THE GRAPH SPECTRAL TOKEN: ENHANCING GRAPH TRANSFORMERS WITH SPECTRAL INFORMATION *

Zihan Pengmei
Department of Chemistry
The University of Chicago
Chicago,IL,USA
zpengmei@uchicago.edu

Zimu Li
Yau Mathematical Science Center
Tsinghua University
Beijing, China
lizm@mail.sustech.edu.cn

ABSTRACT

Graph Transformers have emerged as a powerful alternative to Message-Passing Graph Neural Networks (MP-GNNs) to address limitations such as over-squashing of information exchange. However, incorporating graph inductive bias into transformer architectures remains a significant challenge. In this report, we propose *the Graph Spectral Token*, a novel approach to directly encode graph spectral information, which captures the global structure of the graph, into the transformer architecture. By parameterizing the auxiliary [CLS] token and leaving other tokens representing graph nodes, our method seamlessly integrates spectral information into the learning process. We benchmark the effectiveness of our approach by enhancing two existing graph transformers, GraphTrans and SubFormer. The improved GraphTrans, dubbed GraphTrans-Spec, achieves over 10% improvements on large graph benchmark datasets while maintaining efficiency comparable to MP-GNNs. SubFormer-Spec demonstrates strong performance across various datasets. The code for our implementation is available at `https://github.com/zpengmei/SubFormer-Spec`.

***K*eywords** Graph Spectrum · Graph Transformer · Graph Neural Networks

1 Introduction

Graph transformers have demonstrated impressive results compared to conventional Message-Passing Graph Neural Networks (MP-GNNs) in various graph benchmarks. They aim to solve inherent limitations of MP-GNNs, such as the over compression of information, where the recursive neighborhood aggregation can lead to loss of local information, and the under-reaching problem, where the receptive field of nodes is limited by the number of layers [1, 2, 3, 4, 5]. The self-attention mechanism in graph transformers works as a fully-connected graph neural network, allowing for more efficient information exchange. GraphTrans [2] and SubFormer [5] are two similar graph transformer architectures that combine shallow MP-GNN layers for local feature extraction and standard Transformer blocks for global information exchange. However, SubFormer incorporates the molecular coarse-graining assumption [6, 7], which simplifies the graph structure by grouping nodes into substructures, while GraphTrans does not. SubFormer demonstrates that by incorporating proper prior knowledge, such as the coarse-graining assumption, satisfactory performance can be achieved without further complicating the updating function.

In this report, we propose *the Graph Spectral Token* as a general method to include graph spectrum information, which captures the global structure of the graph, into the design of graph transformers. Starting from BERT [8], an auxiliary [CLS] token has been introduced as a global learnable pooling function from all the other tokens [9]. Conventionally, the [CLS] token is initialized with random parameters. Instead, we propose to encode the graph spectral information via the [CLS] token. Updating both spectral and ordinary graph features simultaneously through Transformer further enhances the model's expressive power than simply utilizing graph spectrum via graph convolution or its generalization [10, 11]. We extensively benchmark the improved graph transformers, termed as SubFormer-Spec and GraphTrans-Spec

*Technical Report.

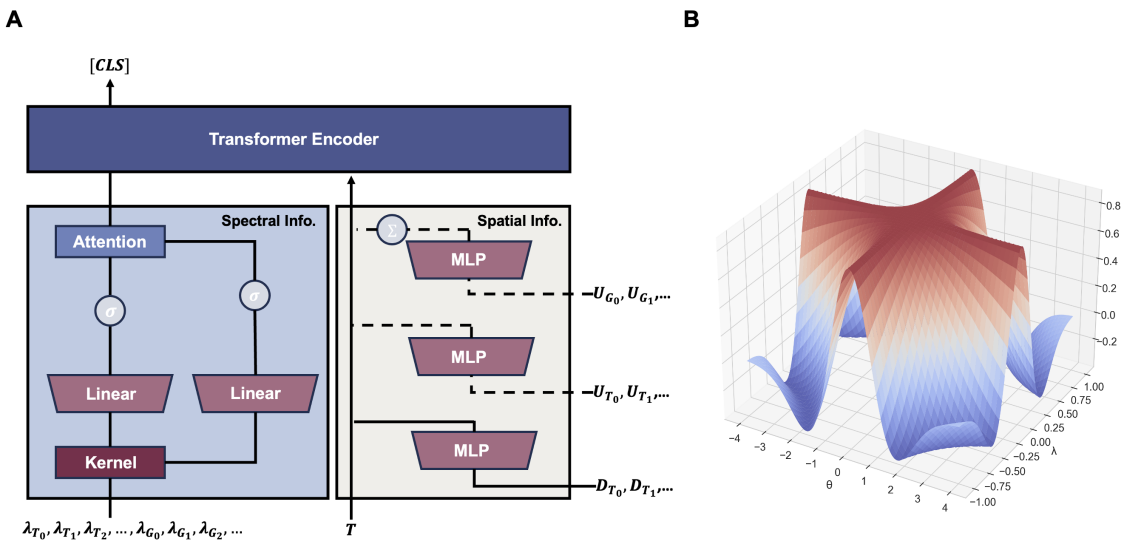

Figure 1: Illustration of the proposed *the Graph Spectral Token* for graph Transformer. (A) Spectral information is processed with an auxiliary network and assigned to the [CLS] token, while ordinary node features are processed through conventional tokens with node degree and optional Laplacian eigenvectors information. (B) Visualization of Mexican Hat kernel along time constants θ and eigenvalues λ.

on multiple molecular modeling datasets. The impressive results suggest the potential to further investigate *the Graph Spectral Token* as an effective and efficient way to inject graph spectral information into graph transformers.

