Gradformer: Graph Transformer with Exponential Decay

Chuang Liu[1] , **Zelin Yao**[1] , **Yibing Zhan**[2] , **Xueqi Ma**[3] , **Shirui Pan**[4] , **Wenbin Hu**[1*]
[1]School of Computer Science, Wuhan University, Wuhan, China
[2]JD Explore Academy, JD.com, China
[3]School of Computing and Information Systems, The University of Melbourne, Melbourne, Australia
[4]School of Information and Communication Technology, Griffith University, Brisbane, Australia
{chuangliu, zelinyao, hwb}@whu.edu.cn, zhanyibing@jd.com, xueqim@student.unimelb.edu.au,
s.pan@griffith.edu.au

Abstract

Graph Transformers (GTs) have demonstrated their advantages across a wide range of tasks. However, the self-attention mechanism in GTs overlooks the graph's inductive biases, particularly biases related to structure, which are crucial for the graph tasks. Although some methods utilize positional encoding and attention bias to model inductive biases, their effectiveness is still suboptimal analytically. Therefore, this paper presents Gradformer, a method innovatively integrating GT with the intrinsic inductive bias by applying an exponential decay mask to the attention matrix. Specifically, the values in the decay mask matrix diminish exponentially, correlating with the decreasing node proximities within the graph structure. This design enables Gradformer to retain its ability to capture information from distant nodes while focusing on the graph's local details. Furthermore, Gradformer introduces a learnable constraint into the decay mask, allowing different attention heads to learn distinct decay masks. Such an design diversifies the attention heads, enabling a more effective assimilation of diverse structural information within the graph. Extensive experiments on various benchmarks demonstrate that Gradformer consistently outperforms the Graph Neural Network and GT baseline models in various graph classification and regression tasks. Additionally, Gradformer has proven to be an effective method for training deep GT models, maintaining or even enhancing accuracy compared to shallow models as the network deepens, in contrast to the significant accuracy drop observed in other GT models. Codes are available at https://github.com/LiuChuang0059/Gradformer.

1 Introduction

Graph Transformers (GTs) [Dwivedi and Bresson, 2021] have shown remarkable success in achieving state-of-the-art performance across various applications. Unlike the local message-passing in graph neural networks (GNNs), GTs can capture long-range information from distant nodes. Specifically, the self-attention mechanism in GTs allows each node to directly attend to other nodes in a graph, enabling information aggregation from arbitrary nodes.

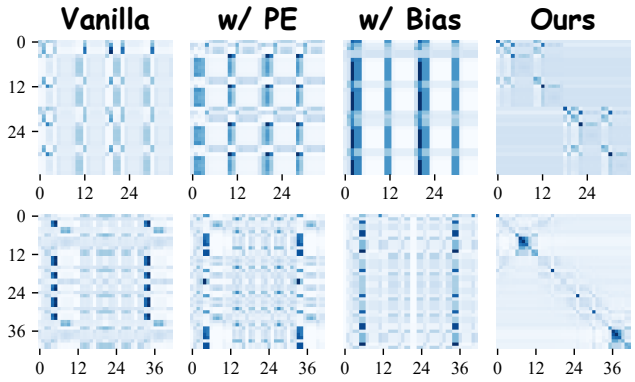

Figure 1: Visualization of attention patterns in different GT models with two graphs from the OGBG-HIV dataset. From left to right: vanilla GT, GT with position encoding (*w/ PE*), GT with attention bias (*w/ Bias*), and GT with our proposed decay mask (*ours*).

The self-attention in GTs offers considerable flexibility and the capacity to aggregate information both globally and adaptively. However, this mechanism significantly neglects intrinsic inductive biases in graphs, which poses challenges in capturing the essential graph structural information. Without considering structural relationships, the indiscriminate attention to all nodes in a graph will render the self-attention mechanism's inadequate focus on key information and the aggregation of redundant information. This comes with the generation of meaningless attention scores and, ultimately, suboptimal learning outcomes. Furthermore, the above issue is particularly evident in scenarios with limited data.

Numerous studies have focused on integrating graph inductive biases into self-attention learning mechanisms. These studies are broadly divided into two main approaches: 1) Injected position encoding: Studies such as GT [Dwivedi and Bresson, 2021] and SAN [Kreuzer *et al.*, 2021] suggest using laplacian eigenfunctions as positional encodings (PEs) to contain the structural characteristics of graphs. 2) Attention bias: Graphormer [Ying *et al.*, 2021] proposes a direct addi-

*Corresponding Author

tion of structural encodings into the attention mechanism as a bias, enhancing the GT's capacity to model graph-structured data. Notably, applying PEs, which are typically concatenated with input features, would affect attention scores and could be considered as a way to introduce attention bias, as illustrated in Figure 2. However, concatenated PEs and attention bias fail to directly and effectively guide attention scores to fully capture structural information in graphs. From the Figure 1, it is evident that the attention patterns in GTs augmented with PE or attention bias exhibit minimal deviation from those observed in the vanilla GTs. This observation suggests that the inductive biases introduced by these methods have a limited effect on the model's attention mechanism. Furthermore, the attention in these models appears to be inattentive (characterized by dense patterns), potentially leading to a failure in focusing on key information and aggregating redundant information. These findings highlight a critical need for more sophisticated approaches capable of effectively incorporating structural insights into the self-attention framework of GTs. Therefore, the goal is to develop a method that not only enhances the model's attention mechanism by focusing on structurally significant features but also reduces redundancy in the information aggregation process.

Different from the aforementioned methods, we propose Gradformer, a novel method that innovatively integrates GTs with inductive bias. Specifically, Gradformer integrates an exponential decay mask into the GT self-attention architecture. This mask, multiplied with the attention score, explicitly controls each node's attention weights relative to other nodes, ensuring that the attention weights decay with an increasing node distance. In addition, the exponential decay ensures a gradual reduction in attention weights at the boundary of the full attention zone, avoiding an abrupt truncation. As a result, the introduced decay mask, rooted in the graph's structural bias, effectively guides the learning process within the self-attention framework. Furthermore, Gradformer incorporates a learnable constraint within the decay mask, applied to the attention heads, dynamically adjusting the node distance for full attention in a graph. This constraint amplifies the model's ability to discern local structural nuances and diversifies the attention heads, facilitating more effective assimilation of diverse structural information.

