (as the approximate step bypasses computation of the full attention matrix, will be discussed later), significantly reducing the model's ability to learn graph structure. As a result, current approaches to improving the scalability of graph Transformer sacrifice global receptive field or structural expressivity to some extent and still require improvement.

To address these issues, we draw inspiration from anchor-based graph neural networks [You *et al.*, 2019] and propose AnchorGT, a novel attention architecture for graph Transformers with almost linear complexity, high flexibility, and strong expressive power. Previous anchor-based graph neural networks often use a randomly sampled set of anchor nodes as a tool for GNNs to capture the global relative positions of nodes in a graph. In our approach, we select a set of k-dominating set nodes, which are nodes that can be quickly computed and are structurally important, as the anchor set to support the learning and propagation of broad-area information in the graph structure. Based on these anchors, our redesigned attention mechanism allows each node to attend to both its neighbors and the anchor nodes in the global context, effectively reducing computational complexity while retaining the fundamental characteristics of the attention mech-

anism. With the k-dominating set anchors and the new attention mechanism, AnchorGT layer achieves almost linear complexity, has global receptive field for each node, and is compatible with many structural encodings and graph Transformer methods. Moreover, we theoretically prove that the AnchorGT layer with structural encoding that satisfies certain conditions is strictly more expressive than graph neural network based on Weisfeiler-Lehman test, further demonstrating the superiority of our method.

In experiments, we replaced the attention mechanism module of three state-of-the-art graph transformer methods with anchor-based attention and tested their performance and scalability on both graph-level and node-level datasets. The results show that the AnchorGT variants of these methods have significantly reduced memory consumption and training time while maintaining excellent performance, proving that AnchorGT can effectively improve the scalability of graph Transformers without sacrificing performance. We summarize the contribution of this paper as follows:

- We propose AnchorGT, a novel attention architecture for graph Transformers as a flexible building block to improve the scalability of a wide range of graph Transformer models.

- In AnchorGT, we propose a novel approach of using k-dominating set anchors to efficiently propagate global graph information, and design a new attention mechanism that combines local and global information based on the anchor method. AnchorGT has almost linear complexity and global receptive field, and can be integrated with a wide range of existing graph Transformer methods.

- Theoretically, we prove that AnchorGT layer can be strictly more expressive than WL-GNNs with certain structural encoding.

- In experiments, the AnchorGT variants of three state-of-the-art graph Transformer models achieved competitive performance on both graph-level and node-level tasks, with significantly improved scalability.

2 Related Work

2.1 Graph Transformer

Along with the recent surge of Transformer, many prior works have attempted to bring Transformer architecture to the graph domain, including GT [Dwivedi and Bresson, 2020], GROVER [Rong *et al.*, 2020], Graphormer [Ying *et al.*, 2021], SAN [Kreuzer *et al.*, 2021], SAT [Chen *et al.*, 2022], ANS-GT [Zhang *et al.*, 2022], GraphGPS [Rampášek *et al.*, 2022], GRPE [Park *et al.*, 2022], EGT [Hussain *et al.*, 2022], NodeFormer [Wu *et al.*,], GOAT [Kong *et al.*, 2023]. [Müller *et al.*, 2023] is an complete survery of graph Transformers. GT [Dwivedi and Bresson, 2020] provides a generalization of Transformer architecture for graphs with modifications like using Laplacian eigenvectors as positional encodings and adding edge feature representation to the model. GROVER [Rong *et al.*, 2020] is a molecular large-scale pretrain model that applies Transformer to node embeddings cal-

culated by GNN layers. Graphormer [Ying *et al.*, 2021] proposes an enhanced Transformer with centrality, spatial and edge encodings, and achieves state-of-the-art performance on many molecular graph representation learning benchmarks. SAN [Kreuzer *et al.*, 2021] presents a learned positional encoding that cooperates with full Laplacian spectrum to learn the position of each node in the graph. Gophormer [Zhao *et al.*, 2021] applies structural-enhanced Transformer to sampled ego-graphs to improve node classification performance and scalability. SAT [Chen *et al.*, 2022] studies the question of how to encode structural information to Transformers and proposes the Structure-Aware-Transformer to generate position-aware information for graph data. ANS-GT [Zhang *et al.*, 2022] proposes an adaptive sampling strategy to effectively scale up graph Transformer to large graphs. GraphGPS [Rampášek *et al.*, 2022] proposes a general, powerful and scalable architecture for building graph Transformers and use the fast attention method to improve scalability. GOAT [Kong *et al.*, 2023] proposes a global graph Transformer architecture with excellent performance and efficiency. NodeFormer [Wu *et al.*,] introduces a novel all-pair message passing scheme for efficiently propagating node signals between arbitrary nodes based on the Gumbel-Softmax operator.