2 Incorporation into Graph Transformers

2.1 The Graph Spectral Token

To begin with, we map eigenvalues into a higher-dimensional space which is expected to create new representations that capture more complex patterns or relationships that may not be easily discernible in the original input space, mitigating issues as eigenvalue multiplicities. To be specific, let $\boldsymbol{\lambda}$ be the column vector of graph normalized Laplacian eigenvalues, we process each of them using the Mexican hat kernel (as illustrated in Figure 1.B)

$$g(\theta_i \lambda_i) = \frac{2}{\sqrt{3}\pi^{1/4}}(1 - \theta_i^2 \lambda_i^2) \exp(-\frac{\theta_i^2 \lambda_i^2}{2}), \tag{1}$$

which a specific type of kernel used in signal analysis for feature extraction [12]. It is optional to choose other functions referred to as *spectral kernels* here like heat kernel, Gaussian kernel [13] or even trigonometric functions [11]. It is also worth mentioning that the expansion of spectrum can be done either before the kernel or after the kernel. We then compute the following *spectral attention* among the embedded eigenvalues through the Softmax with tunable weight matrix W_1:

$$\boldsymbol{s} = (s_j) = \left(\frac{\exp(W_1 g(\boldsymbol{\theta} \cdot \boldsymbol{\lambda})_i}{\sum_j \exp(W_1 g(\boldsymbol{\theta} \cdot \boldsymbol{\lambda})_j} \right) \tag{2}$$

and initialize a vector on the graph spectral token labeled by 0 as

$$z_0^{(0)} = \boldsymbol{s} \odot W_2 \boldsymbol{\lambda}, \tag{3}$$

where $\odot$ means Hardmard product on vectors.

2.2 SubFormer-Spec Architecture

In SubFormer [5], the shallow MP-GNN updates ordinary graph features followed by a standard Transformer which utilizes the self-attention mechanism to learn the coarse-grained tree [6, 7]. Consider two matrices: $U_G \in \mathbb{R}^{n \times n}$ and $U_T \in \mathbb{R}^{m \times n}$. The columns of U_G and U_T represent the Laplacian eigenvectors of a graph and its coarse-grained tree,

respectively. Additionally, let $S \in 0, 1^{m \times n}$ denote the matrix that assigns graph nodes to the corresponding nodes of its coarse-grained tree.

We first concatenate U_T and the matrix product SU_G, processed through learnable functions Φ_i, as the eigenvector positional encoding (EPE):

$$U = [\Phi_1(U_T), \Phi_2(SU_G)] \tag{4}$$

Learnable functions can be implemented as either a fully-connected layer or as a SignNet, which is designed to ensure eigenvector invariance to sign flips, as discussed in [14]. Let $Z \in \mathbb{R}^{m \times d}$ denote the coarse-grained tree feature. We prepare the input for the Transformer block by concatenating Z with node degree positional embedding D_T (DPE) and the previously mentioned EPE.

$$Z^{'(0)} = \text{FFL}([Z, D_T]), \quad Z^{(0)} = \text{FFL}([Z^{'(0)}, U]). \tag{5}$$

Note that we set $Z_0^{(0)} = z_0^{(0)}$ defined in Eq.(3) which incorporates the spectral information of *both* G and T. Then EPE is optional in our architecture since both the graph spectral token and MP-GNN layers already provide sufficient structural information. Then We update $Z^{(0)}$ using the standard Transformer.

Algorithm 1 SubFormer-Spec

$X = \text{Embedding}(X_{ir}) \in \mathbb{R}^{n \times d_1}, Z = \text{Embedding}(Z_{js}) \in \mathbb{R}^{m \times d_2}$ ▷ Input embedding
$X'^{(l} = \text{MPNN}(X^l)$.
$X^{(l+1)} = X^{(l)'} + \text{FFL}(S^T Z^{(l)} W_2^{(l)})$. ▷ Expanding coarse-grained feature to the original graph
$Z^{(l+1)} = Z^{(l)} + \text{FFL}(S X^{(l+1)} W_3^{(l)})$. ▷ Compressing graph feature to the coarse-grained tree
$D_T = (d_{T,jr}) \in \mathbb{R}^{m \times d_T}$ ▷ Embedding of node degrees
$Z^{(0)} = \text{FFL}([Z^{(0)}, D_T]) \in \mathbb{R}^{n \times d}$ ▷ Concatenation with DPE
if initialize message-passing with EPE **then**
    $U = [\Phi_1(U_T), \Phi_2(U_G)]$ ▷ Preparing EPE using a Fully-connected layer or SignNet
    $Z^{(0)} = \text{FFL}([Z^{(0)}, U]) \in \mathbb{R}^{n \times d}$ ▷ Concatenation with EPE
end if
$\boldsymbol{\lambda} = [\boldsymbol{\lambda}_T, \boldsymbol{\lambda}_G], \boldsymbol{\theta} = (\theta_i), W_1, W_2$ ▷ Initialization of spectral input with three collections of weights
$g(\theta_i \lambda_i) = (2/\sqrt{3}\pi^{1/4})(1 - \theta_i^2 \lambda_i^2) \exp(-\theta_i^2 \lambda_i^2/2)$ ▷ Feature Extraction Using Mexican Kernel Function
$\boldsymbol{s} = \text{Softmax}(W_1 g(\boldsymbol{\theta} \cdot \boldsymbol{\lambda})), z_0^{(0)} = \boldsymbol{s} \odot W_2 \boldsymbol{\lambda}$ ▷ High dimensional embedding weighted by correlation scores
Transformer Encoder
Read out the auxiliary token with three-layer MLPs

