ENHANCING GRAPH TRANSFORMERS WITH SPECTRAL GUIDANCE IN ATTENTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Existing Graph Transformers often overlook the limitations of self-attention mechanism without inductive bias. The pure self-attention tends to aggregate features from unrelated nodes and misalign attention with graph structures, leading to suboptimal modeling of relational dependencies. Moreover, operating solely in the spatial domain, self-attention underutilizes graph spectral components that correspond to more detailed and comprehensive relational patterns. To address the above issues, we propose the Spectral-Guided Attention Graph Transformer (SGA-Former), which introduces rich structural priors from the graph spectral domain to guide attention learning. Specifically, we design two Spectral Relation Metrics as attention bias, which capture complementary low and high-frequency structural patterns. To leverage these priors, we develop the Spectral-Guided Attention Enhancer (SGA-Enhancer), which filters redundant attention scores and emphasizes important node relationships based on the spectral metrics. Incorporating SGA-Enhancer, SGA-Former builds dual-branch Spectral Attention Layers that jointly utilize both spectral views, enabling more balanced and structure-aware attention learning. Extensive experiments show that SGA-Former consistently achieves superior performance across a wide range of graph learning tasks.

1 INTRODUCTION

Transformers (Vaswani et al., 2017) have achieved significant success in modalities like natural language (Vaswani et al., 2017) and vision (Neil and Dirk, 2020), inspiring growing interest in adapting them for graph data to address limitations of Graph Neural Networks (Khemani et al., 2024), such as over-smoothing (Li et al., 2018), over-squashing (Oono and Suzuki, 2019), and limited expressive power (Morris et al., 2019). Graph Transformers (Min et al., 2022) aim to overcome the above issues by leveraging learned attention mechanisms instead of strictly following the input graph topology. However, the pure Transformer architecture inherently lacks strong inductive biases (Neil and Dirk, 2020). Therefore, incorporating graph inductive biases becomes central to adapting Transformers to graph data, enabling perception and utilization of the underlying graph structure.

There are two mainstream approaches to introducing graph inductive biases into Transformer-based architectures. The first approach embeds GNN modules in series or parallel with attention layers, to assist in modeling the graph structure (Rampášek et al., 2022; Chen et al., 2022; Zhang et al., 2024). The second injects positional or structural encodings into node or edge features, enabling implicit modeling of graph topology (Ma et al., 2023; Dwivedi et al., 2021; Ying et al., 2021). Although structural cues are introduced in above Transformer-based studies, the self-attention mechanism still suffers from inherent limitations due to the absence of inductive bias. Specifically, global attention tends to aggregate features from unrelated nodes, leading to noisy attention distributions (Xing et al., 2024), while the absence of explicit grounding in actual connection strengths reduces the model sensitivity to graph-dependent relational structures (Liu et al., 2024; Zhuang et al., 2025; Zhao et al., 2023).

Some studies begin to explore reducing redundant attention and guiding attention to focus on meaningful node relationships. Exphormer (Shirzad et al., 2023) is the first to sparsify attention in Graph Transformers by approximating the full graph structure, demonstrating that modeling all pairwise node relationships is unnecessary. Gradformer (Liu et al., 2024) and MSA-GT (Zhuang et al., 2025) introduce distance-based attention bias (Shehzad et al., 2024) to explicitly guide attention across

multiple levels of structural information. However, by relying on coarse approximations or local spatial metrics, existing approaches fail to capture critical global structural patterns and struggle to model complex relational dependencies. To address the above limitations, we turn to the spectral domain, where different frequency bands exhibit distinctive advantages in encoding diverse global structural patterns (Sandryhaila and Moura, 2014; Ortega et al., 2018). We design complementary low-pass and high-pass spectral filters to generate attention biases and visualize their effects in comparison with existing methods on molecular and community graphs. As illustrated in Fig.1 and Fig.2, our approach extracts more informed relational patterns via frequency-specific filtering, thereby better guiding attention with enhanced global structural awareness. To this end, we propose the first approach that leverages spectral-domain priors as inductive bias to explicitly guide selective attention learning in Graph Transformers.

In this work, we propose the Spectral-Guided Attention Graph Transformer (SGA-Former), a novel architecture that explicitly integrates spectral guidance into graph attention. Specifically, we introduce two Spectral Relation Metrics, $\mathbf{M}_{\text{low}}$ and $\mathbf{M}_{\text{high}}$, derived from graph signal filtering theory. These metrics capture complementary structural patterns: $\mathbf{M}_{\text{low}}$ encodes global smoothness and structural coherence (low-frequency components), while $\mathbf{M}_{\text{high}}$ emphasizes local variations and sharpness (high-frequency components). We further provide spatial-domain interpretations, offering an alternative message-passing perspective for understanding these metrics. Building upon these spectral priors, we design the Spectral-Guided Attention Enhancer (SGA-Enhancer), which refines attention learning by filtering redundant attention scores and highlighting structurally important node relationships. Incorporating SGA-Enhancer, SGA-Former employs a dual-branch Spectral-Guided Attention Layer (SGA-Layer) that simultaneously leverages low- and high-frequency structural priors. This design promotes more balanced, structure-aware attention learning, leading to improved performance and generalization across diverse graph tasks. From a theoretical perspective, we further show that SGA-Former with Spectral Relation Metrics achieve stronger expressive power compared to the commonly used SPD (shortest path distance) metric. The contributions of this paper are:

- We propose SGA-Former, a novel graph Transformer that first leverages spectral priors to explicitly facilitate selective attention learning. Guided by spectral structural cues, SGA-Former achieves structure-aware and spectrally-informed representation learning.
- We design two Spectral Relation Metrics, $\mathbf{M}_{\text{low}}$ and $\mathbf{M}_{\text{high}}$ to effectively capture complementary structural patterns across different frequency bands, with both both spectral and spatial interpretations.
- We propose SGA-Enhancer, which selectively filters and reweights attention scores based on spectral priors. By integrating SGA-Enhancer into a dual-branch layer architecture, SGA-Former achieves improved structural modeling and superior performance across diverse graph learning tasks. Theoretical analysis is also provided to demenstrate the strong expressiveness of SGA-Former.

2 PRELIMINARIES

Graph Transformer: The computation of a Graph Transformer (GT) layer, similar to that of the standard Transformer, can be decomposed into two fundamental components: the multi-head self-attention (MHA) mechanism and the feed-forward network (FFN). Given the input node features $\mathbf{H}$ and the edge feature $\mathbf{E}$, each attention head computes representations by attending to all node features. A single attention head is formulated as

$$\text{Attn}(\mathbf{H}, \mathbf{E}) = \text{Softmax}(\text{Score}(\mathbf{H}, \mathbf{E})) \cdot \text{V}(\mathbf{H}), \tag{1}$$

where $\text{Score}(\cdot)$ denotes a function that computes the raw attention scores, and $\text{V}(\cdot)$ is a linear projection of node features. To better perceive graph structure, we adopt the attention mechanism and feature encoding strategy from GRIT (Ma et al., 2023). GRIT updates edge features during attention computation, allowing structural information to be implicitly integrated. The specific calculation process of the GRIT attention head is provided in the Appendix B.2. The multi-head self-attention (MHA) then aggregates the outputs from multiple attention heads by

$$\text{MHA}(\mathbf{H}, \mathbf{E}) = \text{Concat}\left(\text{Attn}_1(\mathbf{H}, \mathbf{E}), \ldots, \text{Attn}_K(\mathbf{H}, \mathbf{E})\right) \mathbf{W}_O, \tag{2}$$

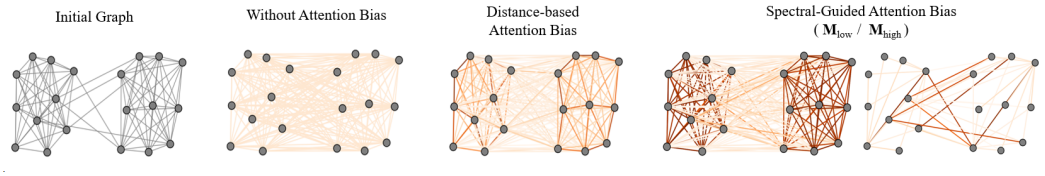


Figure 1: Visualization comparison of different attention biases in a stochastic block model with 2 communities. Thicker and darker edges indicate higher edge weight. The *Without Attention Bias* refers to most approaches without explicitly injected selective attention mechanisms. The *Distance-based Attention Biased* using $1/\text{dist}(i, j)$ as edge weight between node i and j. In the proposed *Spectral-Guided Attention Bias*, $\mathbf{M}_{\text{low}}$ reflects connectivity patterns within communities, while $\mathbf{M}_{\text{high}}$ emphasizes cross-community links that highlight structural boundaries.