2.2 Anchor-based Graph Neural Network

Anchor-based Graph Neural Network is first proposed by P-GNN [You *et al.*, 2019]. To generate position-aware node embeddings that captures the positional information of nodes within the global context, P-GNN proposes to incorporate distance information from the target node to a set of randomly sampled anchor nodes in the whole graph during the representation generation process. Following P-GNN, AS-GNN [Dong *et al.*, 2021] proposes the MVC algorithm, utilizing complex network theory, to select structurally important nodes based on subgraph partitioning as anchor nodes to enhance the performance of GNN representations of graph structure, instead of random selection. IDGL [Chen *et al.*, 2020b] proposes an graph learning framework for jointly and iteratively learning graph structure and graph embedding with anchor-based approximation technique to reduce computational complexity. A-GNN [Liu *et al.*, 2020] selects anchors through greedy algorithm with the minimum point cover algorithm, and learns a non-linear aggregation scheme to incorporate the anchor-based structural information into message-passing scheme. Previous anchor-based graph neural network methods have explored various ways of incorporating anchor information to improve graph neural networks, and our work is the first to apply anchor-based methods to graph Transformer, resulting in a efficient and flexible new graph Transformer framework.

3 Proposed Approach

In this section, we will describe the overall architecture of the model, including definition of anchor sets and the anchor-based attention module.

Notations Let $G = (V, E)$ denote a graph, where $V = \{v_1, v_2, \ldots, v_n\}$ is the node set that consists of n vertices and $E \subset V \times V$ is the edge set. For $v, u \in V$, let $\text{SPD}(v, u)$ be
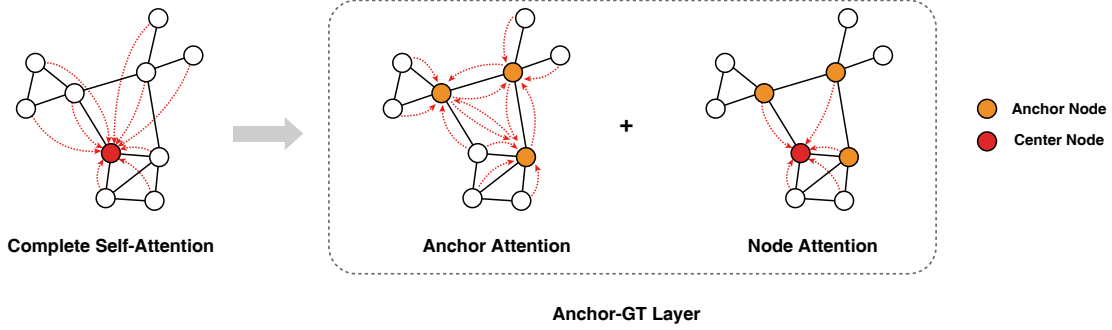
Figure 1: An illustration of the proposed AnchorGT.

the shortest path distance between v and u in G. For node $v \in V$, let $N(v) = \{v' : v' \in V, (v, v') \in E\}$ denote the set of its neighbors, and let $N_k(v) = \{u : u \in V, \text{SPD}(v, u) \leq k\}$ be the $k-$hop neighborhood of v. Let each node v_i be associated with a feature vector $\boldsymbol{x}_i \in \mathbb{R}^F$ where F is the hidden dimension, and let $\mathbf{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n]^\top \in \mathbb{R}^{n \times F}$ denote the feature matrix.

3.1 $k-$Dominating Set Anchor

In graphs, nodes and edges often possess unique structural positions, where some nodes occupy central positions in the overall structure (such as hub nodes in social network graphs [Wasserman *et al.*, 1994] and key carbon atoms in molecular graphs), while others are situated in peripheral locations. Studies on complex network dynamics [Chen *et al.*, 2012] have also demonstrated that certain key nodes and edges in real-world graphs often possess higher information propagation probabilities than other nodes, and these nodes generally contain more significant global structural information. Therefore, we can select a subset of nodes that possess key structural information as anchors, making them an efficient medium for conveying global information in the graph.

In the proposed AnchorGT, we choose the $k-$dominating set ($k-$DS) as the anchor set, as this type of anchor set can be calculated with approximate linear complexity and has structural significance. For a $k-$DS anchor set S, any node $v \in V$ is an anchor node or there exists an anchor $u \in S$ such that $\text{SPD}(v, s) \leq k$. Formally, the $k-$dominating set is a classical concept in graph theory, defined as follows:

Definition 3.1 ($k-$dominating set). *For graph $G = (V, E)$, its $k-$dominating set S is a subset of V such that for any $v \in V$, there exists $u \in S$ satisfying $SPD(v, u) \leq k$.*

In order to compute the $k-$dominating set, we follow previous studies [Nguyen *et al.*, 2020] and adopt an approximate linear complexity greedy algorithm. The specific steps of the algorithm are described in Algorithm 1 in appendix. This algorithm first assigns labels to all nodes in the graph, and repeatedly selects the labeled node with the highest degree from the graph and adds it to the anchor set, while removing labels from its $k-$hop neighborhood, until all nodes are unlabeled. The complexity of degree sorting is $O(N \log N)$. The algorithm performs at most N iterations with at most n_k queries

of the adjacency matrix in each iteration, where n_k is the maximum number of $k-$hop neighborhood nodes. Therefore, the overall complexity of the algorithm is $O(N(\log N + n_k))$, which is an approximate linear complexity with respect to N since $\log N \ll N$ and $n_k \ll N$. In the experiment, the time required to compute the 2-DS of the ogbn-products dataset with over 3 million nodes is less than 20 minutes, which is negligible compared to the training time of the network and demonstrates that the computational complexity of this part is completely acceptable. The $k-$dominating set S possesses a key property - the union of the $k-$hop neighborhoods of all nodes in the set results in the complete node set V, which will be crucial for our model to achieve global perception.