3 Emperical Results

3.1 ZINC and Long-Range Graph Benchmark Datasets

The ZINC dataset [15] serves as a standard benchmark consisting of small drug-like molecular graphs. Incorporating spectral information into the SubFormer-Spec model has shown to slightly improve performance, making it comparable to other state-of-the-art methods. It's noteworthy that many contemporary graph learning methods perform equally well on the ZINC dataset. For instance, SubFormer-Spec demonstrates a lower validation error than some methods, while achieving a similar test error, as detailed in Table 1.

In the domain of long-range graph benchmarks, particularly with the Peptides-Struct and Peptides-func datasets [16], SubFormer-Spec outperforms its predecessor, the original SubFormer model as listed in Table 2. Graphs in these datasets are substantially larger than those in the ZINC dataset, incorporating a greater degree of long-range interactions. Interestingly, the benefit of incorporating the spectral token is more prominent in large graph datasets. This aligns with the findings in [17], which indicate that the eigen spectrum of graphs becomes an increasingly powerful discriminative feature as graph size escalates.

3.2 MoleculeNet

For a comprehensive evaluation, we selected several datasets from MoleculeNet, adhering to the recommended settings outlined in [23]. Table 3 shows the effectiveness of the spectral token across a diverse range of chemical datasets. Despite kernel methods are generally doing better on small-scale datasets as SIDER and BBBP.

Table 1: Results on ZINC dataset. Top 3 results are highlighted.Available benchmarks are taken from [18, 19, 20, 11, 21]

Model	Test MAE(↓)	Valid MAE	Walltime(s)
GCN	0.278±0.003	-	4
GAT	0.384±0.007	-	13
GIN	0.526±0.051	-	10
HIMP	0.151±0.006	-	-
Transformer+LapPE	0.226±0.014	-	45
SAN	0.181±0.004	-	74
SAN+RWPE	0.104±0.004	-	135
Graphormer	0.122±0.006	-	-
GraphGPS	0.070±0.004	-	21
Graph MLP-mixer	0.073±0.001	-	6
Graph Vit	0.085±0.005	-	-
Specformer	**0.066±0.003**	-	156
PDF	**0.066±0.002**	0.085±0.004	4
Spec-GN	0.070±0.002	0.088±0.003	-
SubFormer	0.077±0.003	0.085±0.002	3
SubFormer-Spec	**0.068±0.005**	**0.072±0.003**	7

Table 2: Results on Peptides-Struct/Func datasets from long-range graph benchmarks. Top 3 results are highlighted. Available benchmarks are taken from [18, 19, 22].

Peptides-Func	Test AP(↑)
GCN	0.5930±0.0023
GINE	0.5498±0.0079
HIMP	0.5672±0.0038
GTR	0.6519±0.0036
Transformer+LapPE	0.6326±0.0126
SAN+LapPE	0.6384±0.0121
SAN+RWPE	0.6439±0.0075
GraphGPS	0.6535±0.0041
GraphTrans	0.6313±0.0039
Graph MLP-mixer	**0.6970±0.0080**
Graph Vit	0.6942±0.0075
SubFormer	0.6732±0.0045
GraphTrans-Spec	**0.6957±0.0115**
SubFormer-Spec	**0.7014±0.0086**

Peptides-Struct	Test MAE(↓)	Walltime(s)
GCN	0.3496±0.0013	5
GINE	0.3547±0.0045	4
HIMP	0.2653±0.0010	6
GTR	0.2502±0.0017	-
Transformer+LapPE	0.2529±0.0016	6
SAN+LapPE	0.2683±0.0043	54
SAN+RWPE	0.2545±0.0012	50
GraphGPS	0.2500±0.0005	12
GraphTrans	0.2777±0.0025	-
Graph MLP-mixer	0.2475±0.0015	9
Graph Vit	**0.2449±0.0016**	9
SubFormer	**0.2464±0.0012**	7
GraphTrans-Spec	0.2487±0.0009	7
SubFormer-Spec	**0.2441±0.0011**	7

3.3 Organic Photovoltaics with Donor-Acceptor (OPDA) Structure Dataset.

Graph transformers, akin to their text counterparts, excel in addressing long-range interactions in large graphs, a task particularly relevant in chemical contexts where charge-transfer is a critical long-range phenomenon. In OPDA molecules [25], charge-transfer involves a delocalizing process where an electron is transferred from a localized donor group to a distant acceptor group. These molecules, designed with separated donor and acceptor groups, are pivotal in tuning physical properties such as the energy gap between the highest occupied molecular orbital (HOMO) and lowest unoccupied molecular orbital (LUMO).