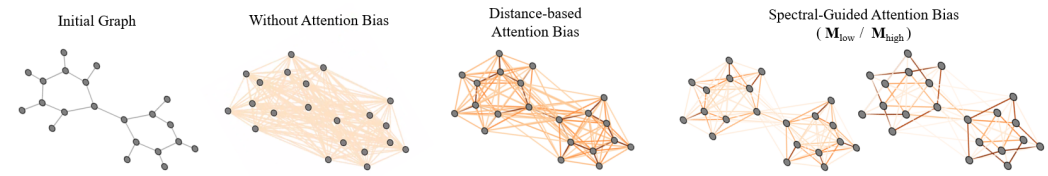


Figure 2: Visualization comparison of different attention biases in the 2-Phenylpyridine module. In the proposed *Spectral-Guided Attention Bias*, $\mathbf{M}_{\text{low}}$ captures more global atomic relationships within functional groups, while $\mathbf{M}_{\text{high}}$ reveals higher-order interactions (eg. as star-like patterns).

where K is the number of heads and $\mathbf{W}_O$ is an output projection matrix. The output is then passed through a feed-forward network, followed by a normalization function to compute the updated node features $\hat{\mathbf{H}}$ by

$$\hat{\mathbf{H}} = \text{Norm}(\text{FFN}(\text{MHA}(\mathbf{H}, \mathbf{E}))). \tag{3}$$

Graph Fourier Transform: Spectral variation of graphs is grounded in the frequency-domain decomposition of the graph Laplacian matrix, which reflects the spatial-domain graph topology. In this work, we adopt the *symmetrically normalized Laplacian* (Boukrab and Pagès-Zamora, 2021), defined as $\mathbf{L}_{\text{sym}} = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}$, where $\mathbf{A}$ is the adjacency matrix and $\mathbf{D}$ is the degree matrix. Since $\mathbf{L}_{\text{sym}}$ is a positive semi-definite matrix, it can be decomposed via eigendecomposition as $\mathbf{L}_{\text{sym}} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^\top$, where $\mathbf{U}$ contains the eigenvectors (graph Fourier bases), and $\mathbf{\Lambda}$ is a diagonal matrix of eigenvalues (corresponding frequencies).

Spectral Prior as Inductive Bias: Different frequency components in the graph spectral domain naturally encode distinct structural properties (Bo et al., 2021; Sandryhaila and Moura, 2014)). Specifically, the low-frequency components correspond to smaller eigenvalues and capture smooth variations over the graph. These components encode global structures, such as clusters or communities. In contrast, high-frequency components are associated with larger eigenvalues and reflect rapid changes in signal values across adjacent nodes. These components highlight local differences, such as structural boundaries and irregular connections. Motivated by the above perspective, we design two Spectral Relation Metrics as attention bias, which act as a bridge between the rich structural patterns encoded in the spectral domain and the guidance of graph attention learning.

3 Methodology

3.1 Spectral Relation Metrics Construction

Spectral Relation Metrics: Exploiting the complementarity and representativeness of spectral components, we design two filters to amplify low and high-frequency signals in the graph Laplacian for deriving Spectral Relation Metrics. For enhancing the low and high value frequencies respectively, the two filter functions are constructed by

$$\text{filter}_{\text{low}}(\mathbf{\Lambda}) = \frac{1}{k}\sum_{i=1}^{k}(\mathbf{I} - \mathbf{\Lambda})^i \tag{4}$$

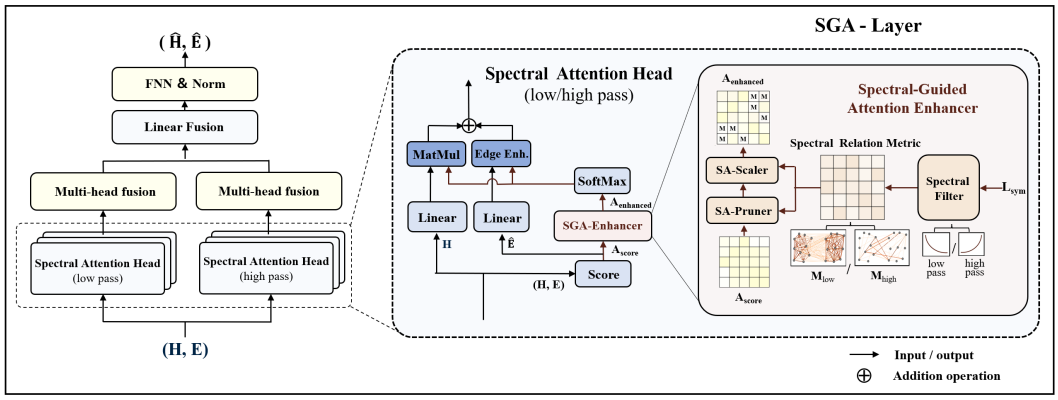


Figure 3: The architecture of the SGA-Former layer.

and

$$\text{filter}_{\text{high}}(\mathbf{\Lambda}) = \frac{1}{k}\sum_{i=1}^{k}(\mathbf{\Lambda} - \mathbf{I})^i, \tag{5}$$

where k is a hyperparameter. As shown in the Fig. 5, given that the eigenvalues in $\mathbf{\Lambda}$ are within the range $[0, 2]$, $\text{filter}_{\text{low}}(\cdot)$ emphasizes smaller eigenvalues while attenuating larger ones, whereas $\text{filter}_{\text{high}}(\cdot)$ does the opposite by reserving larger eigenvalues and suppressing the lower ones.

Subsequently, by applying filtering functions to the Laplacian matrix, we obtain modified graph structures that reflect the strength of node connections associated with specific frequency bands. Therefore, we call the obtained matrices as Spectral Relation Metrics, and denote them as $\mathbf{M}_{\text{high}}$ and $\mathbf{M}_{\text{low}}$. $\mathbf{M}_{\text{high}}$ and $\mathbf{M}_{\text{low}}$ can be calculated as

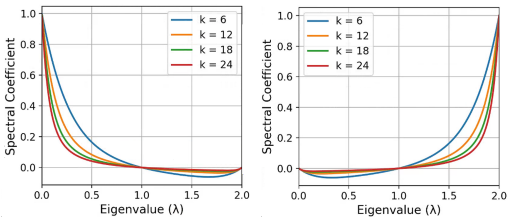
$$\begin{aligned}\mathbf{M}_{\text{low}} = \mathbf{U}\text{filter}_{\text{low}}(\mathbf{\Lambda})\mathbf{U}^\top &= \frac{1}{k}\sum_{i=1}^{k}\mathbf{U}(\mathbf{I} - \mathbf{\Lambda})^i\mathbf{U}^\top \\ &= \frac{1}{k}\sum_{i=1}^{k}(\mathbf{I} - \mathbf{L}_{\text{sym}})^i = \frac{1}{k}\sum_{i=1}^{k}(\mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2})^i\end{aligned} \tag{6}$$

and

$$\begin{aligned}\mathbf{M}_{\text{high}} = \mathbf{U}\text{filter}_{\text{high}}(\mathbf{\Lambda})\mathbf{U}^\top &= \frac{1}{k}\sum_{i=1}^{k}\mathbf{U}(\mathbf{\Lambda} - \mathbf{I})^i\mathbf{U}^\top \\ &= \frac{1}{k}\sum_{i=1}^{k}(\mathbf{L}_{\text{sym}} - \mathbf{I})^i = \frac{1}{k}\sum_{i=1}^{k}(-\mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2})^i\end{aligned} \tag{7}$$

Notably, although computing the frequency components of $\mathbf{L}_{\text{sym}}$ requires costly eigendecomposition, our specifically designed filtering functions enable efficient computation using simple matrix operations on the existing adjacency matrix $\mathbf{A}$ and degree matrix $\mathbf{D}$. The two Spectral Relation Metrics offer complementary insights into the structural properties of the graph. Specifically, $\mathbf{M}_{\text{low}}$ captures smooth and globally consistent relationships by emphasizing low-frequency components, which tend to reflect shared cluster or community structures. In contrast, $\mathbf{M}_{\text{high}}$ emphasizes sharp local variations and highlights structural boundaries or anomalies through high-frequency enhancement. Together, they provide a spectral-aware perspective for node relations from comprehensive viewpoints, enabling more informative modeling of relational dependencies.

Graph Spatial Domain Analysis: In the graph spatial domain, $\mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}$ can be viewed as a message propagation matrix (Kipf, 2016; Wu et al., 2019), where $\mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}\mathbf{A}\mathbf{H}$ represents a single round of message passing across neighboring nodes. Consequently, $(\mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2})^k$ corresponds to the effect of k-step message propagation. From this perspective, each entry $\mathbf{M}_{\text{low},ij}$ in $\mathbf{M}_{\text{low}}$ can be interpreted as the average message propagation strength from node i to node j



(a) Low-pass (b) High-pass

Figure 5: (a) Effect of $\text{filter}_{\text{low}}(\cdot)$, (b) Effect of $\text{filter}_{\text{high}}(\cdot)$.