3.2 Anchor-based Attention

In the following, we will describe the proposed anchor-based attention mechanism. In this method, each node will perform attention calculation on both its $k-$hop neighbors and all anchor nodes, to achieve the goal of reducing complexity and maintaining global receptive field. Specifically, for every $v \in V$ with representation $\boldsymbol{h}_v$, its AnchorGT self-attention computation follows the following scheme:

$$\boldsymbol{q}_v = \boldsymbol{P}_Q(\boldsymbol{h}_v), \tag{1}$$

$$\text{for } u \in R(v): \tag{2}$$

$$\boldsymbol{k}_u = \boldsymbol{P}_K(\boldsymbol{h}_u), \boldsymbol{v}_u = \boldsymbol{P}_V(\boldsymbol{h}_u), \tag{3}$$

$$\alpha_{v,u} = \text{softmax}_{u \in R(v)}\left(\frac{\boldsymbol{q}_v^\top \boldsymbol{k}_u}{\sqrt{d}} + \boldsymbol{F}(\text{SE}(v, u))\right), \tag{4}$$

where $\boldsymbol{P}_Q, \boldsymbol{P}_K, \boldsymbol{P}_V$ are learnable linear projection functions for query, key, value vectors, $\boldsymbol{F}$ is the transform function on structural encodings (e.g. table embedding or linear transform), and $R(v)$ is the *receptive field* of v which we will use to control the scope of attention mechanism. $\text{SE}(v, u)$ is the *structural encoding* of node pair (v, u) to inject structural information into self-attention. The output representation is

$$\boldsymbol{h}_v = \sum_{u \in R(v)} \alpha_{v,u} \boldsymbol{v}_u. \tag{5}$$

Then in anchor-based attention, with $k-$DS anchor set S, the receptive field of nodes is defined as

$$R(v) = N_k(v) \cup S. \tag{6}$$

The above definition constitutes our proposed Anchor-based attention mechanism in the AnchorGT framework, which is independent of specific structural encoding and can easily be extended to multi-head attention scenarios. It can be seamlessly incorporated into various graph Transformer networks to replace the original attention mechanism. As all the $k-$hop neighbors of the anchor points cover the entire node set, we can conclude that after more than one rounds of attention computation, every node has a global receptive field. Moreover, assuming A is the size of the anchor set S, we can easily infer that the computational complexity of the model is $O(N(n_k + A))$. Since $n_k \ll N$ and $A \ll N$ (which we will prove by numbers in Section 5.4), the complexity of Anchor-based attention is also nearly linear. As a result, our proposed model is theoretically more scalable than previous methods with $O(N^2)$ attention mechanism, and it has been shown by experiments to result in significantly lower GPU memory usage and less training time. We primarily utilized the `torch.sparse.sampled_addmm` function from PyTorch package to implement the proposed anchor-based attention module.

Structural Encoding The vanilla self-attention mechanism is ignorant of the graph structure, so it is necessary to incorporate structural encoding to learn graph structural information. Common methods include absolute structural encoding (including node degree [Ying *et al.*, 2021], Laplacian eigenvectors [Kreuzer *et al.*, 2021]) and relative structural encoding (including shortest path length and transition probability [Rampášek *et al.*, 2022]). Our model is not dependent on a specific structural encoding definition. In experiments, we adopt the shortest path structural (SPD) encoding [Ying *et al.*, 2021], which is simple and effective, and can be calculated in linear time. In subsequent theoretical analysis, we will also prove that this encoding makes the model have stronger expressive power than graph neural networks.

Anchor-based Attention with Sampling-based Graph Transformers In the previous paragraphs, we describe the anchor-based attention mechanism in graph Transformer models based on the entire graph. This type of model typically processes multiple small-scale graphs (such as molecular graphs) in batches and predicts node or graph labels, like Graphormer [Ying *et al.*, 2021] and GraphGPS [Rampášek *et al.*, 2022] described in the original papers. However, large-scale graph data exists in the real world, which generally involves node-level or edge-level tasks and is mostly trained using sampling-based training methods, like ANS-GT [Zhang *et al.*, 2022]. In sampling-based training, each training batch samples smaller sub-graph that can be accommodated by the GPU memory from the global large graph, and then the model calculates the node or edge representations on the sub-graph. We can see that our method is also applicable to this type of graph Transformer. Our model only needs to pre-calculate the anchor set of the entire graph, then add the subset of the graph anchor set in the sampled sub-graph to each batch during training.

4 Expressiveness of AnchorGT

Previous results have demonstrated that graph Transformers may possess stronger expressive power than GNNs, and our model can be considered as a simplified one with randomness. Therefore, a natural question arises: can we provide similar theoretical guarantees for the proposed AnchorGT model? In the following section, we will give a affirmative answer to this question - as long as structural encodings that satisfy certain conditions (such as SPD) are utilized, our model can have strictly stronger expressive power than GNNs. In this section, we consider a simple AnchorGT model built by stacked Transformer layers with AnchorGT attention and a global readout function (like mean in GNNs).