The design elements of Gradformer offer multiple benefits: **1)** The decay mask effectively integrates a form of prior knowledge, deeply connected to the graph's structural attributes, into the GT models. **2)** The structure-oriented prior knowledge precisely governs each node's interaction radius and hence defines the extent of its attention relative to other nodes, thus preventing the aggregation of redundant information. This is achieved by the design of exponential decay, which enables mask values for distant nodes approaching near-zero levels, thus effectively reducing their influence in the aggregation process. **3)** The exponential decay mask endows the enhanced attention mechanism in Gradformer with the ability to function as a unified form of GNNs and GTs, synergizing their strengths: the local processing power of GNNs and global aggregation capabilities of GTs (further expounded in Section 3.3). In summary, Gradformer empowers the self-attention mechanism to effectively concentrate on structural information within the graph and limit unnecessary aggregation from distant nodes.

To evaluate the effectiveness of our Gradformer, we conduct extensive experiments on various commonly-used datasets, including the large-scale Open Graph Benchmark (OGB) [Hu *et al.*, 2020]. The experimental results consistently demonstrate that Gradformer outperforms state-of-the-art GT models on most datasets with remarkable stability and accuracy improvements, even as network depth increases. This finding stands in stark contrast to the performance of other GT models, where accuracy tends to decline with increased depth [Zhao *et al.*, 2023]. Additionally, incorporating the decay mask design into Gradformer provides notable advantages for the graph classification task in low-resource settings, underscoring the versatility and practicality of our model. Our main contributions are summarized as follows:

1. We propose a general GT with an exponential decay mask attention mechanism, termed Gradformer, which enhances the self-attention mechanism by fully leveraging the graph's structural information. Therefore, Gradformer maintains the ability to capture long-range information while prioritizing the local information of the graph, guided by an intense inductive bias.
2. We conduct extensive experiments to compare Gradformer with 14 GNN and GT baseline models for the graph classification task on various real-world graph datasets, including OGB. The experimental results consistently validate the effectiveness of the proposed Gradformer.

2 Related Work

Graph Transformers. In recent years, many transformer variants have been applied to graph modeling. Unlike GNNs, transformers display competitive or even superior performance across various tasks. Dwivedi *et al.* [2021] were the first to extend the transformer architecture to graphs and propose PEs [Ding *et al.*, 2020]. Subsequently, numerous variants of GT have been proposed, making significant progress in graph-level tasks [Rong *et al.*, 2020; Chen *et al.*, 2021; Wu *et al.*, 2021; Chen *et al.*, 2022; Yin and Zhong, 2023]. However, these methods are primarily designed for graph-level tasks due to the time and memory constraints of the self-attention layer, which requires $\mathcal{O}(n^2)$ complexity. Therefore, multiple studies [Zhao *et al.*, 2021; Choromanski *et al.*, 2022; Guo *et al.*, 2022; Park *et al.*, 2022; Wu *et al.*, 2022; Wu *et al.*, 2023a; Liu *et al.*, 2023; Liu *et al.*, 2024] have been conducted to enhance GTs' scalability and efficiency, facilitating their application in node-level tasks. Additionally, GTs have been applied in various fields, including natural language processing [Cai *et al.*, 2022], computer vision [Zhu *et al.*, 2021; Tang *et al.*, 2023], recommendation systems [Xu *et al.*, 2019a; Li *et al.*, 2023], multimodal contexts [Chen and Li, 2022; He and Wang, 2023], and reinforcement learning [Hu *et al.*, 2023]. For a more detailed introduction, please refer to the recent GT reviews [Rampasek *et al.*, 2022; Min *et al.*, 2022].

Prior Knowledge in Graph Transformer. Numerous efforts have been undertaken to integrate prior knowledge into GTs to augment their performance. This integration can be

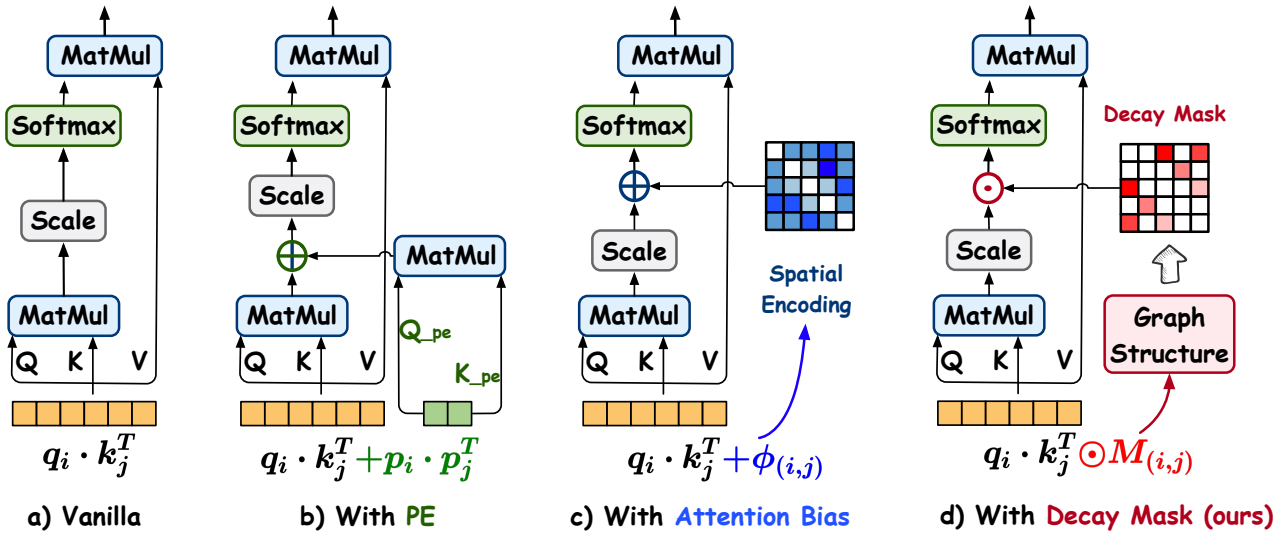

Figure 2: The overview of the Gradformer framework and its comparison with existing methods. **a) Vanilla:** Vanilla self-attention mechanism serves as the baseline. **b) With PE:** In several works [Dwivedi and Bresson, 2021], the PE vector (*i.e.*, $\mathbf{p}_i$) is concatenated with the input node features, which can be interpreted as introducing a bias in the attention score. **c) With Attention Bias:** Some methods [Ying *et al.*, 2021] incorporate an attention bias (*i.e.*, $\phi(i,j)$) into the attention score calculation. This bias often derives from spatial information, such as the shortest path. **d) Ours:** Our method introduces an exponential decay mask that is multiplied with the attention scores. This mask is derived from the structural information of the graph. Moreover, different attention heads utilize distinct masks, made possible through learnable parameters. A comprehensive explanation of the symbols is provided in Section 3.

classified into three categories. *1) Proposed position encodings:* SAN [Kreuzer *et al.*, 2021] suggests utilizing learned Laplace eigenfunctions as a substitute for the original PE. GraphGPS [Rampasek *et al.*, 2022] summarizes possible positional and structural encodings for GTs. *2) Improved self-attention:* Graphormer [Ying *et al.*, 2021] suggests adding structural encodings to the attention metric as a bias to enhance GTs' structured-data modeling. Moreover, RT [Diao and Loynd, 2023] and EGT [Hussain *et al.*, 2022] consider incorporating the edge vectors in their self-attention calculations. *3) Modified architecture:* GraphTrans [Wu *et al.*, 2021] introduces a transformer sub-network above a GNN layer. In GraphGPS, the transformer and GNN layers are placed in parallel. In this manuscript, our proposed method lies on the line of improving the self-attention learning.