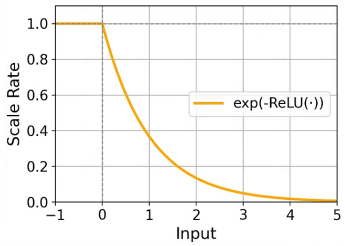


Figure 6: Effect of the scaling operators.

aggregated over propagation steps from 1 to k, capturing the smooth multi-hop diffusion process in the graph (Gasteiger et al., 2019). In contrast, $\mathbf{M}_{\text{high}}$ introduces alternating signs through the term $(-\mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2})^k$, which leads to the suppression of contributions from odd-order propagation paths. These paths are closely related to the message propagation in dense structures such as clusters or communities, and thus, the non-smooth, abrupt structural patterns in the graph are accentuated. As shown in Fig. 1 and Fig. 2 , we visualize the edge weight of the two metrics ($k = 8$) in the stochastic block model with 2 communities and 2-Phenylpyridine module. Across different graph structures, both $\mathbf{M}_{\text{low}}$ and $\mathbf{M}_{\text{high}}$ consistently provide complementary spatial connectivity views and capture more informative relational patterns. These spatial effects demonstrate that the two Spectral Attention Metrics can serve as strong inductive biases for guiding graph attention learning.

Expressness of Spectral Relation Metrics: Recent works (Liu et al., 2024; Zhuang et al., 2025) introduce distance-based attention biases to explicitly guide the attention mechanism. Motivated by this, we evaluate the superior expressiveness of Spectral Relation Metrics over SPD (shortest path distance) using recently proposed graph isomorphism tests inspired by the Weisfeiler-Leman algorithm (Wang et al., 2020). Specifically, Zhang et al. (2023) introduced the Generalized Distance Weisfeiler-Leman (GD-WL) test and applied it to analyze a graph transformer architecture that uses SPD(i, j) as relative positional encodings. They theoretically established that the maximal expressive power of such a graph transformer is upper-bounded by the GD-WL test with SPD. Here, we also use the GD-WL test to showcase the expressiveness of the Spectral Relation Metrics.

Proposition 1. *GD-WL with* $\mathbf{M}$ *is strictly more expressive than GD-WL with the shortest path distance SPD, provided all individual terms in the sums defining* $\mathbf{M}$ *are accessible and* $k > \text{diam}(G)$ *(diameter of Graph* G*).*

The proof is provided in Appendix A.1. Firstly, we show that the GD-WL test using Spectral Relation Metric $\mathbf{M}$ can differentiate between any two graphs that can be distinguished by the GD-WL test with SPD. Next, we show that the GD-WL test with $\mathbf{M}$ is capable of the Dodecahedron and Desargues graphs while the one with SPD cannot.

3.2 Spectral-Guided Attention Enhancer

Based on the designed Spectral relation Metrics, we propose a Spectral-Guided Attention Enhancer (SGA-Enhancer), which aims to prune and scale the attention matrix according to the spectral priors. To this end, SGA-Enhancer is composed of two components in series: Spectral-Aware Attention Pruner (SA-Pruner) and Spectral-Aware Attention Scaler (SA-Scaler). Overall, the operation performed by the SGA-Enhancer can be formally defined as

$$\mathbf{A}_{\text{enhanced}} = \text{SGA-Enhancer}(\mathbf{A}_{\text{score}}, \mathbf{M}, \alpha) = \text{SA-Scaler}\Big(\text{SA-Pruner}(\mathbf{M}, \mathbf{A}_{\text{score}}, \alpha), \mathbf{M}\Big). \tag{8}$$

In this formulation, $\mathbf{M}$ denotes a generic spectral relation metric matrix derived from any frequency domain (e.g., $\mathbf{M}_{\text{low}}$ or $\mathbf{M}_{\text{high}}$); $\mathbf{A}_{\text{score}}$ and $\mathbf{A}_{\text{enhanced}}$ denotes the raw attention score matrix output by the $\text{Score}(\cdot)$ function and the enhanced attention scores refined by SGA-Enhancer; α is a hyperparameter. In the following, we provide a detailed description of the two constituent modules of SGA-Enhancer.

Spectral-Aware Attention Pruner: Based on the node relation metrics constructed in the two spectral domains, more important attention components can be easily identified, allowing the removal of redundant relational learning. Therefore, for each metric, we can construct a Spectral-Aware Attention Pruner as

$$\begin{aligned} &\text{SA-Pruner}(\mathbf{M}, \mathbf{A}_{\text{score}}, \alpha) = \mathbf{A}_{\text{score}} + \text{Mask}_{\mathbf{M}}, \\ &\text{where } \text{Mask}_{\mathbf{M},ij} = \begin{cases} 0, & \text{if } \mathbf{M}_{ij} \text{ is among the top} \\ & \quad \alpha\text{-fraction entries of } \mathbf{M}, \\ -\infty, & \text{otherwise.} \end{cases} \end{aligned} \tag{9}$$

In this formulation, $\text{Mask}_{\mathbf{M}} \in \mathbf{R}^{N \times N}$ is an attention mask of the same dimension as $\mathbf{A}_{\text{score}}$. The hyperparameter $\alpha \in [0, 1]$ specifies the proportion of node pairs with the highest values in $\mathbf{M}$ to be retained. α can be independently configured for each spectral metric to selectively preserve the attention corresponding to the most informative relational signals in the respective frequency domain. The resulting masked attention score matrix is denoted as $\mathbf{A}_{\text{mask}}$.

Spectral-Aware Attention Scaler: To further enhance the guidance provided by spectral information, we aim to adaptively modulate the strength of attention scores based on the Spectral Relation Metrics. To achieve this, we design a Spectral-Aware Attention Scaler as

$$\text{SA-Scaler}(\mathbf{A}_{\text{mask}}, \mathbf{M}) = \mathbf{A}_{\text{mask}} \odot f(\mathbf{M}), \tag{10}$$

where $\odot$ denotes element-wise multiplication, and $f(\cdot)$ is a learnable, parameterized function. Specifically, we implement $f(\mathbf{M})$ as:

$$f(\mathbf{M}) = \exp\left(-\text{ReLU}(a\mathbf{M} + b)\right). \tag{11}$$

where $\exp(-\text{ReLU}(\cdot)$ acts as a scaling operator, mapping input values to the range $[0, 1]$. It preserves values near 1 while progressively attenuating others toward 0. In Fig. 6, we illustrate how inputs are mapped to this range. The learnable scalars a and b control the rate and threshold of this attenuation, enabling the model to adaptively modulate the attention intensity.

3.3 Spectral-Guided Attention Graph Transformer

Multi-head Spectral Attention: The SGA-Enhancer can be seamlessly integrated into attention head to improve the modeling of attention over graph structures. The integration only requires replacing the raw attention scores with the ones refined by SGA-Enhancer before applying the softmax operation, and can be formulated by

$$\mathbf{A}_{\mathbf{M}} = \text{Softmax}\Big(\text{SGA-Enhancer}(\text{Score}(\mathbf{H}, \mathbf{E}), \mathbf{M}, \alpha)\Big), \tag{12}$$

where $\mathbf{A}_{\mathbf{M}}$ denotes the final attention scores enhanced by a specific Spectral Relation Metric $\mathbf{M}$. By denoting the attention head function introduce SGA-Enhancer as $\text{Attn}_{\mathbf{M}}$, the refined multi-head self-attention module can be defined as

$$\text{MHA}_{\mathbf{M}}(\mathbf{H}, \mathbf{E}) = \text{Concat}\Big(\text{Attn}_{\mathbf{M},1}(\mathbf{H}, \mathbf{E}), \ldots, \text{Attn}_{\mathbf{M},K}(\mathbf{H}, \mathbf{E})\Big)\mathbf{W}_O. \tag{13}$$

For each Spectral Relation Metric, we construct a dedicated multi-head self-attention module to capture graph structural information from different spectral components. The two modules are denoted as $\text{MHA}_{\mathbf{M}_{\text{low}}}(\mathbf{H}, \mathbf{E})$ and $\text{MHA}_{\mathbf{M}_{\text{high}}}(\mathbf{H}, \mathbf{E})$.

Spectral-Guided Graph Attention Layer: With the integration of multi-head spectral attention, the complete Spectral-Guided Graph Attention Layer (SGA-Layer) is defined as

$$\hat{\mathbf{H}} = \text{Norm}\big(\text{FFN}\big((1-\beta) \cdot \text{MHA}_{\mathbf{M}_{\text{low}}}(\mathbf{H}, \mathbf{E}) + \beta \cdot \text{MHA}_{\mathbf{M}_{\text{high}}}(\mathbf{H}, \mathbf{E})\big)\big), \tag{14}$$

where β is a learnable parameter restricted in $[0, 1]$ that dynamically balances the contributions from low-frequency and high-frequency attention modules. The overall architecture, SGA-Former, is built by stacking multiple SGA-Layers as shown in Fig. 3. Notably, considering a single spectral branch of the SGA-Former, which is also included in the ablation study, we obtain the following results regarding its expressiveness.