Considering that the graph structure information in graph Transformer models is entirely provided by the structural encoding function, we must first impose certain requirements on these structural encodings to ensure a lower bound on the model's expressive power. We first define the **neighbor-distinguishable** structural encoding:

Definition 4.1 (Neighbor-Distinguishable Structural Encoding). *In $G = (V, E)$, a relative structural encoding function $SE(\cdot, \cdot) \in V \times V \mapsto \mathbb{R}^d$ is neighbor-distinguishable if a mapping $f : \mathbb{R}^d \mapsto \{0, 1\}$ exists such that for $v \in V$ and $u \in R(v)$, $f(SE(v, u)) = 1$ if $(v, u) \in E$, and $f(SE(v, u)) = 0$ if $(v, u) \notin E$.*

For example, the shortest-path-distance structural encoding is neighbor-disguishable since two nodes are adjacent if their SPD is 1. Using the similar method previously outlined in the literature [Ying *et al.*, 2021] to simplify graph Transformer models into GNN models, we can easily prove the following conclusion:

Fact 1. *For any two graphs G_1 and G_2 that can be distinguished by a message-passing GNN model, there exists a set of model parameters such that the AnchorGT model with neighbor-distinguishable encoding can distinguish them.*

Noting that the aggregation function and global readout function in GNNs can both be expressed by graph Transformer, the proof of the above conclusion is intuitive. We provide a detailed proof in the appendix. The above conclusion demonstrates that as long as there is a neighbor-distinguishable encoding, the AnchorGT model's expressive power is not weaker than GNNs. However, what we are most concerned with is: *Does AnchorGT allow the graph Transformer to surpass the expressiveness of GNNs?* To answer this question, we first provide the definition of *anchor-distinguishable* structural encoding:

Definition 4.2 (Anchor-Distinguishable Structural Encoding). *In $G = (V, E)$ with $k-DS$ anchor $S \subset V$, a relative structural encoding function $SE(\cdot, \cdot) \in V \times V \mapsto \mathbb{R}^d$ is anchor-distinguishable if a mapping $f : \mathbb{R}^d \mapsto \{0, 1\}$ exists such that for $v \in V$ and $u \in R(v)$, $f(SE(v, u)) = 1$ if $u \in S$ and $u \notin N_k(v)$, and $f(SE(v, u)) = 0$ otherwise.*

Anchor-distinguishable structural encodings allow the model to access structural information from anchors outside the neighborhood, which is crucial for expressive power. *The SPD encoding is anchor-distinguishable* for any k, because the SPD encoding only produces results $\geq k$ for anchor nodes

outside of k-neighbors. We prove the following result (detailed proof in appendix):

Fact 2. *There exist graphs G_1 and G_2 that can not be distinguished by message-passing GNNs, but can be distinguished by AnchorGT model with $1-DS$ anchors and structural encoding that is both neighbor-distinguishable and anchor-distinguishable.*

The above conclusion theoretically proves that as long as the structural encoding is neighbor-and-anchor-distinguishable (like SPD we use), our AnchorGT model can have strictly stronger expressive power than GNNs; this expressive power comes from the combination of the anchor method and strong structural encoding. Interestingly, under the SPD structural encoding, AnchorGT model can be seen as a simplified version of Graphormer, but can still distinguish G_1 and G_2 (Figure 4 in Appendix) which serve as an example to prove that Graphormer has an expressive ability superior to GNNs [Ying *et al.*, 2021]. Therefore, the AnchorGT method allows attention mechanism to maintain low complexity while maintaining the expressive power advantage of Graph Transformer compared to GNNs.

5 Experiments

In experiments, we test the proposed AnchorGT method on three graph Transformer models and both graph-level and node-level benchmarks. We show that the AnchorGT variants of graph Transformer models can achieve the same performance level as the original models while being significantly more efficient.

5.1 Datasets and Experimental Settings

In graph representation learning experiments, we select 7 datasets: ogb-PCQM4Mv2 from ogb-lsc [Hu *et al.*, 2020; Hu *et al.*, 2021], QM9 and QM8 from MoleculeNet [Wu *et al.*, 2018], LRGB-Peptides-struct and LRGB-Peptides-func from LRGB datasets [Dwivedi *et al.*, 2022], and CLUSTER [Dwivedi *et al.*, 2020]. In node representation learning experiments, we select 4 datasets: Citeseer, Pubmed [Kipf and Welling, 2016], ogbn-arxiv and ogbn-products [Hu *et al.*, 2020]. For datasets with pre-defined splits (ogb datasets, LRGB datasets, CLUSTER), we adopt the pre-defined splits. For datasets without pre-defined splits, we adopt a 8:1:1 train:validation:test random split for graph-level datasets (QM9 and QM8) and a 2:2:6 random split for node-level datasets (Citeseer and Pubmed). For QM9 and QM8, We follow the guidelines in MoleculeNet [Wu *et al.*, 2018] for choosing regression tasks and metrics. We perform joint training on 12 tasks for QM9 and 16 tasks for QM8.