3 Methodology

3.1 Preliminaries

Notations. A graph $\mathcal{G}$ can be represented by an adjacency matrix $\mathbf{A} \in \{0,1\}^{n\times n}$ and a node feature matrix $\mathbf{X} \in \mathbb{R}^{n\times d}$, where n is the number of nodes, d is the node feature dimension, and $\mathbf{A}[i,j] = 1$ if an edge between nodes v_i and v_j exists, otherwise, $\mathbf{A}[i,j] = 0$.

Graph Transformer. GTs [Vaswani *et al.*, 2017; Ying *et al.*, 2021] consist of two essential parts: a multi-head self-attention (MHA) module and a feed-forward network (FFN). Given the node embedding matrix $\mathbf{H}^{(l)} \in \mathbb{R}^{n\times d^{(l)}}$ in a graph, a single attention head is computed as follows:

$$\mathbf{H}^{(l+1)} = \text{softmax}\left(\frac{\mathbf{Q}^{(l)}(\mathbf{K}^{(l)})^\top}{\sqrt{d^{(l)}}}\right)\mathbf{V}^{(l)}, \qquad (1)$$

where $\mathbf{H}^{(l+1)} \in \mathbb{R}^{n\times d^{(l+1)}}$ is the output matrix, $d^{(l+1)}$ is the output hidden dimension, and $\mathbf{Q}^{(l)} \in \mathbb{R}^{n\times d^{(l)}}$, $\mathbf{K}^{(l)} \in \mathbb{R}^{n\times d^{(l)}}$, and $\mathbf{V}^{(l)} \in \mathbb{R}^{n\times d^{(l)}}$ are the query, key, and value vectors, respectively, which are the projection results of $\mathbf{H}^{(l)} \in \mathbb{R}^{n\times d^{(l)}}$:

$$\mathbf{Q}^{(l)} = \mathbf{H}^{(l)}\mathbf{W}^Q; \mathbf{K}^{(l)} = \mathbf{H}^{(l)}\mathbf{W}^K; \mathbf{V}^{(l)} = \mathbf{H}^{(l)}\mathbf{W}^V, \quad (2)$$

where $\mathbf{W}^Q \in \mathbb{R}^{d^{(l)}\times d^{(l+1)}}$, $\mathbf{W}^K \in \mathbb{R}^{d^{(l)}\times d^{(l+1)}}$, and $\mathbf{W}^V \in \mathbb{R}^{d^{(l)}\times d^{(l+1)}}$ are projection matrices. Note that the above single-head self-attention module can be generalized into a MHA via the concatenation operation.

3.2 Proposed Method: Gradformer

This subsection introduces the Gradformer method. First, we describe the fundamental Gradformer architecture with a primary focus on its central module, the exponential decay mask attention mechanism. The mechanism comprises two components: exponential decay and learnable constraints. Subsequently, we explore the Gradformer method in detail, with a focus on its general form and computational complexity.

Architecture

As depicted in Figure 2, Gradformer extends the original GT architecture by incorporating a *decay mask*. The decay mask, derived from the graph's structural information, introduces a strong inductive bias into the self-attention learning.

Masking Attention with Exponential Decay. According to Eq. (1), we observe that the attention matrix is comparable to a row-normalized adjacency matrix of a directly weighted complete graph. This dictates the aggregation of node features in a graph, similar to GCN. Unlike the input graph, this

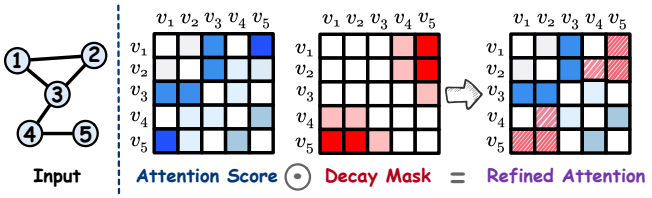

Figure 3: The decay mask mechanism. The blue matrix represents the pairwise dot product, where the intensity of the blue color indicates the magnitude of the attention. The red matrix represents the decay mask, where the intensity of the red color indicates the magnitude of the mask. Once the mask is applied (indicated by red cells with diagonal stripes), the attention values in the masked cells become significantly attenuated. With this attention decay masking, the self-attention mechanism becomes more responsive to the graph's structural characteristics.

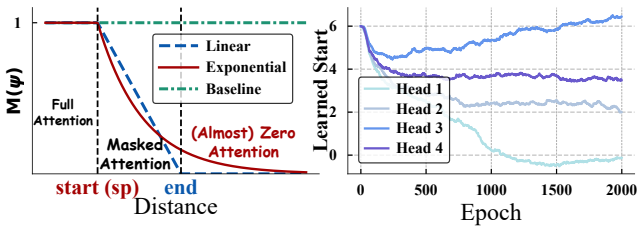

Figure 4: **Left:** The value of the decay mask matrix $\mathbf{M}$ varies with the node-wise distance, ψ. Please note that the configuration of an **end** point is unique to linear decay, whereas exponential decay does not require such a parameter. Furthermore, the **start** point (*i.e.*, sp in Eq. (5)) is a learnable parameter, as depicted in the right part. **Right:** The learned start points for different attention heads demonstrate variation across epochs during training on the ZINC dataset.