Table 3: Results on MoleculeNet. Top 3 results are highlighted.Available benchmarks are taken from [23, 7, 24, 5]

Dataset	**TOX21**	**TOXCAST**	**MUV**	**HIV**	**SIDER**	**BBBP**
Num. Task	12	617	17	1	27	1
Metric	ROC-AUC(↑)	ROC-AUC	RPC-AUC(↑)	ROC-AUC	ROC-AUC	ROC-AUC
RF	0.769±0.015	-	-	0.684±0.009	**0.714±0.000**	**0.781±0.006**
XGBoost	0.794±0.014	0.640±0.005	0.086±0.033	0.756±0.000	0.656±0.027	0.696±0.000
Kernel SVM	0.822±0.006	0.669±0.014	**0.137±0.033**	0.792±0.000	**0.682±0.013**	**0.729±0.000**
LR	0.794±0.015	0.605±0.003	0.070±0.009	0.702±0.018	0.643±0.011	0.699±0.002
GCN	0.840±0.004	0.735±0.002	0.114±0.029	0.761±0.010	0.601±0.013	0.712±0.012
GIN	0.850±0.009	**0.741±0.004**	0.091±0.033	0.756±0.014	0.571±0.012	0.689±0.013
HIMP	**0.874±0.005**	0.721±0.004	0.114±0.041	0.788±0.080	0.562±0.013	0.701±0.011
Graphormer	-	-	-	**0.805±0.005**	-	-
GraphGPS	0.841±0.003	0.714±0.006	0.087	0.788±0.010	0.607±0.011	0.651±0.038
SchNet	0.769	0.685	-	-	0.597	-
EGNN	**0.854**	0.739	-	-	0.604	-
ClofNet	0.842	0.700	-	-	0.603	-
SubFormer	0.851±0.008	**0.752±0.003**	**0.182±0.019**	**0.795±0.008**	0.678±0.014	0.703±0.010
SubFormer-Spec	**0.867±0.005**	**0.764±0.005**	**0.203±0.012**	**0.805±0.008**	**0.683±0.005**	**0.731±0.021**

Table 4: Results on OPDA dataset. Top 1 results are highlighted. All results are MAE, lower the better. Each model is allowed to train 300 epochs.

Property	**$E_{HOMO-LUMO}$**	**Packing density**	**E_{HOMO}**	**E_{LUMO}**	**Dip. moment**	**Walltime(s/epoch)**
GCN	0.845	0.462	1.526	0.647	1.369	0.94
GIN	0.210	0.073	0.273	0.557	1.236	0.91
GATv2	0.299	0.069	1.357	0.445	1.548	1.04
GraphGPS	0.080	0.015	0.068	0.044	0.966	1.92
SchNet	0.331	0.215	0.429	0.297	1.236	1.58
DimeNet++	0.139	0.026	0.190	0.093	1.089	3.97
ClofNet	0.132	0.016	0.123	0.083	1.259	1.34
EGNN	0.112	0.174	0.138	0.076	1.216	1.14
SF	0.070	0.009	0.050	0.031	0.997	1.53
SF-Spec	**0.047**	**0.008**	**0.043**	**0.028**	**0.888**	1.61

References

[1] Vijay Prakash Dwivedi and Xavier Bresson. A Generalization of Transformer Networks to Graphs. *arXiv e-prints*, page arXiv:2012.09699, December 2020.

[2] 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. *Advances in Neural Information Processing Systems*, 34:13266–13279, 2021.

[3] Devin Kreuzer, Dominique Beaini, William L. Hamilton, Vincent Létourneau, and Prudencio Tossou. Rethinking Graph Transformers with Spectral Attention. *arXiv e-prints*, page arXiv:2106.03893, June 2021.

[4] Jinwoo Kim, Dat Nguyen, Seonwoo Min, Sungjun Cho, Moontae Lee, Honglak Lee, and Seunghoon Hong. Pure transformers are powerful graph learners. *Advances in Neural Information Processing Systems*, 35:14582–14595, 2022.

[5] Zihan Pengmei, Zimu Li, Chih-chan Tien, Risi Kondor, and Aaron R Dinner. Transformers are efficient hierarchical chemical graph learners. *arXiv preprint arXiv:2310.01704*, 2023.

[6] Wengong Jin, Regina Barzilay, and Tommi Jaakkola. Junction Tree Variational Autoencoder for Molecular Graph Generation. *arXiv e-prints*, page arXiv:1802.04364, February 2018.

[7] Matthias Fey, Jan-Gin Yuen, and Frank Weichert. Hierarchical Inter-Message Passing for Learning on Molecular Graphs. *arXiv e-prints*, page arXiv:2006.12179, June 2020.

[8] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[9] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[10] Mikael Henaff, Joan Bruna, and Yann LeCun. Deep convolutional networks on graph-structured data. *arXiv preprint arXiv:1506.05163*, 2015.

[11] Deyu Bo, Chuan Shi, Lele Wang, and Renjie Liao. Specformer: Spectral graph neural networks meet transformers. In *The Eleventh International Conference on Learning Representations*, 2023.

[12] Ljubiša Stanković and Ervin Sejdić, editors. *Vertex-Frequency Analysis of Graph Signals*. Signals and Communication Technology. Springer International Publishing, Cham, 2019.

[13] Petar M. Djurić and Cédric Richard, editors. *Cooperative and graph signal processing: principles and applications*. Academic Press, an imprint of Elsevier, London, United Kingdom ; Sand Diego, CA, 2018. OCLC: on1011518558.