We compare our method against (1) GNNs include GCN [Kipf and Welling, 2016], GCNII [Chen *et al.*, 2020a], GAT [Veličković *et al.*, 2017], GIN (GINE) [Xu *et al.*, 2018; Hu *et al.*, 2019], GatedGCN [Bresson and Laurent, 2017] and APPNP [Klicpera *et al.*, 2018]; (2) graph Transformers include GT [Dwivedi and Bresson, 2020], SAN [Kreuzer *et al.*, 2021], Gophormer [Zhao *et al.*, 2021], SAT [Chen *et al.*, 2022], GraphGPS [Rampášek *et al.*, 2022] and ANS-GT [Zhang *et al.*, 2022]. Most of the baseline results are cited from their original pepers, except for the results on QM9 and

QM8 dataset and results of SAT on LRGB datasets. The more detailed settings for baselines are listed in the appendix. All our experiments are performed on one NVIDIA RTX 4090 GPU.

5.2 AnchorGT Achieves Full Attention Performance

First, we analyze how the AnchorGT attention affects the performances of graph Transformer models on both graph-level and node-level benchmarks. We select two important graph Transformer models for graph representation learning: Graphormer [Ying *et al.*, 2021] and GraphGPS [Rampášek *et al.*, 2022], and build their AnchorGT variants. Graphormer generally improves the basic graph Transformer with three kinds of structural encodings: centrality encoding, spatial encoding and edge encoding. Fortunately, the above three types of encodings can be realized within the framework of our anchor-based attention mechanism, so we directly constructed Graphormer-AnchorGT as the AnchorGT variant of Graphormer. Meanwhile, the GraphGPS architecture includes structural encoding, a local message passing module implemented by GNN, and a global attention mechanism module implemented by Transformer network. The global attention module adopts the standard Transformer structure or other linear Transformers, thus we can easily replace it with the anchor-based attention mechanism to construct GraphGPS-AnchorGT, the basic AnchorGT variant of GraphGPS. Additionally, considering the lack of relative structural encoding in Transformer networks of GraphGPS, we construct GraphGPS-AnchorGT-SPD as an enhanced version of GraphGPS-AnchorGT with SPD relative structural encoding to study the effect of structural encodings. For node-level tasks, we selected ANS-GT [Zhang *et al.*, 2022], which achieves state-of-the-art and enhances the performance of graph Transformers in node-level tasks through novel sampling and attention methods. The Transformer component of ANS-GT is similar to that of Graphormer, with only a few specific requirements (we need to set the center node of each sampling as the anchor node). Therefore, we built the ANS-GT-AnchorGT model based on the previous approach. For all models, we adopt configurations specified in the original papers, and conduct a hyperparameter search on some training parameters (e.g., learning rate, batch size) to obtain the optimal configuration of AnchorGT models. We set $k = 2$ for all AnchorGT models..

We summarize the results in Table 1, 2 and 4. For Graphormer and GraphGPS, the inclusion of anchor-based attention results in generally improved performance on some datasets. And for GraphGPS, SPD structure encoding improves the model's structural expressivity and boosts performance, resulting in slightly better performance on five datasets for the GraphGPS-AnchorGT-SPD model compared to the GraphGPS. This corresponds to our theoretical analysis in Section 4, that the expressive power of graph Transformer blocks can be improved by adding structural encodings, and AnchorGT layers with SPD are theoretically more powerful than GNN and naive Transformer layers. And for node-level tasks with ANS-GT, AncorGT also results in performance improvements, which can be attributed to the fact

Dataset	ZINC	QM9	QM8	LRGB-Peptides-struct	LRGB-Peptides-func
Metric	MAE↓	Multi-MAE↓	Multi-MAE↓	MAE↓	AP↑
GCN	0.1379	1.006±0.020	0.0279±0.0001	0.3496±0.0013	0.5930±0.0023
GCNII	-	0.945±0.462	0.0256±0.0002	0.3471±0.0010	0.5543±0.0078
GAT	-	1.112±0.018	0.0317±0.0001	-	-
GIN (GINE)	0.1218	1.225±0.055	0.0276±0.0001	0.3547±0.0045	0.5498±0.0079
GatedGCN	-	-		0.3420±0.0013	0.5864±0.0077
GT	0.226±0.01	-	-	-	-
SAT	-	1.122±0.135	0.0281±0.0025	0.6473±0.0849	0.2601±0.0248
SAN (LapPE)	0.139±0.006	-	-	0.2683±0.0043	0.6384±0.0121
Graphormer	**0.120**±0.008	0.720±0.035	0.0079±0.0001	0.2705±0.0045	0.6453±0.0121
Graphormer-AnchorGT-SPD	0.122±0.006	**0.702**±0.051	**0.0077**±0.0001	**0.2519**±0.0033	**0.6520**±0.0264
Δ	−1.66%	+2.56%	+2.53%	+7.38%	+1.04%

Table 1: Results of Graphormer and its variant Graphormer-AnchorGT on graph representation learning benchmarks.