dynamically forms graph representations through the attention mechanism. However, the basic transformer does not directly consider the input structure (*i.e.*, existing edges) when forming these weighted graphs (*i.e.*, the attention matrices). Consequently, the attention mechanism may overlook the importance of certain neighbouring nodes due to feature similarity, and hence these dynamic graphs collapse immediately after aggregation. To address this issue, we introduce a decay mask $\mathbf{M}$, related to the graph's structural information, to participate in the aggregation process as follows:

$$\mathbf{M} = \lambda^{\psi(v_i, v_j)} \in \mathbb{R}^{n \times n}, \tag{3}$$

where λ is a scalar and $\psi(v_i, v_j)$ refers to a function that measures the spatial distance between v_i and v_j in a graph. The function $\psi(\cdot)$ can be defined by the connectivity between the nodes in the graph. In this paper, we choose $\psi(v_i, v_j)$ to present the distance of the shortest path (SPH) between v_i and v_j if the two nodes are connected. If not, we set the output of $\psi(\cdot)$ to a special value (*i.e.*, -1). Based on the study [Sun *et al.*, 2023], our approach configures the attention mask $\mathbf{M}$ to decrease monotonically and exponentially, as depicted in the left part of Figure 4. Compared to methods with linear decay, exponential decay eliminates the need for an additional endpoint and effectively filters attention at extended distances, with values for distant nodes diminishing to nearly zero. After obtaining the attention mask, it is multiplied with the attention score $\mathbf{S} \in \mathbb{R}^{n \times n}$, as illustrated in Figure 3, and formally defined as:

$$\begin{aligned} \tilde{\mathbf{S}}^{(l)} &= \text{softmax}\left(\mathbf{S}^{(l)} \odot \mathbf{M}^{(l)}\right), \\ \text{Attn}\left(\mathbf{Q}^{(l)}, \mathbf{K}^{(l)}, \mathbf{V}^{(l)}\right) &= \tilde{\mathbf{S}}^{(l)} \mathbf{V}^{(l)}, \end{aligned} \tag{4}$$

where $\tilde{\mathbf{S}}^{(l)}$ denotes the refined attention at layer l, and Attn denotes the self-attention operation with decay mask.

Mask Decay with Learnable Constraints. To enhance the attention mechanism's ability to model local structural information, given the strong locality inherent in graphs, a refined version of Eq. (3) is provided. This version incorporates a distance constraint into the decay operation, effectively tailoring the attention mechanism to be more sensitive to the local information. The refined decay mask is as follows:

$$\mathbf{M} = \lambda^{\text{Relu}(\psi(v_i, v_j) - sp)}, \tag{5}$$

where sp is a scalar that defines the starting point of the decay. Specifically, if the spatial distance in $\psi(v_i, v_j)$ is less than sp, the attention score between v_i and v_j remains unaffected by the decay. Hence, this design accurately controls each node's interaction radius, delineating the extent of its attention relative to other nodes and preventing the aggregation of redundant information.

Furthermore, to enable various attention heads to capture diverse structural information, we make the constraints sp learnable. Consequently, when the decay mechanism is extended to multi-head attention, a decay mask bearing different constraints is applied to each head, as shown below:

$$\tilde{\mathbf{S}}^{(l)} = \left[\mathbf{S}^{(l,1)} \odot \mathbf{M}^{(l,1)}; \hat{\mathbf{S}}^{(l,2)} \odot \mathbf{M}^{(l,2)}; \cdots \hat{\mathbf{S}}^{(l,h)} \odot \mathbf{M}^{(l,h)}\right],$$

where h represents the total number of attention heads. As illustrated in the right part of Figure 4, the constraints of different attention heads are learned to capture various structural information. Specifically, we observe that the learned constraint for Head 3 is relatively large. This characteristic enables Head 3 to focus on global information of the graph. Conversely, Head 1 exhibits a different learning pattern, which allows model to capture proximate information.

3.3 Discussion

Gradformer is a General Form of GNNs and GTs. In essence, Gradformer represents a more generalized form of the GNN [Veličković *et al.*, 2018] and GT [Dwivedi and Bresson, 2021] models. This generalization becomes evident when considering the behavior of the decay mask $\mathbf{M}$ under varying settings of the parameter λ. Specifically, when $\lambda = 1$, the decay mask in Eq. (3) essentially transforms into an all-ones matrix. In this configuration, Gradformer effectively mirrors the GT model, as the decay mask exerts no modifying influence on the attention scores. Additionally, when $\lambda = 0$, the decay mask $\mathbf{M}_0$ retains the mask value of the 1-hop neighbors as one and sets all others to zero. This configuration parallels the adjacency matrix $\mathbf{A}$, thereby aligning Gradformer closely with the GNN models. The formal

	NCI1	PROTE.	MUTAG	COLLAB	IMDB-B	PATTERN	CLUSTER	MOLHIV	ZINC
	Acc. ↑	**Acc.** ↑	**Acc.** ↑	**Acc.** ↑	**Acc.** ↑	**Acc.** ↑	**Acc.** ↑	**AUROC.** ↑	**MAE** ↓
GCN-based methods									
GCN [Kipf and Welling, 2017]	$79.68_{\pm 2.05}$	$71.7_{\pm 4.7}$	$73.4_{\pm 10.8}$	$71.92_{\pm 1.18}$	$74.3_{\pm 4.6}$	$71.89_{\pm 0.33}$	$69.50_{\pm 0.98}$	$75.99_{\pm 1.19}$	$0.367_{\pm 0.011}$
GAT [Veličković *et al.*, 2018]	$79.88_{\pm 0.88}$	$72.0_{\pm 3.3}$	$73.9_{\pm 10.7}$	$75.8_{\pm 1.6}$	$74.7_{\pm 4.7}$	$78.27_{\pm 0.19}$	$70.59_{\pm 0.45}$	–	$0.384_{\pm 0.007}$
GIN [Xu *et al.*, 2019b]	$81.7_{\pm 1.7}$	$73.76_{\pm 4.61}$	$84.5_{\pm 8.9}$	$73.32_{\pm 1.08}$	$75.1_{\pm 4.9}$	$85.39_{\pm 0.14}$	$64.72_{\pm 1.55}$	$77.07_{\pm 1.49}$	$0.526_{\pm 0.051}$
GatedGCN [Li *et al.*, 2016]	$81.17_{\pm 0.79}$	$74.65_{\pm 1.13}$	$85.00_{\pm 2.67}$	$80.70_{\pm 0.75}$	$73.20_{\pm 1.32}$	$85.57_{\pm 0.09}$	$73.84_{\pm 0.33}$	–	$0.282_{\pm 0.015}$
Graph Transformer-based methods									
GT [Dwivedi and Bresson, 2021]	$80.15_{\pm 2.04}$	$73.94_{\pm 3.78}$	$83.9_{\pm 6.5}$	$79.63_{\pm 1.02}$	$73.10_{\pm 2.11}$	$84.81_{\pm 0.07}$	$73.17_{\pm 0.62}$	–	$0.226_{\pm 0.014}$
SAN [Kreuzer *et al.*, 2021]	$80.50_{\pm 1.30}$	$74.11_{\pm 3.07}$	$78.8_{\pm 2.9}$	$79.42_{\pm 1.61}$	$72.10_{\pm 2.30}$	$86.58_{\pm 0.04}$	$76.69_{\pm 0.65}$	$77.85_{\pm 2.47}$	$0.139_{\pm 0.006}$
Graphormer [Ying *et al.*, 2021]	$81.44_{\pm 0.57}$	$75.29_{\pm 3.10}$	$80.52_{\pm 5.79}$	$\underline{81.80_{\pm 2.24}}$	$73.40_{\pm 2.80}$	$86.65_{\pm 0.03}$	$74.66_{\pm 0.24}$	$74.55_{\pm 1.06}$	$0.122_{\pm 0.006}$
GraphTrans [Wu *et al.*, 2021]	$82.60_{\pm 1.20}$	$75.18_{\pm 3.36}$	$\underline{87.22_{\pm 7.05}}$	$79.81_{\pm 0.84}$	$74.50_{\pm 2.89}$	–	–	$76.33_{\pm 1.11}$	–
SAT [Chen *et al.*, 2022]	$80.69_{\pm 1.55}$	$73.32_{\pm 2.36}$	$80.50_{\pm 2.84}$	$80.05_{\pm 0.55}$	$75.90_{\pm 0.94}$	$86.85_{\pm 0.04}$	$77.86_{\pm 0.10}$	–	$0.094_{\pm 0.008}$
EGT [Hussain *et al.*, 2022]	$81.91_{\pm 3.42}$	–	–	–	–	$86.82_{\pm 0.02}$	$\mathbf{79.23_{\pm 0.35}}$	$80.51^{**}_{\pm 0.30}$	$0.108_{\pm 0.009}$
GraphGPS [Rampasek *et al.*, 2022]	$\underline{84.21_{\pm 2.25}}$	$\underline{75.77_{\pm 2.19}}$	$85.00_{\pm 3.16}$	$81.40_{\pm 1.26}$	$\mathbf{77.40_{\pm 0.63}}$	$86.69_{\pm 0.06}$	$78.02_{\pm 0.18}$	$78.80_{\pm 0.49}$	$0.070_{\pm 0.004}$
LGI-GT [Yin and Zhong, 2023]	$82.18_{\pm 1.90}$	–	–	–	–	$\mathbf{86.93_{\pm 0.04}}$	$78.19_{\pm 0.10}$	–	$\mathbf{0.069_{\pm 0.002}}$
KDLGT [Wu *et al.*, 2023b]	–	–	–	–	–	–	–	$\underline{78.98_{\pm 1.78}}$	$0.130_{\pm 0.002}$
DeepGraph [Zhao *et al.*, 2023]	–	–	–	–	–	$90.66^{*}_{\pm 0.06}$	$77.91_{\pm 0.14}$	–	$0.072_{\pm 0.004}$
Gradformer (ours)	$\mathbf{86.01_{\pm 1.47}}$	$\mathbf{77.50_{\pm 1.86}}$	$\mathbf{88.00_{\pm 2.45}}$	$\mathbf{82.01_{\pm 1.06}}$	$\underline{77.10_{\pm 0.54}}$	$\underline{86.89_{\pm 0.07}}$	$\underline{78.55_{\pm 0.16}}$	$\mathbf{79.15_{\pm 0.89}}$	$\mathbf{0.069_{\pm 0.002}}$