[14] Derek Lim, Joshua Robinson, Lingxiao Zhao, Tess Smidt, Suvrit Sra, Haggai Maron, and Stefanie Jegelka. Sign and basis invariant networks for spectral graph representation learning. *arXiv preprint arXiv:2202.13013*, 2022.

[15] John J Irwin, Teague Sterling, Michael M Mysinger, Erin S Bolstad, and Ryan G Coleman. Zinc: a free tool to discover chemistry for biology. *Journal of chemical information and modeling*, 52(7):1757–1768, 2012.

[16] Vijay Prakash Dwivedi, Ladislav Rampášek, Michael Galkin, Ali Parviz, Guy Wolf, Anh Tuan Luu, and Dominique Beaini. Long range graph benchmark. *Advances in Neural Information Processing Systems*, 35:22326–22340, 2022.

[17] Richard C Wilson and Ping Zhu. A study of graph spectra for comparing graphs and trees. *Pattern Recognition*, 41(9):2833–2841, 2008.

[18] Ladislav Rampášek, Michael Galkin, Vijay Prakash Dwivedi, Anh Tuan Luu, Guy Wolf, and Dominique Beaini. Recipe for a general, powerful, scalable graph transformer. *Advances in Neural Information Processing Systems*, 35:14501–14515, 2022.

[19] Xiaoxin He, Bryan Hooi, Thomas Laurent, Adam Perold, Yann LeCun, and Xavier Bresson. A generalization of vit/mlp-mixer to graphs. In *International Conference on Machine Learning*, pages 12724–12745. PMLR, 2023.

[20] Mingqi Yang, Wenjie Feng, Yanming Shen, and Bryan Hooi. Towards better graph representation learning with parameterized decomposition & filtering. *arXiv preprint arXiv:2305.06102*, 2023.

[21] Mingqi Yang, Yanming Shen, Rui Li, Heng Qi, Qiang Zhang, and Baocai Yin. A new perspective on the effects of spectrum in graph neural networks. In *Proceedings of the 39th International Conference on Machine Learning*, 2022.

[22] Zhongyu Huang, Yingheng Wang, Chaozhuo Li, and Huiguang He. Growing like a tree: Finding trunks from graph skeleton trees. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023.

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

[24] Chengxuan Ying, Tianle Cai, Shengjie Luo, Shuxin Zheng, Guolin Ke, Di He, Yanming Shen, and Tie-Yan Liu. Do Transformers Really Perform Bad for Graph Representation? *arXiv e-prints*, page arXiv:2106.05234, June 2021.

[25] David D Landis, Jens S Hummelshøj, Svetlozar Nestorov, Jeff Greeley, Marcin Dułak, Thomas Bligaard, Jens K Nørskov, and Karsten W Jacobsen. The computational materials repository. *Computing in Science & Engineering*, 14(6):51–57, 2012.

A Datasets

We summarize the datasets used in the study, which are mainly composed of MoleculeNet datasets [23] (TOX21, TOXCAST, MUV, HIV, SIDER and BBBP), ZINC [15], long-range graph benchmarks (Peptides-func/struct) [16], and OPDA dataset [25]. We follow all recommended data split methods and evaluation metrics as proposed in the original literature. For OPDA dataset, we randomly split the dataset 8:1:1 for training, validation, and testing. Additionally, we illustrates random samples from OPDA datasets to showcase the characteristics of OPDA systems in Figure 2.

Table 5: Summary of dataset statistics used in this study.

Dataset	**# Graphs**	**Avg. Nodes**	**Avg. Cliques**	**Avg. Edges**	**Task**	**Metric**
ZINC	12000	23.2	14.2	24.9	Regression	Mean Abs. Error
Peptides-func	15535	150.9	168.9	307.3	10-task classif.	Avg. Precision
Peptides-struct	15535	150.9	168.9	307.3	11-task regression	Mean Abs. Error
OPDA	5356	55.8	28.1	63.9	Regression	Mean Abs. Error
TOX21	7831	18.6	12.9	38.6	12-task classif.	ROC-AUC
TOXCAST	8597	18.7	13.0	38.4	617-task classif.	ROC-AUC
MUV	93087	24.2	14.1	52.6	17-task classif.	RPC-AUC
HIV	41127	25.5	15.8	54.9	1-task classif.	ROC-AUC
SIDER	1427	33.6	25.9	70.7	27-task classif.	ROC-AUC
BBBP	2050	23.9	14.5	51.6	1-task classif.	ROC-AUC

B SubFormer-Spec Hyperparameters

The hyperparameters applied to all benchmark datasets in this study are detailed in Table 6 and 7. We arbitrarily choose to use 16 eigenvalues from the original graph and coarse-grain tree (16+16=32 total) for samll graphs, which should be treated a vital hyper-parameter theoretically. And we picked 32+32 eigenvalues for Peptides-Struct dataset and 64+32 for Peptides-Func dataset. We did not perform a systematic search of hyperparameters due to limited computational resources and we are already satisfied with the performance. We use common random seeds as 4321, 1234, 42, 1, etc.