Dataset	ogb-PCQM4Mv2	QM9	LRGB-Peptides-struct	LRGB-Peptides-func	CLUSTER
Metric	Validation MAE↓	MAE↓	MAE↓	AP↑	Accuracy↑
GCN	0.1379	1.006±0.020	0.3496±0.0013	0.5930±0.0023	68.498±0.976
GCNII	-	0.945±0.462	0.3471±0.0010	0.5543±0.0078	-
GAT	-	1.112±0.018	-	-	70.587±0.447
GIN (GINE)	0.1218	1.225±0.055	0.3547±0.0045	0.5498±0.0079	64.716±1.553
GatedGCN	-	-	0.3420±0.0013	0.5864±0.0077	73.840±0.326
SAT	-	1.122±0.135	0.6473±0.0849	0.2601±0.0248	77.856±0.104
SAN (LapPE)	-	-	0.2683±0.0043	0.6384±0.0121	76.691±0.65
Graphormer	0.0864	0.729±0.044	0.2705±0.0045	0.6453±0.0121	77.552±0.546
GraphGPS	0.0878	*0.705*±0.081	0.2500±0.0005	**0.6535**±0.0041	78.016±0.180
GraphGPS-AnchorGT	0.0897	0.712±0.074	0.2674±0.0011	0.6214±0.0056	*78.123*±0.365
GraphGPS-AnchorGT-SPD	**0.0862**	**0.665**±0.086	**0.2437**±0.0006	*0.6460*±0.0031	**78.802**±0.252
Δ	1.86%	+5.6%	+2.58%	−1.15%	+1.01%

Table 2: Results of GraphGPS and its AnchorGT variants on graph representation learning benchmarks.

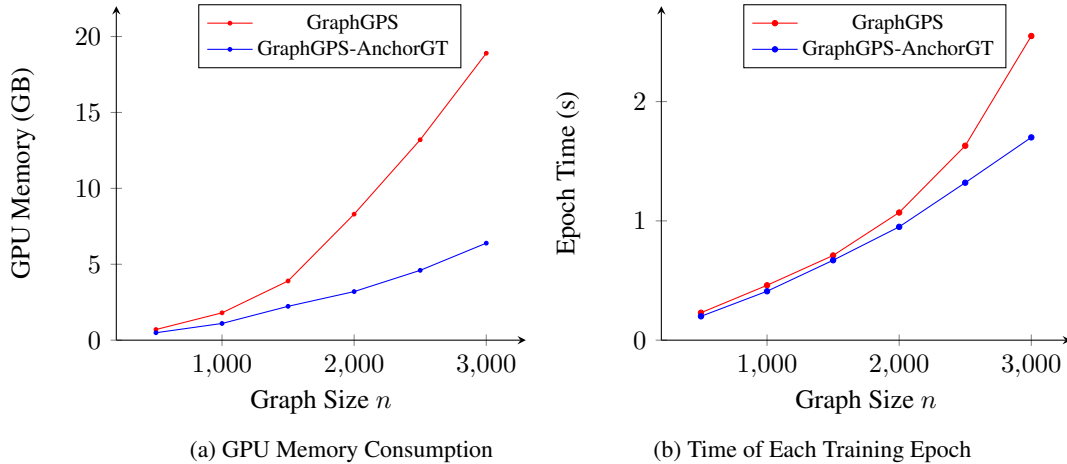(a) GPU Memory Consumption

(b) Time of Each Training Epoch

Figure 2: GPU memory consumption and training epoch time of synthetic efficiency tests on GraphGPS and its AnchorGT variant.

that anchor-based attention can reduce long-range noises in large graphs.

5.3 AnchorGT is Fast and Memory Efficient

Next, we investigate AnchorGT's ability to improve model efficiency, in terms of both GPU memory utilization during training and time of each training epoch. To run a standarized benchmark and avoid structural noises in real-world

Dataset	N	E	avg. degree	$|A|(k=1)$	$|A|(k=2)$	$|A|(k=3)$	$|A|(k=4)$
QM9	2449029	121318152	25.2	OOM	80970	60079	50436
ogbn-products	18.03	18.66	1.03	6.8	3.5	2.25	1.55

Table 3: Statistics and number of anchors under different k values of qm9 and ogbn-products dataset.

Dataset	Citeseer	ogbn-arxiv	ogbn-products
Metric		Accuracy↑	
GCN	79.43±0.26	71.24±0.29	75.64±0.21
GAT	80.13±0.62	57.88±0.18	79.45±0.59
GraphSAGE	79.23±0.53	71.49±0.27	78.50±0.14
APPNP	79.33±0.35	-	-
SAN	70.64±0.97	-	-
Gophormer	76.43±0.78	-	-
ANS-GT	**80.25**±0.39	71.86±0.20	**80.50**±0.78
ANS-GT-AnchorGT	82.01±0.21	**72.34**±0.45	81.04±0.52
Δ	+2.19%	+0.67%	+0.67%

Table 4: Results of node representation learning benchmarks.

datasets, we construct a synthetic random graph dataset to characterize the efficiency of the model. We generate a training dataset consisting of 100 Erdos-Renyi random graphs, where the probability of an edge is 0.0001 (approximately reflecting the sparsity density of real-world graph datasets), and the number of nodes in each graph is controlled by a parameter n. A GraphGPS model consisting of only an input feature encoder and Transformer units with a hidden layer dimension of 128 and 5 Transformer blocks with 4 attention heads is constructed as the baseline. The batch size is set to 10. The anchorGT version of the model is kept consistent and k is set to 2. After warm-up, we measure the peak memory utilization using the built-in `get_summary` function of `pytorch.cuda` and calculate the time of each training epoch.