Notations: *: DeepGraph utilizes a distinct evaluation metric on the pattern dataset, diverging from other models. Employing this metric, our method achieves an accuracy of $90.88_{\pm 0.06}$. **: The result is derived from the fine-tuning of a large pretrained model.

Table 1: Experimental results on eight common datasets (the mean accuracy (**Acc.**), AUROC, and MAE, and standard deviation over 10 different runs). **Bold**: the best performance per dataset. Underline: the second best performance per dataset.

derivation of the above process is as follows:

$$\mathbf{H}^{(l+1)} = \frac{1}{\sqrt{d^{(l)}}} \left(\mathbf{S}^{(l)} \odot \mathbf{M}_0^{(l)} \right) \mathbf{H}^{(l)} \mathbf{W}; \quad \mathbf{M}_0^{(l)} = \mathbf{A}. \quad (6)$$

Taking the GCN layer as an example, the message passing mechanism is expressed as follows:

$$\mathbf{H}^{(l+1)} = \sigma \left(\mathbf{A} \mathbf{H}^{(l)} \mathbf{W} \right), \quad (7)$$

where σ denotes the Sigmoid activation function. From these equations, we can deduce that Gradformer essentially constitutes a generalized GNN form, possessing at least the same expressive capability as traditional GNNs.

In summary, Gradformer surpasses traditional GNNs by broadening its receptive field to encompass more relevant nodes. Furthermore, compared to GTs, Gradformer demonstrates superior capacity in fusing node representations with graph structure, capturing more topological information.

Computational Complexity. This subsection analyzes Gradformer's complexity. Compared to vanilla GTs, Gradformer incurs only the additional computational cost of pointwise multiplication between the decay and attention matrices. This process has significantly less computational complexity than self-attention. Additionally, Gradformer's additional memory requirement is the mask matrix, which shares the same size as the self-attention one. To illustrate Gradformer's complexity, detailed experimental results for time and memory are provided in the following section (See Figure 7).

4 Experiments

4.1 Experimental Settings

Datasets. We utilize nine commonly-used real-world datasets from various sources to ensure diversity, including five graph datasets from the TU database [Morris *et al.*, 2020] (*i.e.*, NCI1, PROTEINS, MUTAG, IMDB-B, and COLLAB), three datasets from Benchmarking GNN [Dwivedi *et al.*, 2023] (*i.e.*, PATTERN, CLUSTER, and ZINC), and one dataset from OGB [Hu *et al.*, 2020] (*i.e.*, OGBG-MOLHIV), involving diverse domains (*e.g.*, synthetic, social, biology, and chemistry), sizes (*e.g.*, ZINC and OGBG-MOLHIV are large datasets), and tasks (*e.g.*, node classification, graph classification and regression). For all datasets, we strictly follow the evaluation metrics and dataset split recommended by the given benchmarks [Ying *et al.*, 2021].

Baseline. To demonstrate the effectiveness of our proposed method, we compare Gradformer with the following 14 baselines: **(I) 4 standard GCN-based models:** GCN [2017],

Dataset	Method	4 layers	12 layers	24 layers
NCI1	GraphGPS	$84.21_{\pm 2.25}$	$84.09_{\pm 1.68}$	$71.90_{\pm 1.46}$
	Ours	$\mathbf{86.01}_{\pm 1.47}$	$\mathbf{84.31}_{\pm 1.26}$	$\mathbf{84.25}_{\pm 1.77}$
	Method	**12 layers**	**24 layers**	**48 layers**
CLUSTER	GraphGPS	$\mathbf{78.06}_{\pm 0.12}$	$\mathbf{78.39}_{\pm 0.14}$	$70.91_{\pm 3.29}$
	Ours	$77.78_{\pm 0.25}$	$78.31_{\pm 0.10}$	$\mathbf{78.55}_{\pm 0.14}$
PATTERN	GraphGPS	$86.74_{\pm 0.04}$	$86.68_{\pm 0.06}$	$85.83_{\pm 0.38}$
	Ours	$\mathbf{86.75}_{\pm 0.05}$	$\mathbf{86.78}_{\pm 0.04}$	$\mathbf{86.89}_{\pm 0.07}$

Table 2: Performance of Gradformer with deep layers.