Table 6: Hyperparameter settings for ZINC and Long-range Graph Benchmarks

Model Component	ZINC	Peptides-Struct	Peptides-Func
Optimization			
Warmup Epoch	50	0	20
Epoch	950	100	200
Learning rate	0.001	0.0005	0.001
Optimizer	AdamW	AdamW	AdamW
Scheduler	Cosine	ROP	Cosine
Batch size	32	64	64
Local MP			
# Layers	2	2	2
# Hidden Features	64	64	64
MP Type	GINE	GINE	GINE
Aggregation	Sum	Sum	Sum
Activation	ReLU	ReLU	ReLU
Tree Activation	LeakyReLU	LeakyReLU	LeakyReLU
Dropout	0	0.05	0.05
Edge Dropout	0	0	0
PE			
PE Dim.	10	-	-
PE Type	DEG, LapPE	DEG, None	DEG, None
PE Merge	Concat	Concat	Concat
Transformer			
# Hidden Features	128	128	128
# FFN Hidden Features	128	128	128
# Layers	3	3	3
# Heads	8	8	8
Activation	ReLU	ReLU	GELU
Dropout	0.1	0.05	0.2
Readout			
# Hidden Features	192	128	128
Activation	ReLU	ReLU	GELU

OPDA Dataset. We allow all models to train 300 epoches. All targets are normalized and we only consider heavy atoms following the convention. For OPDA tasks, we use a consistent architecture comprising three GINE MP-GNN layers with ReLU and LeakyReLU activations and four transformer encoder layers with 128 channels, 8 MHA heads, and 256-channel FFNs with GELU activation. A dropout rate of 0.05 is applied for regularization. Positional encoding is exclusively DEG. The readout MLP is 128-channel wide with GELU activation and we just readout from the coarse-grained graph. The model is trained with batches of 64 using an Adam optimizer (learning rate 0.001) and an ROP scheduler.

For all MP-GNNs, we set the dimension to 128-channel wide and 8 layers with a dropout rate of 0.2. For SchNet, we use the default hyperparameters as implemented in the pytorch geometric package. For DimeNet++, we use hidden channel width of 128, 3 interaction blocks, embedding size of 64. For EGNN and ClofNet, we set the hidden dimension to 128 with 4 layers with the attention. While the optimizer settings are the same.

Table 7: Hyperparameter settings for TOX21, TOXCAST, MUV, HIV, SIDER, and BBBP

Model Component	**TOX21**	**TOXCAST**	**MUV**	**HIV**	**SIDER**	**BBBP**
Optimization						
Epoch	50	50	50	15	50	100
Learning rate	0.0001	0.0001	0.0001	0.0001	0.0005	0.0005
Optimizer	AdamW	AdamW	AdamW	AdamW	AdamW	AdamW
Scheduler	None	None	None	ROP	ROP	ROP
Batch size	32	32	128	32	32	64
Local MP						
# Layers	6	3	3	4	3	2
# Hidden Features	256	256	256	128	256	64
MP Type	GINE	GINE	GINE	AGAT	GINE	GINE
Aggregation	Sum	Sum	Sum	Sum	Sum	Sum
Activation	ReLU	ReLU	ReLU	ReLU	ReLU	ReLU
Tree Activation	LeakyReLU	LeakyReLU	LeakyReLU	LeakyReLU	LeakyReLU	LeakyReLU
Dropout	0.1	0.2	0.1	0	0.2	0
Edge Dropout	0.1	0.2	0	0	0.2	0
PE						
PE Dim.	16	20	16	16	10	10
PE Type	DEG, -	DEG, -	DEG, -	DEG, LapPE	DEG, LapPE	DEG, LapPE
PE Merge	Concat	Concat	Concat	Concat	Concat	Concat
Transformer						
# Hidden Features	512	512	512	256	256	128
# FFN Hidden Features	1024	512	512	512	256	128
# Layers	4	4	4	4	4	3
# Heads	8	8	8	8	8	4
Activation	ReLU	ReLU	ReLU	ReLU	ReLU	ReLU
Dropout	0.2	0.5	0.2	0.3	0.5	0.2
Readout						
# Hidden Features	768	768	768	256	768	128
Activation	ReLU	ReLU	ReLU	ReLU	ReLU	ReLU

For GraphGPS, we apply 6 layers with multihead attention, with the random walk position encoding dimension of 20, with the dropout rate of 0.2, and the summation aggregation. We use AdamW optimizer with learning rate of 0.001 and the ROP scheduler.