Proposition 2. *The power of a graph transformer with* $\mathbf{M}$ *to distinguish non-isomorphic graphs is at most equivalent to that of the GD-WL test with* $\mathbf{M}$*, assuming that additional edge features are ignored. With appropriately chosen parameters, and a sufficient number of attention heads and layers, a graph transformer with* $\mathbf{M}$ *can match the expressiveness of the GD-WL test with* $\mathbf{M}$*.*

The proof of this proposition can be found in Appendix A.2. This result provides a precise characterization of the expressiveness and limitations of graph transformers utilizing $\mathbf{M}$. By combining Proposition 1, Proposition 2, and the proofs in Ying et al. (2021) (Appendix A.1), we immediately obtain the following corollary:

Corollary 1. ***(Expressiveness of Graph Transformers with*** $\mathbf{M}$***).*** *There exists a graph transformer* T*, using* $\mathbf{M}$ *with fixed parameters, that is more expressive than graph transformers of the same architecture that use SPD or do not incorporate any relative positional encoding, regardless of their parameters.*

This result provides a fine-grained analysis of the expressiveness of graph transformers with $\mathbf{M}$, demonstrating SGA-Former with $\mathbf{M}$ outperforms using SPD in terms of expressiveness.

4 Experiments

4.1 Benchmarking SGA-Former

Datasets: We evaluate the effectiveness of our method across five benchmarks from the Benchmarking GNNs work (Dwivedi et al., 2023) and two from the Long-Range Graph Benchmark (Dwivedi et al., 2022). These datasets span a wide range of graph learning tasks—including node and graph classification, as well as graph regression. Additionally, we also evaluate our method on a large-scale dataset ZINC-full (250K graphs) (Irwin et al., 2012).

Baselines: We compare our method with a range of state-of-the-art graph learning models. These methods include classical Graph Neural Networks such as GCN (Kipf, 2016), GIN (Xu et al., 2018), GAT (Veličković et al., 2017), GatedGCN (Bresson and Laurent, 2017), GatedGCN-LSPE (Dwivedi et al., 2021), and PNA (Corso et al., 2020); Graph Transformers including GRIT (Ma et al., 2023), GraphGPS (Rampášek et al., 2022), Graphormer (Ying et al., 2021), K-Subgraph SAT (Chen et al., 2022), EGT (Hussain et al., 2022), SAN (Kreuzer et al., 2021), Graphormer-UPRE (Luo et al., 2022), Graphormer-GD (Zhang et al., 2023); and recent GNN variants with strong performance such as DGN (Beaini et al., 2021), GSN (Bouritsas et al., 2022), CRaW1 (Toenshoff et al., 2021), and GIN-AK+ (Zhao et al., 2021). Additionally, we consider recent attention-guiding optimization approaches, including Exphormer (Shirzad et al., 2023), GradFormer (Liu et al., 2024) and MSA-GT (Zhuang et al., 2025).

Benchmarks from Benchmarking GNNs: We comprehensively evaluate the proposed SGA-Former on five benchmark datasets from Benchmarking GNNs (Dwivedi, Joshi, et al. 2023), including ZINC, MNIST, CIFAR10, PATTERN, and CLUSTER. As reported in Table 1, SGA-Former achieves the highest average performance on all datasets except ZINC, outperforming both MPNN-based models and Graph Transformer baselines, including several state-of-the-art attention enhancement approaches. These results demonstrate the effectiveness and strong generalization ability of SGA-Former across diverse graph learning tasks, largely attributed to its spectral-guided attention mechanism.

Long Range Graph Benchmarks: Next, we evaluate our method on the Long-Range Graph Benchmark (LRGB) (Dwivedi et al., 2022), which is designed to assess the ability of graph models to capture long-range dependencies. Specifically, we conduct experiments on two peptide-related datasets from LRGB: Peptides-func (a 10-task multilabel classification task) and Peptides-struct (an 11-task regression task). As summarized in Table 2, our model consistently achieves the highest average performance on both benchmarks, surpassing leading MPNN-based methods and Graph Transformer variants. These results demonstrate the effectiveness of our approach in modeling long-range interactions within complex molecular graphs.

ZINC-full Dataset: We further evaluate our model on the large-scale ZINC-full dataset (Irwin et al., 2012), which contains 250,000 molecular graphs and serves as a challenging benchmark for assessing the scalability of graph neural networks. In addition to MPNN-based and Transformer-based

Table 1: Performance comparison of different models across various benchmarks. In the experiment of each dataset, the best result is bolded and the second best result is underlined.

Model	ZINC	MNIST	CIFAR10	PATTERN	CLUSTER
	MAE↓	Acc↑	Acc↑	Acc↑	Acc↑
GCN	0.367 ± 0.011	90.705 ± 0.218	55.710 ± 0.381	71.892 ± 0.334	68.498 ± 0.976
GIN	0.526 ± 0.051	96.485 ± 0.252	55.255 ± 1.527	85.387 ± 0.136	64.716 ± 1.553
GAT	0.384 ± 0.007	95.535 ± 0.205	64.223 ± 0.455	78.271 ± 0.186	70.587 ± 0.447
GatedGCN	0.282 ± 0.015	97.340 ± 0.143	67.312 ± 0.311	85.568 ± 0.088	73.840 ± 0.326
GatedGCN-LSPE	0.090 ± 0.001	–	–	–	–
PNA	0.188 ± 0.004	97.94 ± 0.12	70.35 ± 0.63	–	–
DGN	0.168 ± 0.003	–	72.838 ± 0.417	86.680 ± 0.034	–
GSN	0.101 ± 0.010	–	–	–	–
CRaW1	0.085 ± 0.004	97.944 ± 0.050	69.013 ± 0.259	–	–
GIN-AK+	0.080 ± 0.001	–	72.19 ± 0.13	86.850 ± 0.057	–
SAN	0.139 ± 0.006	–	–	86.581 ± 0.037	76.691 ± 0.650
Graphormer	0.122 ± 0.006	–	–	–	–
K-Subgraph SAT	0.094 ± 0.008	–	–	86.848 ± 0.037	77.856 ± 0.104
EGT	0.108 ± 0.009	98.173 ± 0.087	68.702 ± 0.409	86.821 ± 0.020	79.232 ± 0.348
Graphormer-URPE	0.086 ± 0.007	–	–	–	–
Graphormer-GD	0.081 ± 0.009	–	–	–	–
GPS	0.070 ± 0.004	98.051 ± 0.126	72.298 ± 0.356	86.685 ± 0.059	78.016 ± 0.180
GRIT	$\mathbf{0.059 \pm 0.002}$	98.108 ± 0.111	$\underline{76.468 \pm 0.881}$	$\underline{87.196 \pm 0.076}$	$\underline{80.026 \pm 0.277}$
Exphormer	–	$\underline{98.550 \pm 0.039}$	74.690 ± 0.125	86.740 ± 0.015	78.070 ± 0.037
GradFormer	$\underline{0.069 \pm 0.002}$	–	–	86.892 ± 0.070	78.550 ± 0.016
MSA-GT	0.122 ± 0.001	98.405 ± 0.128	72.971 ± 0.331	86.732 ± 0.044	78.578 ± 0.041
SGA-Former (ours)	$\mathbf{0.059 \pm 0.002}$	$\mathbf{98.580 \pm 0.130}$	$\mathbf{77.43 \pm 0.301}$	$\mathbf{87.252 \pm 0.066}$	$\mathbf{80.102 \pm 0.242}$

Table 2: Test performance on two benchmarks from *Long-range Graph Benchmarks* (LRGB). Best results are bold, second best underlined (mean ± s.d. of 4 runs).

Model	Peptides-func (AP↑)	Peptides-struct (MAE↓)
GCN	0.5930 ± 0.0023	0.3496 ± 0.0013
GINE	0.5498 ± 0.0079	0.3547 ± 0.0045
GatedGCN+RWSE	0.6069 ± 0.0035	0.3357 ± 0.0006
CKGCN	0.6952 ± 0.0068	0.2477 ± 0.0018
Transformer+LapPE	0.6326 ± 0.0126	0.2529 ± 0.0016
SAN+LapPE	0.6384 ± 0.0121	0.2683 ± 0.0043
SAN+RWSE	0.6439 ± 0.0075	0.2545 ± 0.0012
GPS	0.6535 ± 0.0041	0.2500 ± 0.0012
Exphormer	0.6527 ± 0.0034	0.2481 ± 0.0007
GRIT	0.6988 ± 0.0082	<u>0.2460 ± 0.0012</u>
MSA-GT	<u>0.6605 ± 0.0180</u>	0.2470 ± 0.0015
SGA-Former (ours)	**0.7070 ± 0.0262**	**0.2430 ± 0.0011**

Table 3: Test performance on ZINC-full. Best results are bold, second best underlined (mean ± s.d. over 4 runs.)

Method	Model	ZINC-full (MAE ↓)
GNN	GIN	0.088 ± 0.002
	GraphSAGE	0.126 ± 0.003
	GAT	0.111 ± 0.002
	GCN	0.113 ± 0.002
	MoNet	0.090 ± 0.002
	SignNet	0.024 ± 0.003
Graph Transformers	Graphormer	0.052 ± 0.005
	Graphormer-URPE	0.028 ± 0.002
	Graphormer-GD	0.025 ± 0.004
	GRIT	<u>0.023 ± 0.001</u>
	MSA-GT	0.025 ± 0.002
	SGA-Former (ours)	$\mathbf{0.020 \pm 0.002}$

GNNs, the comparison also includes position-enhanced models such as SignNet (Lim et al., 2022). As shown in Table 3, SGA-Former achieves the best mean performance, consistently outperforming all baseline models across different architectural paradigms.