We plot the results in Figure 2. As shown in the figure, the standard Transformer model's memory consumption exhibits a quadratic increase with the increasing number of graph size, limiting the scalability of the model. However, our AnchorGT model's memory consumption exhibits near-linear growth, maintaining low memory consumption as the graph size increases, reducing memory consumption by about 60% compared to the baseline model. Therefore, our approach of saving memory consumption without sacrificing performance is meaningful for graph Transformer models. For training time, due to the extra computation introduced by the anchor method, AnchorGT saves around $10-30\%$ of the time compared to the original model. In summary, this synthetic test well demonstrates that AnchorGT is fast and memory efficient.

5.4 Effect of k

Finally, we study a key parameter k in the model, which determines the size of the neighborhood covered by the anchor points. Generally speaking, the larger k is, the larger the coverage range of each anchor point, and the fewer the total num-

ber of anchor points. Specifically, the theoretical complexity of the model is $O(N(n_k + A))$, where n_k increases with the increase of k, but A decreases. Thus, the influence of parameter k on the computational complexity and performance of the model is complex. In Table 3, we present the number of anchors for the QM9 and ogbn-products datasets, and in Figure 3 we plot the relative memory usage and performance of the GraphGPS-AnchorGT-SPD model on the QM9 dataset under different k values. As depicted in Figure 3 [2], the value of k has an impact on both the performance and complexity of the model, and the model exhibits the best performance and efficiency when $k = 2$. In general, we found the following conclusions in experiments: $k = 1$ is not a good choice, which will lead to too many anchor points, increase the complexity of the model, and reduce performance; when $k \geq 3$, the $k-$hop neighborhood of each node is also too large and this will hurt performance and efficiency. So in general, taking $k = 2$ can achieve good performance and complexity at most circumstances.
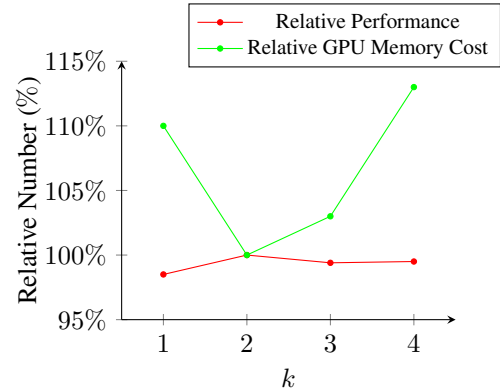

Figure 3: Relative Performance and Memory Cost (%) of GraphGPS-AnchorGT-SPD on qm9 dataset. The settings stay the same with Table 2.

6 Conclusion

In our study, we have presented AnchorGT, a highly efficient and adaptable attention architecture for graph Transformers. AnchorGT's versatility enables it to seamlessly integrate with a variety of graph Transformers, thereby significantly enhancing the scalability of the model, all while maintaining optimal performance levels.

---

[2]Note that ogbn-products is a singlue large graph and qm9 consists of 130k small graphs, so the overall $N, E, |A|$ are reported for ogbn-products and the average $N, E, |A|$ of all graphs are reported for qm9. OOM stands for out of memory when calculating.

Acknowledgements

References

[Bresson and Laurent, 2017] Xavier Bresson and Thomas Laurent. Residual gated graph convnets. *arXiv preprint arXiv:1711.07553*, 2017.

[Chen *et al.*, 2012] Duanbing Chen, Linyuan Lü, Ming-Sheng Shang, Yi-Cheng Zhang, and Tao Zhou. Identifying influential nodes in complex networks. *Physica a: Statistical mechanics and its applications*, 391(4):1777–1787, 2012.

[Chen *et al.*, 2020a] Ming Chen, Zhewei Wei, Zengfeng Huang, Bolin Ding, and Yaliang Li. Simple and deep graph convolutional networks. In *International conference on machine learning*, pages 1725–1735. PMLR, 2020.

[Chen *et al.*, 2020b] Yu Chen, Lingfei Wu, and Mohammed Zaki. Iterative deep graph learning for graph neural networks: Better and robust node embeddings. *Advances in neural information processing systems*, 33:19314–19326, 2020.

[Chen *et al.*, 2022] Dexiong Chen, Leslie O'Bray, and Karsten Borgwardt. Structure-aware transformer for graph representation learning. In *International Conference on Machine Learning*, pages 3469–3489. PMLR, 2022.

[Dong *et al.*, 2021] Lijun Dong, Hong Yao, Dan Li, Yi Wang, Shengwen Li, and Qingzhong Liang. Improving graph neural network via complex-network-based anchor structure. *Knowledge-Based Systems*, 233:107528, 2021.

[Dwivedi and Bresson, 2020] Vijay Prakash Dwivedi and Xavier Bresson. A generalization of transformer networks to graphs. *arXiv preprint arXiv:2012.09699*, 2020.

[Dwivedi *et al.*, 2020] Vijay Prakash Dwivedi, Chaitanya K Joshi, Thomas Laurent, Yoshua Bengio, and Xavier Bresson. Benchmarking graph neural networks. *arXiv preprint arXiv:2003.00982*, 2020.

[Dwivedi *et al.*, 2022] Vijay Prakash Dwivedi, Ladislav Rampášek, Mikhail Galkin, Ali Parviz, Guy Wolf, Anh Tuan Luu, and Dominique Beaini. Long range graph benchmark. *arXiv preprint arXiv:2206.08164*, 2022.