GAT [2018], GIN [2019b], and GatedGCN [2016]; **(II) 10 GT-based graph models:** GraphTransformer [2021], SAN [2021], Graphormer [2021], GraphTrans [2021], SAT [2022], EGT [2022], GraphGPS [2022], LGI-GT [2023], KDLGT [2023b], and DeepGraph [2023]. For each baseline, we utilize the recommended settings as per the official implementation guidelines.

Implementation Details. We assess our proposed model's effectiveness by measuring its performance in graph classification and regression tasks. To ensure reliability, we conduct 10 trials for each model using different random seeds. Accordingly, we report the average test accuracy/AUROC/MAE based on the epoch when the best validation accuracy/AUROC/MAE is achieved. Furthermore, all the experiments are conducted on a server equipped with 8 NVIDIA A100s.

4.2 Overall Performance

We evaluate the proposed model's effectiveness by comparing it with GT models that are both shallow and deep. For each model and dataset, we conduct 10 trials with random seeds, and then measure the mean accuracy and standard deviation, which are reported in Tables 1 and 2.

Performance of Gradformer. From the results in Table 1, we observe that: **1)** Gradformer demonstrates exceptional performance, achieving state-of-the-art results on five datasets while remaining competitive on three others. This highlights the efficacy of the proposed method. **2)** Notably, on small-scale datasets such as NCI1 and PROTEINS, Gradformer outperforms all 14 methods, exhibiting improvements of 2.13% and 2.28%, respectively. This demonstrates Gradformer's effective integration of inductive biases into the GT model, a crucial advantage especially in scenarios with limited data availability. To further validate this observation, additional experiments under low-resource conditions for the large-scale datasets have been conducted (see next section). **3)** Furthermore, Gradformer delivers competitive results on large-scale datasets (*e.g.*, ZINC), demonstrating its potential applicability across various dataset scales.

Performance of Gradformer with Deep Layers. Table 2 displays the results of Gradformer with deep layers. Based on these results, the following observations can be made. **1)** Increasing the number of layers notably decreases the accuracy of deep GTs, such as GraphGPS (*e.g.*, a drop of 14.4% on NCI1 and 9.54% on CLUSTER). **2)** In contrast, Gradformer demonstrates a significant improvement in this context. It sustains performance on the NCI1 dataset and surpasses shallow models on CLUSTER and PATTERN datasets. This improvement can be attributed to Gradformer's inherent flexibility in accommodating increased model depth. Specifically, while self-attention in shallow networks is inclined to focus locally, it adopts a more global perspective in deeper networks. Therefore, incorporating residual connections enhances information flow between layers, stabilizing the learning process. The decay mask utilized by Gradformer plays a pivotal role in ensuring that the updated node information maintains higher fidelity to the original data, producing effects similar to those observed in shallower models.

NCI1 ↑	5%	10%	25%	100%
GraphGPS	$69.54_{\pm 0.96}$	$74.70_{\pm 0.44}$	$76.55_{\pm 1.19}$	$84.21_{\pm 2.25}$
Ours	$\mathbf{71.20}_{\pm 0.49}$	$\mathbf{76.38}_{\pm 0.85}$	$\mathbf{77.98}_{\pm 1.22}$	$\mathbf{86.01}_{\pm 1.47}$
ZINC ↓	5%	10%	25%	100%
GraphGPS	$0.438_{\pm 0.021}$	$0.295_{\pm 0.012}$	$0.182_{\pm 0.014}$	$0.070_{\pm 0.004}$
Ours	$\mathbf{0.429}_{\pm 0.019}$	$\mathbf{0.289}_{\pm 0.011}$	$\mathbf{0.171}_{\pm 0.009}$	$\mathbf{0.069}_{\pm 0.002}$

Table 3: Results on Low-resource Settings. 5% denotes the utilization of 5% of the datasets as the training sets.

	NCI1	PATTERN	CLUSTER	ZINC
GraphGPS	84.2	86.68	78.02	0.070
w/o MPNN	**80.04** ↓ 4.16	71.01 ↓ 15.67	68.29 ↓ 9.73	0.217 ↓ 0.147
w/o PE	**83.67** ↓ 0.53	86.63 ↓ 0.05	77.27 ↓ 0.75	0.113 ↓ 0.043
Ours	86.01	86.56	78.22	0.069
w/o MPNN	80.80 ↓ 5.21	**86.49** ↓ 0.07	**73.92** ↓ 4.30	**0.187** ↓ 0.118
w/o PE	85.01 ↓ 1.00	**86.70** ↑ 0.14	**77.58** ↓ 0.64	0.116 ↓ 0.047

Table 4: Analysis of Gradformer's performance without the MPNN or PE module. The downward arrow (↓) denotes a reduction in model performance relative to the baseline method.

4.3 Further Discussions

Results on Low-resource Settings. We conduct experiments using datasets with a reduced number of labels on NCI1 and ZINC datasets. The results in Table 3 indicate that Gradformer consistently outperforms GraphGPS across all settings, particularly when the number of labels is very low, with improvements of 2.4% and 2.3% for 5% and 10% labeled data, respectively. This finding indicates that our method effectively incorporates the inductive biases into the GTs, enabling them to efficiently assimilate graph information with a limited number of labeled datasets. Furthermore, this discovery highlights the potential applicability of our model under resource-constrained conditions, such as the biomedical field where labeled data are scarce and costly.

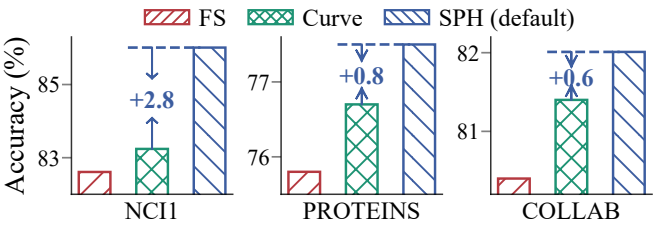

Figure 5: Ablation study of graph structure index.

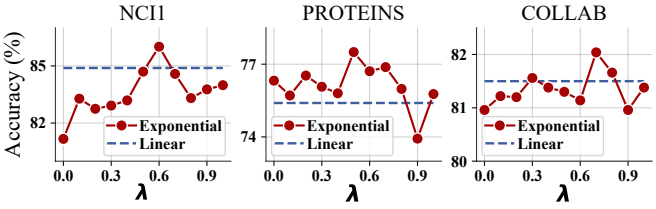

Figure 6: The parameter analysis of the decay ratio, along with an ablation study of the decay function.