4.2 Ablation and Sensitivity Analysis

Ablation Study: Table 4 reports ablation results on MNIST and CIFAR10, highlighting the effectiveness of each component in SGA-Former. Removing the SGA-Enhancer significantly reduces performance, confirming its core role. Both SA-Pruner and SA-Scaler contribute to performance improvement. Additionally, removing either the low-pass or high-pass branch leads to performance drops, indicating that both frequency-guided branches are beneficial and complementary.

Attention Keeping Rate α: We conduct a sensitivity analysis on the attention keeping rate α in SA-Pruner, where the model operates with only a single branch (either low-pass or high-pass) active, as shown in Table 5. The results demonstrate that the model consistently achieves strong performance across a broad range of α values on both MNIST and CIFAR, despite being restricted to a single branch. For the low-pass branch, accuracy gener-

ally increases with higher α, peaking around 0.80–0.90. In contrast, the high-pass branch remains relatively stable, with only minor fluctuations across different α values. Notably, extremely low α values (e.g., 0.05) result in slight performance drops, indicating that retaining too few attention scores may hinder the model's ability to capture meaningful features.

Spectral Filter Hyperparameter k: We investigate the sensitivity of the spectral filter hyperparameter k. As shown in Table 6, the performance remains consistently strong across a broad range of k values on both MNIST and CIFAR. Extremely small values of k (e.g., $k = 2$) lead to noticeable performance degradation, indicating that insufficient frequency components may hinder expressive capacity. As shown in Figure 5, increasing k sharpens both low-pass and high-pass filters, enhancing their frequency selectivity. This improved filtering effect helps stabilize model performance at larger k values.

Table 4: Ablations on MNIST and CIFAR10, showing Acc and change relative to SGA-Former. (mean ± s.d. of 4 runs)

	MNIST (Acc↑)	**CIFAR10 (Acc↑)**
SGA-Former (ours)	**98.58 ± 0.130**	**77.43 ± 0.301**
- Remove SGA-Enhancer	98.11 ± 0.111	76.47 ± 0.281
- Remove SA-Pruner	98.23 ± 0.121	77.01 ± 0.321
- Remove SA-Scaler	98.38 ± 0.091	76.88 ± 0.295
- Remove low pass branch	98.13 ± 0.132	76.51 ± 0.326
- Remove high pass branch	98.34 ± 0.109	77.01 ± 0.321

Table 5: Performance across different attention keeping rate for single-branch configurations.

	α	0.05	0.20	0.40	0.60	0.80	0.90	1.0
(low)	MNIST	97.76	98.25	98.24	98.33	98.27	98.30	98.10
	CIFAR	76.12	76.36	76.37	76.33	76.71	76.96	76.66
(high)	MNIST	97.96	98.05	98.04	98.13	98.10	98.14	98.01
	CIFAR	75.88	76.37	76.15	76.47	75.95	76.02	75.80

Table 6: Performance comparison under different spectral filter hyperparameter k.

k	2	4	6	8	16	24	32
MNIST	98.10	98.37	98.54	98.49	98.49	98.55	98.54
CIFAR	76.20	76.85	77.42	77.44	77.62	77.35	77.37

5 Conclusion

In this work, we propose SGA-Former, a novel Graph Transformer architecture that first introduces spectral inductive biases to guide attention learning in graphs. Grounded in spectral graph theory, we design two complementary relation metrics, enabling the model to effectively perceive both low and high-frequency topological patterns. To leverage these spectral priors, we propose the Spectral-Guided Attention Enhancer (SGA-Enhancer) as a core component, which combines a pruning module and an adaptive scaling module to refine attention weights based on structural relevance. Beyond its theoretically stronger expressive power, extensive experiments on multiple graph benchmarks show that SGA-Former consistently outperforms existing baselines. In the future, further work can focus on designing more diverse or learnable spectral filters for richer structural modeling and developing adaptive pruning strategies to dynamically control attention sparsity and improve efficiency.

References

Beaini, D., Passaro, S., Létourneau, V., Hamilton, W., Corso, G., and Liò, P. (2021). Directional graph networks. In *International Conference on Machine Learning*, pages 748–758. PMLR.

Bo, D., Shi, C., Wang, L., and Liao, R. (2023). Specformer: Spectral graph neural networks meet transformers. *arXiv preprint arXiv:2303.01028*.

Bo, D., Wang, X., Shi, C., and Shen, H. (2021). Beyond low-frequency information in graph convolutional networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 3950–3957.

Boukrab, R. and Pagès-Zamora, A. (2021). Random-walk laplacian for frequency analysis in periodic graphs. *Sensors*, 21(4):1275.

Bouritsas, G., Frasca, F., Zafeiriou, S., and Bronstein, M. M. (2022). Improving graph neural network expressivity via subgraph isomorphism counting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(1):657–668.

Bresson, X. and Laurent, T. (2017). Residual gated graph convnets. *arXiv preprint arXiv:1711.07553*.

Chen, D., O'Bray, L., and Borgwardt, K. (2022). Structure-aware transformer for graph representation learning. In *International conference on machine learning*, pages 3469–3489. PMLR.

Corso, G., Cavalleri, L., Beaini, D., Liò, P., and Veličković, P. (2020). Principal neighbourhood aggregation for graph nets. *Advances in neural information processing systems*, 33:13260–13271.

Dwivedi, V. P., Joshi, C. K., Luu, A. T., Laurent, T., Bengio, Y., and Bresson, X. (2023). Benchmarking graph neural networks. *Journal of Machine Learning Research*, 24(43):1–48.

Dwivedi, V. P., Luu, A. T., Laurent, T., Bengio, Y., and Bresson, X. (2021). Graph neural networks with learnable structural and positional representations. *arXiv preprint arXiv:2110.07875*.

Dwivedi, V. P., Rampášek, L., Galkin, M., Parviz, A., Wolf, G., Luu, A. T., and Beaini, D. (2022). Long range graph benchmark. *Advances in Neural Information Processing Systems*, 35:22326–22340.

Gasteiger, J., Weißenberger, S., and Günnemann, S. (2019). Diffusion improves graph learning. *Advances in neural information processing systems*, 32.

Hussain, M. S., Zaki, M. J., and Subramanian, D. (2022). 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.

Irwin, J. J., Sterling, T., Mysinger, M. M., Bolstad, E. S., and Coleman, R. G. (2012). Zinc: a free tool to discover chemistry for biology. *Journal of chemical information and modeling*, 52(7):1757–1768.

Khemani, B., Patil, S., Kotecha, K., and Tanwar, S. (2024). A review of graph neural networks: concepts, architectures, techniques, challenges, datasets, applications, and future directions. *Journal of Big Data*, 11(1):18.

Kipf, T. (2016). Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Kreuzer, D., Beaini, D., Hamilton, W., Létourneau, V., and Tossou, P. (2021). Rethinking graph transformers with spectral attention. *Advances in Neural Information Processing Systems*, 34:21618–21629.

Li, Q., Han, Z., and Wu, X.-M. (2018). Deeper insights into graph convolutional networks for semi-supervised learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32.

Lim, D., Robinson, J., Zhao, L., Smidt, T., Sra, S., Maron, H., and Jegelka, S. (2022). Sign and basis invariant networks for spectral graph representation learning. *arXiv preprint arXiv:2202.13013*.

Liu, C., Yao, Z., Zhan, Y., Ma, X., Pan, S., and Hu, W. (2024). Gradformer: Graph transformer with exponential decay. *arXiv preprint arXiv:2404.15729*.

Luo, S., Li, S., Zheng, S., Liu, T.-Y., Wang, L., and He, D. (2022). Your transformer may not be as powerful as you expect. *Advances in Neural Information Processing Systems*, 35:4301–4315.

Ma, L., Lin, C., Lim, D., Romero-Soriano, A., Dokania, P. K., Coates, M., Torr, P., and Lim, S.-N. (2023). Graph inductive biases in transformers without message passing. In *International Conference on Machine Learning*, pages 23321–23337. PMLR.

Min, E., Chen, R., Bian, Y., Xu, T., Zhao, K., Huang, W., Zhao, P., Huang, J., Ananiadou, S., and Rong, Y. (2022). Transformer for graphs: An overview from architecture perspective. *arXiv preprint arXiv:2202.08455*.

Morris, C., Ritzert, M., Fey, M., Hamilton, W. L., Lenssen, J. E., Rattan, G., and Grohe, M. (2019). Weisfeiler and leman go neural: Higher-order graph neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 4602–4609.

Neil, H. and Dirk, W. (2020). Transformers for image recognition at scale. *Online: https://ai. googleblog. com/2020/12/transformers-forimage-recognitionat. html*.

Oono, K. and Suzuki, T. (2019). Graph neural networks exponentially lose expressive power for node classification. *arXiv preprint arXiv:1905.10947*.