[Hu *et al.*, 2019] Weihua Hu, Bowen Liu, Joseph Gomes, Marinka Zitnik, Percy Liang, Vijay Pande, and Jure Leskovec. Strategies for pre-training graph neural networks. *arXiv preprint arXiv:1905.12265*, 2019.

[Hu *et al.*, 2020] 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. *arXiv preprint arXiv:2005.00687*, 2020.

[Hu *et al.*, 2021] Weihua Hu, Matthias Fey, Hongyu Ren, Maho Nakata, Yuxiao Dong, and Jure Leskovec. Ogb-lsc: A large-scale challenge for machine learning on graphs. *arXiv preprint arXiv:2103.09430*, 2021.

[Hussain *et al.*, 2022] Md Shamim Hussain, Mohammed J Zaki, and Dharmashankar Subramanian. Global self-attention as a replacement for graph convolution. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 655–665, 2022.

[Kipf and Welling, 2016] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[Klicpera *et al.*, 2018] Johannes Klicpera, Aleksandar Bojchevski, and Stephan Günnemann. Predict then propagate: Graph neural networks meet personalized pagerank. *arXiv preprint arXiv:1810.05997*, 2018.

[Kong *et al.*, 2023] Kezhi Kong, Jiuhai Chen, John Kirchenbauer, Renkun Ni, C. Bayan Bruss, and Tom Goldstein. GOAT: A global transformer on large-scale graphs. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 17375–17390. PMLR, 23–29 Jul 2023.

[Kreuzer *et al.*, 2021] Devin Kreuzer, Dominique Beaini, William L Hamilton, Vincent Létourneau, and Prudencio Tossou. Rethinking graph transformers with spectral attention. *arXiv preprint arXiv:2106.03893*, 2021.

[Liu *et al.*, 2020] Chao Liu, Xinchuan Li, Dongyang Zhao, Shaolong Guo, Xiaojun Kang, Lijun Dong, and Hong Yao. A-gnn: Anchors-aware graph neural networks for node embedding. In *International Conference on Heterogeneous Networking for Quality, Reliability, Security and Robustness*, pages 141–153. Springer, 2020.

[Müller *et al.*, 2023] Luis Müller, Mikhail Galkin, Christopher Morris, and Ladislav Rampášek. Attending to graph transformers. *arXiv preprint arXiv:2302.04181*, 2023.

[Nguyen *et al.*, 2020] Minh Hai Nguyen, Minh Hoàng Hà, Diep N Nguyen, et al. Solving the k-dominating set problem on very large-scale networks. *Computational Social Networks*, 7(1):1–15, 2020.

[Park *et al.*, 2022] Wonpyo Park, Woong-Gi Chang, Donggeon Lee, Juntae Kim, et al. Grpe: Relative positional encoding for graph transformer. In *ICLR2022 Machine Learning for Drug Discovery*, 2022.

[Rampášek *et al.*, 2022] Ladislav Rampášek, Mikhail Galkin, Vijay Prakash Dwivedi, Anh Tuan Luu, Guy Wolf, and Dominique Beaini. Recipe for a general, powerful, scalable graph transformer. *arXiv preprint arXiv:2205.12454*, 2022.

[Rong *et al.*, 2020] Yu Rong, Yatao Bian, Tingyang Xu, Weiyang Xie, Ying Wei, Wenbing Huang, and Junzhou Huang. Self-supervised graph transformer on large-scale molecular data. *arXiv preprint arXiv:2007.02835*, 2020.

[Vaswani *et al.*, 2017] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[Veličković *et al.*, 2017] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

[Wasserman *et al.*, 1994] Stanley Wasserman, Katherine Faust, et al. Social network analysis: Methods and applications. 1994.

[Wu *et al.*,] Qitian Wu, Wentao Zhao, Zenan Li, David Wipf, and Junchi Yan. Nodeformer: A scalable graph structure learning transformer for node classification. In *Advances in Neural Information Processing Systems*.

[Wu *et al.*, 2018] Zhenqin Wu, Bharath Ramsundar, Evan N Feinberg, Joseph Gomes, Caleb Geniesse, Aneesh S Pappu, Karl Leswing, and Vijay Pande. Moleculenet: a benchmark for molecular machine learning. *Chemical science*, 9(2):513–530, 2018.

[Xu *et al.*, 2018] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826*, 2018.

[Ying *et al.*, 2021] Chengxuan Ying, Tianle Cai, Shengjie Luo, Shuxin Zheng, Guolin Ke, Di He, Yanming Shen, and Tie-Yan Liu. Do transformers really perform bad for graph representation? *arXiv preprint arXiv:2106.05234*, 2021.

[You *et al.*, 2019] Jiaxuan You, Rex Ying, and Jure Leskovec. Position-aware graph neural networks. In *International conference on machine learning*, pages 7134–7143. PMLR, 2019.

[Zhang *et al.*, 2022] Zaixi Zhang, Qi Liu, Qingyong Hu, and Chee-Kong Lee. Hierarchical graph transformer with adaptive node sampling. *arXiv preprint arXiv:2210.03930*, 2022.

[Zhao *et al.*, 2021] Jianan Zhao, Chaozhuo Li, Qianlong Wen, Yiqi Wang, Yuming Liu, Hao Sun, Xing Xie, and Yanfang Ye. Gophormer: Ego-graph transformer for node classification. *arXiv preprint arXiv:2110.13094*, 2021.