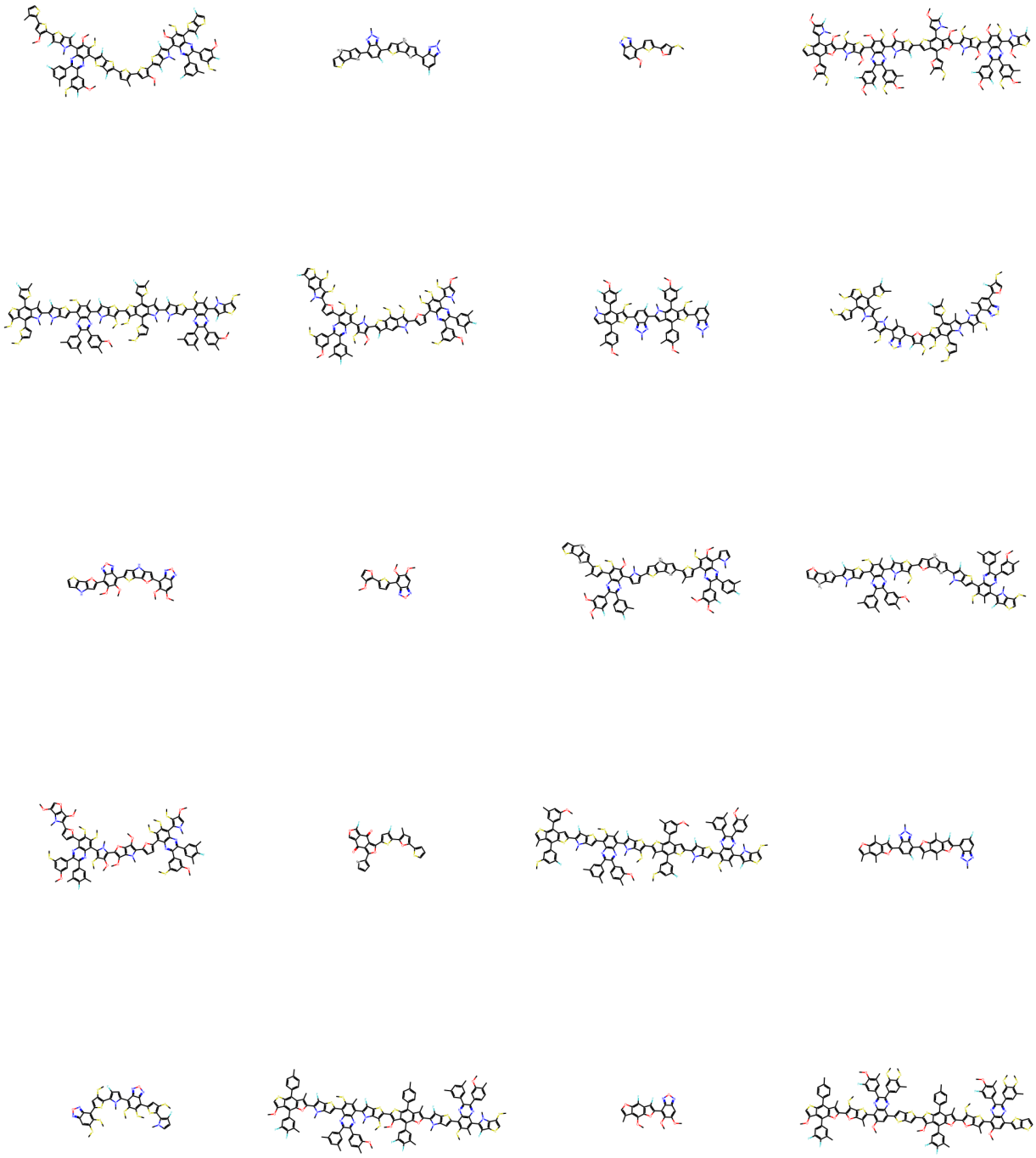

Figure 2: Illustration of molecules included in the OPDA dataset, samples are randomly drawn.

C Training Curves

To better demonstrate the training dynamics of SubFromer-Spec model, we plotted the training curves in this section for visualization.

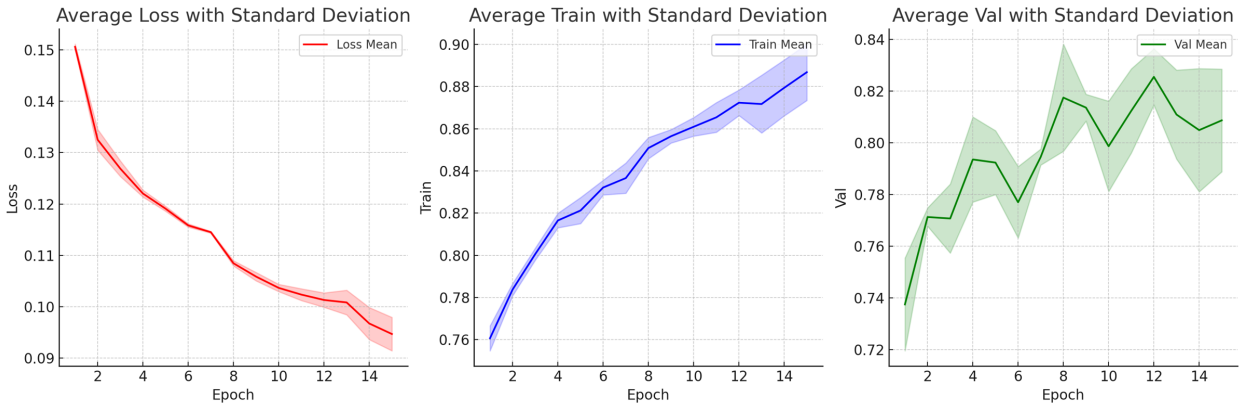

Figure 3: Training curve of the SubFormer-Spec on HIV dataset.