Ortega, A., Frossard, P., Kovačević, J., Moura, J. M., and Vandergheynst, P. (2018). Graph signal processing: Overview, challenges, and applications. *Proceedings of the IEEE*, 106(5):808–828.

Rampášek, L., Galkin, M., Dwivedi, V. P., Luu, A. T., Wolf, G., and Beaini, D. (2022). Recipe for a general, powerful, scalable graph transformer. *Advances in Neural Information Processing Systems*, 35:14501–14515.

Sandryhaila, A. and Moura, J. M. (2014). Discrete signal processing on graphs: Frequency analysis. *IEEE Transactions on signal processing*, 62(12):3042–3054.

Shehzad, A., Xia, F., Abid, S., Peng, C., Yu, S., Zhang, D., and Verspoor, K. (2024). Graph transformers: A survey. *arXiv preprint arXiv:2407.09777*.

Shirzad, H., Velingker, A., Venkatachalam, B., Sutherland, D. J., and Sinop, A. K. (2023). Exphormer: Sparse transformers for graphs. In *International Conference on Machine Learning*, pages 31613–31632. PMLR.

Toenshoff, J., Ritzert, M., Wolf, H., and Grohe, M. (2021). Graph learning with 1d convolutions on random walks. *arXiv preprint arXiv:2102.08786*.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, 30.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., and Bengio, Y. (2017). Graph attention networks. *arXiv preprint arXiv:1710.10903*.

Wang, S., Li, B. Z., Khabsa, M., Fang, H., and Ma, H. (2020). Linformer: Self-attention with linear complexity. *arXiv preprint arXiv:2006.04768*.

Wu, F., Souza, A., Zhang, T., Fifty, C., Yu, T., and Weinberger, K. (2019). Simplifying graph convolutional networks. In *International conference on machine learning*, pages 6861–6871. Pmlr.

Xing, Y., Wang, X., Li, Y., Huang, H., and Shi, C. (2024). Less is more: on the over-globalizing problem in graph transformers. *arXiv preprint arXiv:2405.01102*.

Xu, K., Hu, W., Leskovec, J., and Jegelka, S. (2018). How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826*.

Ying, C., Cai, T., Luo, S., Zheng, S., Ke, G., He, D., Shen, Y., and Liu, T.-Y. (2021). Do transformers really perform badly for graph representation? *Advances in neural information processing systems*, 34:28877–28888.

Zhang, B., Luo, S., Wang, L., and He, D. (2023). Rethinking the expressive power of gnns via graph biconnectivity. *arXiv preprint arXiv:2301.09505*.

Zhang, P., Yan, Y., Zhang, X., Li, C., Wang, S., Huang, F., and Kim, S. (2024). Transgnn: Harnessing the collaborative power of transformers and graph neural networks for recommender systems. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1285–1295.

Zhao, H., Ma, S., Zhang, D., Deng, Z.-H., and Wei, F. (2023). Are more layers beneficial to graph transformers? *arXiv preprint arXiv:2303.00579*.

Zhao, L., Jin, W., Akoglu, L., and Shah, N. (2021). From stars to subgraphs: Uplifting any gnn with local structure awareness. *arXiv preprint arXiv:2110.03753*.

Zhuang, J., Li, J., Shi, C., Lin, X., and Fu, Y.-G. (2025). Enhanced graph transformer: Multi-scale attention with heterophilous curriculum augmentation. *Knowledge-Based Systems*, 309:112874.

APPENDIX

A METHOD DETAILS

A.1 PROOF OF PROPOSITION 1

Restatement of Proposition 1: GD-WL with $\mathbf{M}$ is strictly more expressive than GD-WL with the shortest path distance SPD, provided all individual terms in the sums defining $\mathbf{M}$ are accessible.
Proof. First, we show that GD-WL with $\mathbf{M}$ is at least as expressive as GD-WL with shortest path distances (SPD). Then, we provide a specific example of two graphs that cannot be distinguished by GD-WL with SPD, but can be distinguished by GD-WL with $\mathbf{M}$.

Let

$$\mathbf{M}^{(t)} = (\pm D^{-1/2} A D^{-1/2})^t$$

denote the individual terms in the sum

$$\mathbf{M} = \sum_{t=1}^{k} \mathbf{M}^{(t)}, \quad t \leq k.$$

Since $k \geq \operatorname{diam}(G)$, for any pair of nodes i, j, we have

$$\operatorname{SPD}(i,j) = \begin{cases} 0, & \text{if } i = j, \\ \min\{t : \mathbf{M}_{i,j}^{(t)} > 0\}, & \text{if } i \neq j. \end{cases}$$

Consequently, $\mathbf{M}$ can be reduced to SPD via the mapping

$$\operatorname{SPD}(i,j) = f(\mathbf{M})_{i,j},$$

ensuring that GD–WL with $\mathbf{M}$ is at least as expressive as GD–WL with SPD.

To demonstrate the strict expressivity advantage of GD-WL with $\mathbf{M}$, we consider two non-isomorphic graphs—the Desargues graph and the Dodecahedral graph. As shown in Figure 6 of Zhang et al. (2023), GD-WL using SPD fails to distinguish these graphs. In contrast, as visualized in Figure 7, the heatmaps of $\mathbf{M}_{\text{low}}$ and $\mathbf{M}_{\text{high}}$ reveal clearly distinct value distributions across the two graphs (k is set to 6). This indicates that in the first round of color refinement in GD-WL test, defined as

$$\chi_G^1(v) := \text{hash}\Big\{\big(\mathbf{M}_{v,u}, \chi_G^0(u)\big) : u \in V\Big\},$$

the two graphs produce different multi-sets of node colors. Consequently, GD-WL with $\mathbf{M}$ successfully differentiates them, demonstrating its superior structural discriminative power compared to SPD-based GD-WL.

A.2 PROOF OF PROPOSITION 2

Restatement of Proposition 2: The power of a graph transformer with $\mathbf{M}$ to distinguish non-isomorphic graphs is at most equivalent to that of the GD-WL test with $\mathbf{M}$, ignoring any additional edge features. With appropriately chosen parameters, and a sufficient number of attention heads and layers, a graph transformer with T can match the expressiveness of the GD-WL test with $\mathbf{M}$.

Proof. The theorem is divided into two parts: the first and second halves. We begin by considering the first half: The power of a graph transformer with $\mathbf{M}$ to distinguish non-isomorphic graphs is at most equivalent to that of the GD-WL test with $\mathbf{M}$, ignoring any additional edge features.

Recall that the GD-WL with $\mathbf{M}$ is straightforward and can be expressed as:

$$\chi_G^t(v) := \text{hash}\left\{\big(\mathbf{M}_{v,u}, \chi_G^{t-1}(u)\big) : u \in V\right\}$$

where $\chi_G^t(v)$ represents a color mapping function.

Suppose after t iterations, a graph transformer $\mathbf{S}$ with $\mathbf{M}$ satisfies $\mathbf{S}(G_1) \neq \mathbf{S}(G_2)$, yet GD-WL with $\mathbf{M}$ fails to distinguish G_1 and G_2 as non-isomorphic. This implies that from iteration 0 to t in

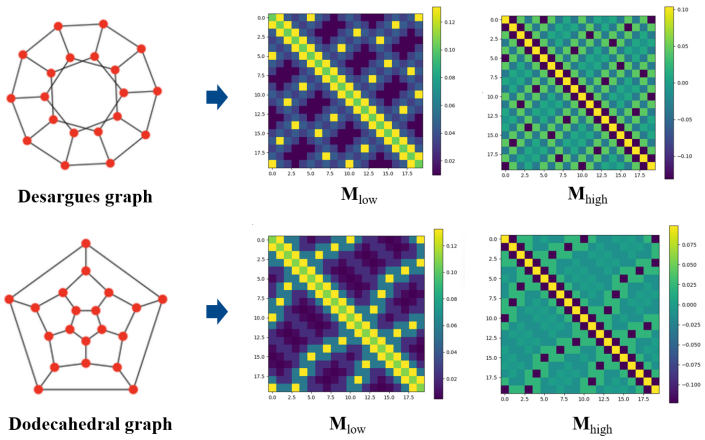


Figure 7: The heatmaps of $\mathbf{M}$ in the Desargues graph and the Dodecahedral graph.

the GD-WL test, G_1 and G_2 always have the same collection of node labels. Specifically, since G_1 and G_2 have the same GD-WL node labels at each iteration $i+1$ for $i = 0, \ldots, t-1$, they share the same collection of GD-WL node labels:

$$\left\{\left(\mathbf{M}_{v,u}, \chi_G^i(u)\right) : u \in V\right\}.$$

Otherwise, the GD-WL test would have produced different node labels at iteration $i+1$ for G_1 and G_2.