Performance without MPNN or Position Encoding. In our subsequent analysis, we examine the impact of two key GT components: the Message Passing Neural Network (MPNN) module and PE. This investigation involves four graph datasets. It is important to note that except for the specific components under study, all other aspects of the models are maintained in line with the complete models. The findings, summarized in Table 4, indicate that: **1)** The removal of the MPNN module leads to a slight decline in Gradformer's performance, especially in contrast to GraphGPS. For instance, the performance of GraphGPS without MPNN diminishes by 15.67 on the PATTERN dataset, whereas Gradformer only experiences a reduction of 0.07. **2)** The absence of PE seems to have a minimal, and in some cases even beneficial, impact on Gradformer. The model demonstrates improved performance on certain datasets (*i.e.*, PATTERN) when PE is omitted. This finding suggests that Gradformer's architecture, particularly its decay mask design, can compensate for the absence of positional information. In summary, these results imply that our method effectively enables self-attention mechanisms to capture more structural information.

The Impact of Graph Structure Index. As detailed in Section 3.2, our model's decay mask is formulated based on specific graph structural indices. In addition to the previously mentioned SPH, our study also examines another structural index: the discrete Ricci curvature (Curve) [Lai *et al.*, 2023]. This index is the graph distance based on Riemannian manifold. Furthermore, we evaluate the effectiveness of a feature-based index, specifically the feature cosine similarity (FS). The results, illustrated in Figure 5, show that SPH consistently outperforms the other two indexes. Notably, the accuracy achieved with FS is significantly lower than that obtained with the structural indices (*i.e.*, SPH and Curve). These findings further emphasize the importance of incorporating structural information in enhancing model performance.

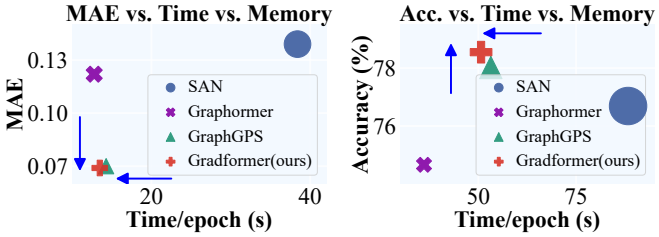

Figure 7: Efficiency analysis comparing Gradformer with three baseline models on two datasets: ZINC (*left*) and CLUSTER (*right*). The size of the markers indicates GPU memory usage.

The Impact of Decay Ratio and Decay Function. In an analysis conducted on three datasets, NCI1, PROTEINS, and COLLAB, the impact of varying the parameter λ is examined. The results are depicted in Figure 6. These findings reveal that as λ increases, the model's accuracy initially rises, reaching optimal performance typically around the values of 0.5 or 0.6. Subsequently, the model's accuracy declines. Furthermore, as elaborated in Section 3.3, specific λ values confer distinct characteristics to Gradformer. For instance, at $\lambda = 1$, Gradformer aligns with the conventional GT model, whereas at $\lambda = 0$, it nearly regresses to a GNN model. Notably, across varying decay ratios, Gradformer consistently exhibits superior performance over the standard GNN model. In addition, observations indicate that the exponential decay function consistently outperforms the linear one.

Efficiency Analysis. To validate Gradformer's efficiency, its training cost is compared with prominent methods such as SAN [Kreuzer *et al.*, 2021], Graphormer [Ying *et al.*, 2021], and GraphGPS [Rampasek *et al.*, 2022], with a focus on metrics such as running time and GPU memory usage. The comparative results are presented in Figure 7. These findings reveal that Gradformer achieves an optimal balance between efficiency and effectiveness. Notably, Gradformer outperforms SAN and GraphGPS in computational efficiency and accuracy. Although Gradformer exhibits a marginally longer runtime compared to Graphormer, the former significantly surpasses the latter in accuracy, highlighting its superiority in balancing resource usage with high-performance outcomes.

5 Conclusion

In this study, we introduce Gradformer, a novel integration of GT with intrinsic inductive biases, achieved by applying an exponential decay mask with learnable parameters to the attention matrix. Through extensive experimentation across 9 graph datasets, Gradformer has shown its superiority by outperforming 14 contemporary GTs and GNNs. Notably, Gradformer's strength lies in its ability to maintain or even surpass shallow models in accuracy while deepening the network architecture. This is a feat not commonly observed in other GTs where accuracy tends to decline significantly in the same context. Despite its competitive performance, Gradformer still has areas for further improvement, including **1)** exploring the feasibility of achieving state-of-the-art structure without the use of MPNN, and **2)** investigating the potential for the decay mask operation to significantly improve GT efficiency.

Acknowledgments

The work of Wenbin Hu was supported by the National Key Research and Development Program of China (2023YFC2705700). This work was supported in part by the Natural Science Foundation of China (No. 82174230), Artificial Intelligence Innovation Project of Wuhan Science and Technology Bureau (No. 2022010702040070), Natural Science Foundation of Shenzhen City (No. JCYJ20230807090211021).

Contribution Statement

Chuang Liu and Zelin Yao contributed equally to this work.

References

[Cai *et al.*, 2022] Weishan Cai, Wenjun Ma, Jieyu Zhan, and Yuncheng Jiang. Entity alignment with reliable path reasoning and relation-aware heterogeneous graph transformer. In *IJCAI*, 2022.

[Chen and Li, 2022] Sijia Chen and Baochun Li. Multimodal dynamic graph transformer for visual grounding. In *CVPR*, 2022.

[Chen *et al.*, 2021] Jianwen Chen, Shuangjia Zheng, Ying Song, Jiahua Rao, and Yuedong Yang. Learning attributed graph representation with communicative message passing transformer. In *IJCAI*, 2021.

[Chen *et al.*, 2022] Dexiong Chen, Leslie O'Bray, and Karsten Borgwardt. Structure-aware transformer for graph representation learning. In *ICML*, 2022.

[Choromanski *et al.*, 2022] Krzysztof Choromanski, Han Lin, Haoxian Chen, Tianyi Zhang, Arijit Sehanobish, Valerii Likhosherstov, Jack Parker-Holder, Tamas Sarlos, Adrian Weller, and Thomas Weingarten. From block-toeplitz matrices to differential equations on graphs: towards a general theory for scalable masked transformers. In *ICML*, 2022.

[Diao and Loynd, 2023] Cameron Diao and Ricky Loynd. Relational attention: Generalizing transformers for graph-structured tasks. In *ICLR*, 2023.

[Ding *et al.*, 2020] Liang Ding, Longyue Wang, and Dacheng Tao. Self-attention with cross-lingual position representation. In *ACL*, 2020.