Now, we show that for the same graph (say, $G = G_1$), if $\chi_G^i(v) = \chi_G^i(w)$, then the graph transformer node features $\mathbf{h}_v^i = \mathbf{h}_w^i$ for any iteration i. This is true for $i = 0$ because both the GD-WL and the graph transformer start with identical node features. Assuming this holds for iteration j, if $\chi_G^{j+1}(v) = \chi_G^{j+1}(w)$, we deduce:

$$\left\{\left(\mathbf{M}_{v,u}, \chi_G^j(u)\right) : u \in V\right\} = \left\{\left(\mathbf{M}_{w,u}, \chi_G^j(u)\right) : u \in V\right\}.$$

For simplicity, we set the parameter α of the SA-Pruner to 1 and ignore edge feature injection. We then deduce:

$$\mathbf{h}_v^{j+1} = \sum_{u \in V} \text{Softmax}\left(\text{Score}(\mathbf{H}^j)_{u,v} \odot f(\mathbf{M})_{u,v}\right) \cdot \mathbf{V}(\mathbf{h}_u^j) = \varphi\left(\left\{\left(\mathbf{M}_{v,u}, \chi_G^j(u)\right) : u \in V\right\}\right).$$

Hence,

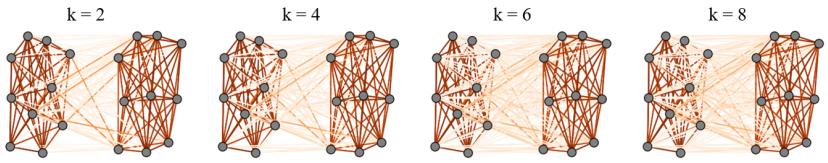
$$\mathbf{h}_v^{j+1} = \varphi\left(\left\{\left(\mathbf{M}_{v,u}, \chi_G^j(u)\right) : u \in V\right\}\right) = \varphi\left(\left\{\left(\mathbf{M}_{w,u}, \chi_G^j(u)\right) : u \in V\right\}\right) = \mathbf{h}_w^{j+1}.$$

By induction, if $\chi_G^i(v) = \chi_G^i(w)$, then $\mathbf{h}_v^i = \mathbf{h}_w^i$ for any iteration i. Therefore, from G_1 and G_2 having identical GD-WL node labels, it follows that they must also have the same graph transformer node features. Consequently, $\mathbf{h}_v^{i+1} = \mathbf{h}_w^{i+1}$. Given that the graph-level readout function is permutation-invariant with respect to the collection of node features, we conclude that $\mathbf{S}(G_1) = \mathbf{S}(G_2)$, which leads to a contradiction.

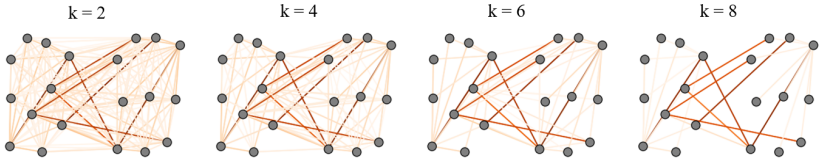
This completes the proof of the first half of the theorem. For the second half of the theorem, we can entirely rely on the proof of Theorem E.3 in Zhang et al. (2023) (provided in Appendix E.3), which presents a similar situation.

A.3 Visualization of Spectral Filtering Effects

Although we have already visualized the spectral responses of the low-pass and high-pass filters in the main text, demonstrating that larger values of k lead to stronger filtering effects on low- and high-frequency components, we additionally visualize the corresponding Spectral Relation Metrics in the spatial domain in Fig. 8 and Fig. 9. As k increases, the learned structural patterns are shown to converge to stable configurations that consistently reflect the respective frequency characteristics.

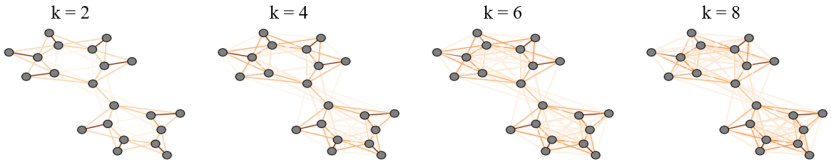


(a) Low-pass

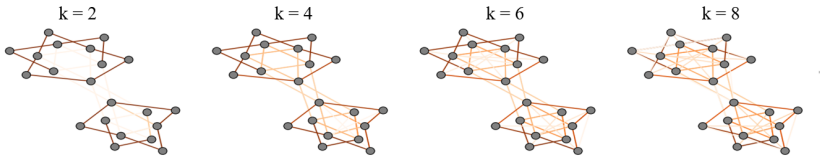


(b) High-pass

Figure 8: Spatial visualization of (a) $\mathbf{M}_{\text{low}}$ and (b) $\mathbf{M}_{\text{high}}$ on a stochastic block model with two communities, under different k settings.



(a) Low-pass



(b) High-pass

Figure 9: Spatial visualization of (a) $\mathbf{M}_{\text{low}}$ and (b) $\mathbf{M}_{\text{high}}$ in the 2-Phenylpyridine module, under different k settings.

A.4 Perpetides of Normalized Laplacian Matrix $\mathbf{L}_{\text{SYM}}$

1. Definitions and Properties

The normalized graph Laplacian matrix $\mathbf{L}_{\text{sym}}$ is defined as:

$$\mathbf{L}_{\text{sym}} = D^{-1/2}(D - A)D^{-1/2} = I - D^{-1/2}AD^{-1/2}$$

where D is the degree matrix (a diagonal matrix where each entry $D_{ii} = \deg(i)$), A is the adjacency matrix, where $A_{ij} = 1$ if there is an edge between nodes i and j, and 0 otherwise.

The matrix $\mathbf{L}_{\text{sym}}$ is symmetric and positive semi-definite, with real eigenvalues satisfying:

$$0 = \lambda_0 < \lambda_1 \leq \cdots \leq \lambda_{\max} \leq 2.$$

2. Proof of Orthogonality of Eigenvectors

Since $\mathbf{L}_{\text{sym}}$ is real and symmetric, its eigenvectors corresponding to distinct eigenvalues are orthogonal. Let $\lambda_i \neq \lambda_j$ be two distinct eigenvalues, with corresponding eigenvectors u_i and u_j. Then:

$$\mathbf{L}_{\text{sym}}u_i = \lambda_i u_i, \quad \mathbf{L}_{\text{sym}}u_j = \lambda_j u_j$$

Taking the inner product of the first equation:

$$u_j^T \mathbf{L}_{\text{sym}} u_i = \lambda_i u_j^T u_i$$

Taking the inner product of the second equation:

$$u_i^T \mathbf{L}_{\text{sym}} u_j = \lambda_j u_i^T u_j$$

Since $\mathbf{L}_{\text{sym}}$ is symmetric, the left-hand sides of these equations are equal, so we obtain:

$$(\lambda_i - \lambda_j) u_i^T u_j = 0$$

For $\lambda_i \neq \lambda_j$, this implies:

$$u_i^T u_j = 0$$

Thus, the eigenvectors corresponding to distinct eigenvalues of $\mathbf{L}_{\text{sym}}$ are orthogonal.

3. PROOF OF EIGENVALUE RANGE $[0, 2]$

We now prove that the eigenvalues of the normalized Laplacian matrix $\mathbf{L}_{\text{sym}}$ lie in the range $[0, 2]$.

LOWER BOUND: $\lambda_0 = 0$

We first show that all eigenvalues are non-negative, with the smallest eigenvalue $\lambda_0 = 0$. Using the Rayleigh quotient:

$$R(\mathbf{f}) = \frac{\mathbf{f}^T \mathbf{L}_{\text{sym}} \mathbf{f}}{\mathbf{f}^T \mathbf{f}} \geq 0$$

where $\mathbf{f}$ is any vector. For $\mathbf{L}_{\text{sym}}$, the Rayleigh quotient is always non-negative. Additionally, we have the expression for the Rayleigh quotient as:

$$R(\mathbf{f}) = \sum_{(i,j) \in E} W_{ij} \left(\frac{\sqrt{f_i}}{\sqrt{d_i}} - \frac{\sqrt{f_j}}{\sqrt{d_j}} \right)^2$$

Since the sum is always non-negative, we conclude that the smallest eigenvalue $\lambda_0 = 0$, which corresponds to the constant eigenvector (where all components of $\mathbf{f}$ are equal).

UPPER BOUND: $\lambda_{\max} \leq 2$

Next, we show that the largest eigenvalue $\lambda_{\max}$ is bounded above by 2. Using the spectral properties of $\mathbf{L}_{\text{sym}}$, we can write:

$$\mathbf{L}_{\text{sym}} + \mathbf{L}_{\text{sym}}^2 = 2I$$

This implies:

$$\mathbf{L}_{\text{sym}}^2 = 2I - \mathbf{L}_{\text{sym}}$$

Thus, we deduce that the largest eigenvalue is:

$$\lambda_{\max} \leq 2$$

For bipartite graphs, $\lambda_{\max} = 2$, while for non-bipartite graphs, we have $\lambda_{\max} < 2$.

Conclusion

We have proven the following:

- **Orthogonality:** The eigenvectors corresponding to distinct eigenvalues of the normalized Laplacian matrix $\mathbf{L}_{\text{sym}}$ are orthogonal.
- **Eigenvalue Range:** The eigenvalues of $\mathbf{L}_{\text{sym}}$ lie in the range $[0, 2]$, with the smallest eigenvalue being $\lambda_0 = 0$ (corresponding to the constant eigenvector) and the largest eigenvalue $\lambda_{\max} = 2$ for bipartite graphs, and strictly less than 2 for non-bipartite graphs.