[Dwivedi and Bresson, 2021] Vijay Prakash Dwivedi and Xavier Bresson. A generalization of transformer networks to graphs. In *AAAI Workshop*, 2021.

[Dwivedi *et al.*, 2023] Vijay Prakash Dwivedi, Chaitanya K Joshi, Anh Tuan Luu, Thomas Laurent, Yoshua Bengio, and Xavier Bresson. Benchmarking graph neural networks. *Journal of Machine Learning Research*, 2023.

[Guo *et al.*, 2022] Lingbing Guo, Qiang Zhang, and Huajun Chen. Unleashing the power of transformer for graphs. *arXiv:2202.10581*, 2022.

[He and Wang, 2023] Xuehai He and Xin Wang. Multimodal graph transformer for multimodal question answering. In *ACL*, 2023.

[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:2005.00687*, 2020.

[Hu *et al.*, 2023] Shengchao Hu, Li Shen, Ya Zhang, and Dacheng Tao. Graph decision transformer. *arXiv preprint arXiv:2303.03747*, 2023.

[Hussain *et al.*, 2022] Md Shamim Hussain, Mohammed J. Zaki, and Dharmashankar Subramanian. Global self-attention as a replacement for graph convolution. In *SIGKDD*, 2022.

[Kipf and Welling, 2017] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *ICLR*, 2017.

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

[Lai *et al.*, 2023] Xin Lai, Yang Liu, Rui Qian, Yong Lin, and Qiwei Ye. Deeper exploiting graph structure information by discrete ricci curvature in a graph transformer. *Entropy*, 2023.

[Li *et al.*, 2016] Yujia Li, Richard Zemel, Marc Brockschmidt, and Daniel Tarlow. Gated graph sequence neural networks. In *ICLR*, 2016.

[Li *et al.*, 2023] Chaoliu Li, Lianghao Xia, Xubin Ren, Yaowen Ye, Yong Xu, and Chao Huang. Graph transformer for recommendation. In *SIGIR*, 2023.

[Liu *et al.*, 2023] Chuang Liu, Yibing Zhan, Xueqi Ma, Liang Ding, Dapeng Tao, Jia Wu, and Wenbin Hu. Gapformer: Graph transformer with graph pooling for node classification. In *IJCAI*, 2023.

[Liu *et al.*, 2024] Chuang Liu, Yibing Zhan, Xueqi Ma, Liang Ding, Dapeng Tao, Jia Wu, Wenbin Hu, and Bo Du. Exploring sparsity in graph transformers. *Neural Networks*, 2024.

[Min *et al.*, 2022] Erxue Min, Runfa Chen, Yatao Bian, Tingyang Xu, Kangfei Zhao, Wenbing Huang, Peilin Zhao, Junzhou Huang, Sophia Ananiadou, and Yu Rong. Transformer for graphs: An overview from architecture perspective. *arXiv:2202.08455*, 2022.

[Morris *et al.*, 2020] Christopher Morris, Nils M Kriege, Franka Bause, Kristian Kersting, Petra Mutzel, and Marion Neumann. Tudataset: A collection of benchmark datasets for learning with graphs. *arXiv:2007.08663*, 2020.

[Park *et al.*, 2022] Jinyoung Park, Seongjun Yun, Hyeonjin Park, Jaewoo Kang, Jisu Jeong, Kyung-Min Kim, Jung-woo Ha, and Hyunwoo J Kim. Deformable graph transformer. *arXiv:2206.14337*, 2022.

[Rampasek *et al.*, 2022] Ladislav Rampasek, Mikhail Galkin, Vijay Prakash Dwivedi, Anh Tuan Luu, Guy Wolf, and Dominique Beaini. Recipe for a general, powerful, scalable graph transformer. In *NeurIPS*, 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. In *NeurIPS*, 2020.

[Sun *et al.*, 2023] Yutao Sun, Li Dong, Shaohan Huang, Shuming Ma, Yuqing Xia, Jilong Xue, Jianyong Wang, and Furu Wei. Retentive network: A successor to transformer for large language models. *arXiv preprint arXiv:2307.08621*, 2023.

[Tang *et al.*, 2023] Hao Tang, Zhenyu Zhang, Humphrey Shi, Bo Li, Ling Shao, Nicu Sebe, Radu Timofte, and Luc Van Gool. Graph transformer gans for graph-constrained house generation. In *CVPR*, 2023.

[Vaswani *et al.*, 2017] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017.

[Veličković *et al.*, 2018] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *ICLR*, 2018.

[Wu *et al.*, 2021] Zhanghao Wu, Paras Jain, Matthew Wright, Azalia Mirhoseini, Joseph E Gonzalez, and Ion Stoica. Representing long-range context for graph neural networks with global attention. In *NeurIPS*, 2021.

[Wu *et al.*, 2022] Qitian Wu, Wentao Zhao, Zenan Li, David Wipf, and Junchi Yan. Nodeformer: A scalable graph structure learning transformer for node classification. In *NeurIPS*, 2022.

[Wu *et al.*, 2023a] Qitian Wu, Chenxiao Yang, Wentao Zhao, Yixuan He, David Wipf, and Junchi Yan. DIFFormer: Scalable (graph) transformers induced by energy constrained diffusion. In *ICLR*, 2023.

[Wu *et al.*, 2023b] Yi Wu, Yanyang Xu, Wenhao Zhu, Guojie Song, Zhouchen Lin, Liang Wang, and Shaoguo Liu. Kdlgt: A linear graph transformer framework via kernel decomposition approach. In *IJCAI*, 2023.

[Xu *et al.*, 2019a] Chengfeng Xu, Pengpeng Zhao, Yanchi Liu, Victor S. Sheng, Jiajie Xu, Fuzhen Zhuang, Junhua Fang, and Xiaofang Zhou. Graph contextualized self-attention network for session-based recommendation. In *IJCAI*, 2019.

[Xu *et al.*, 2019b] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *ICLR*, 2019.

[Yin and Zhong, 2023] Shuo Yin and Guoqiang Zhong. Lgi-gt: Graph transformers with local and global operators interleaving. In *IJCAI*, 2023.

[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 badly for graph representation? In *NeurIPS*, 2021.

[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:2110.13094*, 2021.

[Zhao *et al.*, 2023] Haiteng Zhao, Shuming Ma, Dongdong Zhang, Zhi-Hong Deng, and Furu Wei. Are more layers beneficial to graph transformers? In *ICLR*, 2023.

[Zhu *et al.*, 2021] Yiran Zhu, Xing Xu, Fumin Shen, Yanli Ji, Lianli Gao, and Heng Tao Shen. Posegtac: Graph transformer encoder-decoder with atrous convolution for 3d human pose estimation. In *IJCAI*, 2021.