Thus, the eigenvalues of $\mathbf{L}_{\text{sym}}$ lie within the interval $[0, 2]$, completing the proof.

A.5 Comparison with Current Spectral Graph Transformer Approaches

It is important to note that while a few existing works have explored the combination of spectral domain and graph transformers, the motivations and objectives of these works significantly differ from our study. The core of our approach lies in incorporating structural knowledge from the spectral domain to explicitly constrain the learning process of attention (such as pruning and scaling). In contrast, Kreuzer et al. (2021) aims to introduce spectral domain knowledge for position encoding, implicitly guiding the learning of attention, but does not directly constrain attention pruning or align it with spectral domain signals. Notably, such position encoding could also be incorporated into our framework. On the other hand, Bo et al. (2023) focuses on using the transformer framework to learn graph spectral filters, but strictly speaking, this is not a standard Graph Transformer, as it does not directly learn the attention between nodes.

B Implementation Details

B.1 Node and Edge Positional Encoding

In this work, we incorporate edge and node positional encoding from GRIT (Ma et al., 2023) as auxiliary structural features to enhance the model's input representations. Specifically, the input positional encoding $\mathbf{F}_{\text{edge},ij}$ are constructed as

$$\mathbf{F}_{\text{edge},ij} = [\mathbf{I}, \mathbf{T}, \mathbf{T}^2, \ldots, \mathbf{T}^{K-1}]_{i,j} \in \mathbf{R}^K, \tag{15}$$

where $\mathbf{T} = \mathbf{D}^{-1}\mathbf{A}$. The node positional encodings $\mathbf{F}_{\text{node},i} = \mathbf{F}_{\text{edge},ii}$.

B.2 GRIT Attention Computation

In this work, we adopt the GRIT attention calculation module to better encode graph structures. GRIT updates edge features during attention computation, allowing structural information to be implicitly integrated. Specifically, the attention score between node i and j is defined as

$$\text{Score}(\mathbf{H}, \mathbf{E})_{i,j} = \mathbf{W}_{\text{A}} \cdot \text{Update}(\mathbf{E}_{i,j}, \mathbf{H}_i, \mathbf{H}_j), \tag{16}$$

where $\text{Update}(\cdot)$ denotes a function that computes the updated edge representation $\hat{\mathbf{E}}_{i,j}$ as

$$\begin{aligned} \hat{\mathbf{E}}_{i,j} &= \text{Update}(\mathbf{E}_{i,j}, \mathbf{H}_i, \mathbf{H}_j) \\ &= \sigma\Big(\rho\Big((\mathbf{W}_{\text{Q}}\mathbf{H}_i + \mathbf{W}_{\text{K}}\mathbf{H}_j) \odot \mathbf{W}_{\text{Ew}}\mathbf{E}_{i,j}\Big) + \mathbf{W}_{\text{Eb}}\mathbf{E}_{i,j}\Big), \end{aligned} \tag{17}$$

where $\sigma(\cdot)$ and $\rho(\cdot)$ denote non-linear activation functions, $\odot$ represents element-wise multiplication, and $\mathbf{W}_{\text{A}}$, $\mathbf{W}_{\text{Q}}$, $\mathbf{W}_{\text{K}}$, $\mathbf{W}_{\text{Ew}}$, $\mathbf{W}_{\text{Eb}}$ are learnable weight matrices. Given $\mathbf{A} = \text{Softmax}(\text{Score}(\mathbf{H}, \mathbf{E}))$, the updated edge features are further incorporated into the aggregation step as an edge enhancement term by

$$\text{Attn}(\mathbf{H}, \mathbf{E}) = \mathbf{A} \cdot \mathbf{V}(\mathbf{H}) + \text{EdgeEnh}(\mathbf{A}, \hat{\mathbf{E}}). \tag{18}$$

The edge enhancement term $\text{EdgeEnh}(\cdot)$ is defined as

$$\text{EdgeEnh}(\mathbf{A}, \hat{\mathbf{E}})_i = \sum_{j \in \mathcal{V}} \mathbf{A}_{ij} \cdot \mathbf{W}_{\text{Ev}}\hat{\mathbf{E}}_{i,j}, \tag{19}$$

where $\mathcal{V}$ denotes the node set of the input graph and $\mathbf{W}_{\text{Ev}}$ is a weight matrix.

B.3 Notes on Reproducibility

Our implementation is based on *PyTorch* and *PyTorch Geometric*. Some graph operations in *PyTorch Geometric* are non-deterministic when executed on GPUs, which may lead to slight variations in results across different runs, even under the same random seed. This behavior is common across many existing GNN implementations.

C Experimental Details

C.1 Description of Datasets

A summary of the dataset statistics and characteristics is provided in Table 7. The first five datasets are adopted from (Dwivedi et al., 2023), while the middle two are sourced from (Dwivedi et al., 2022). Readers are referred to (Rampášek et al., 2022) for more details of the datasets.

Table 7: Statistics of datasets used in our experiments.

Dataset	# Graphs	Avg. # nodes	Avg. # edges	Directed	Prediction level	Prediction task	Metric
ZINC(-full)	12,000 (250,000)	23.2	24.9	No	graph	regression	Mean Abs. Error
MNIST	70,000	70.6	564.5	Yes	graph	10-class classif.	Accuracy
CIFAR10	60,000	117.6	941.1	Yes	graph	10-class classif.	Accuracy
PATTERN	14,000	118.9	3,039.3	No	inductive node	binary classif.	Weighted Accuracy
CLUSTER	12,000	117.2	2,150.9	No	inductive node	6-class classif.	Weighted Accuracy
Peptides-func	15,535	150.9	307.3	No	graph	10-task classif.	Avg. Precision
Peptides-struct	15,535	150.9	307.3	No	graph	11-task regression	Mean Abs. Error

C.2 Dataset splits and Random Seed

We conduct the experiments on the standard train/validation/test splits of the evaluated benchmarks, following previous works (Rampášek et al., 2022). For each dataset, we execute 4 trials with different random seeds (0, 1, 2, 3) and report the mean performance and standard deviation.

C.3 Hyperparameters

Due to constraints in time and computational resources, we refrain from conducting an exhaustive hyperparameter search. Instead, we initialize from the hyperparameter configuration of GRIT (Ma et al., 2023) and apply limited tuning to strike a balance between model complexity and performance. The final configurations are summarized in Table 8 and Table 9.

C.4 Experiment Device

For all datasets, we conducted the experiments with *NVIDIA A100*.

D Use of LLMs Statement

In preparing this manuscript, we used Chatgpt to assist in language polishing and improving readability of the text. The Chatgpt was used solely for drafting and refining phrasing; all scientific content, ideas, derivations, and experimental results were developed and verified by the authors. We take full responsibility for the accuracy, validity, and integrity of all content in this manuscript, including any portions generated with the assistance of LLMs.

Table 8: Hyperparameters for five datasets from BenchmarkingGNNs.

Hyperparameter	**ZINC**	**MNIST**	**CIFAR10**	**PATTERN**	**CLUSTER**
# Transformer Layers	11	5	8	7	18
Hidden dim	64	64	52	64	96
# Heads	8	4	4	8	8
Dropout	0	0	0	0	0.01
Attention dropout	0.2	0.2	0.5	0.1	0.15
Graph pooling	sum	mean	mean	mean	–
α (low-pass branch)	0.7	0.3	0.3	0.9	0.9
α (high-pass branch)	0.7	0.3	0.3	0.9	0.9
k (filter setting)	20	18	18	20	16
PE dim (RW-steps)	21	18	18	21	32
PE encoder	linear	linear	linear	linear	linear
Batch size	256	64	64	32	16
Learning Rate	0.001	0.001	0.001	0.001	0.0005
# Epochs	2000	200	200	100	100
# Warmup epochs	50	5	5	5	5
Weight decay	1e−5	4e−5	1e−5	1e−4	1e−5
# Parameters	473,473	377,738	394,246	508,641	2,912,166

Table 9: Hyperparameters for ZINC-full and the Long-Range Graph Benchmark.

Hyperparameter	**ZINC-full**	**Peptides-func**	**Peptides-struct**
# Transformer Layers	10	5	7
Hidden dim	64	96	96
# Heads	8	4	8
Dropout	0	0	0.05
Attention dropout	0.2	0.5	0.2
Graph pooling	sum	mean	mean
α (low-pass branch)	0.3	0.3	0.9
α (high-pass branch)	0.3	0.3	0.9
k (filter setting)	20	16	16
PE dim (RW-steps)	21	17	18
PE encoder	linear	linear	linear
Batch size	512	32	32
Learning Rate	0.001	0.003	0.0003
# Epochs	2000	200	200
# Warmup epochs	50	5	5
Weight decay	1e−7	1e−5	0.0
# Parameters	729,601	840,186	1,160,747