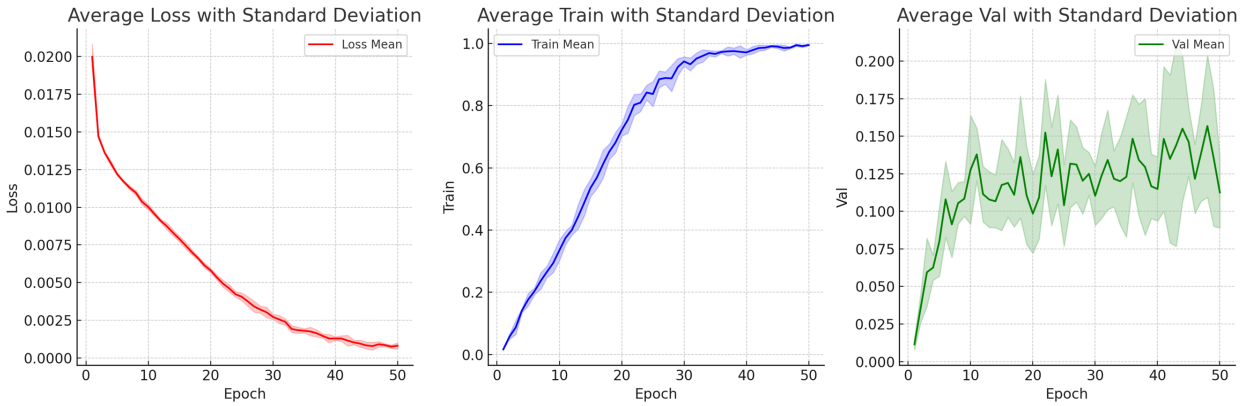

Figure 4: Training curve of the SubFormer-Spec on MUV dataset.

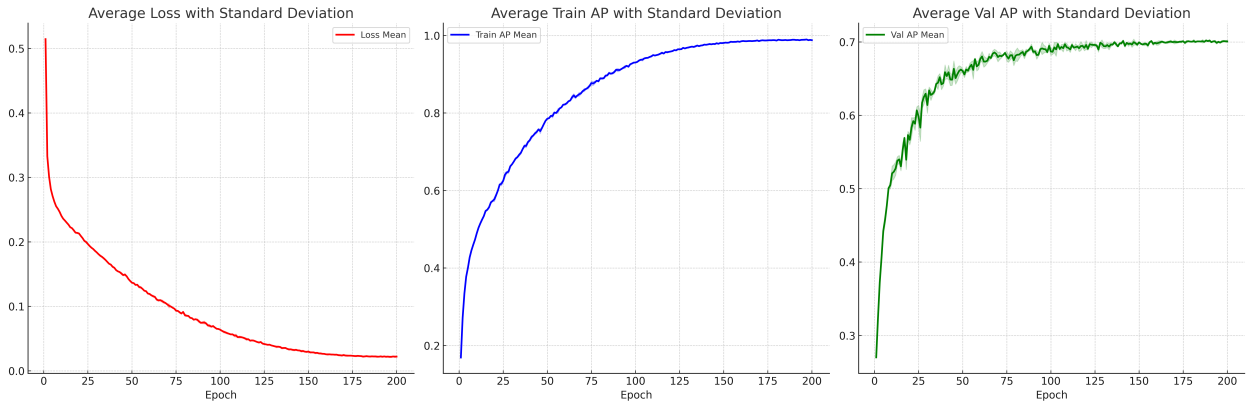

Figure 5: Training curve of the SubFormer-Spec on Peptides-Func dataset.

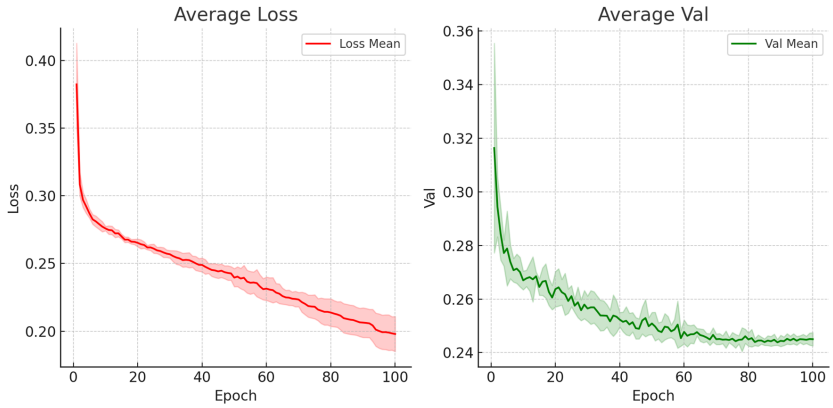

Figure 6: Training curve of the SubFormer-Spec on Peptides-Struct dataset.

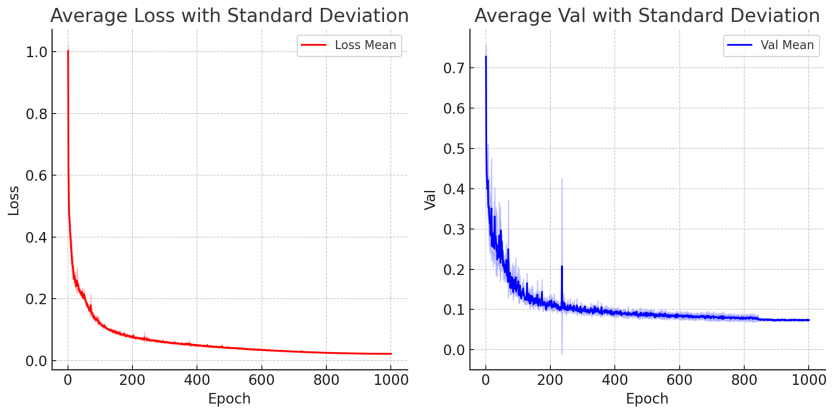

Figure 7: Training curve of the SubFormer-Spec on ZINC dataset.

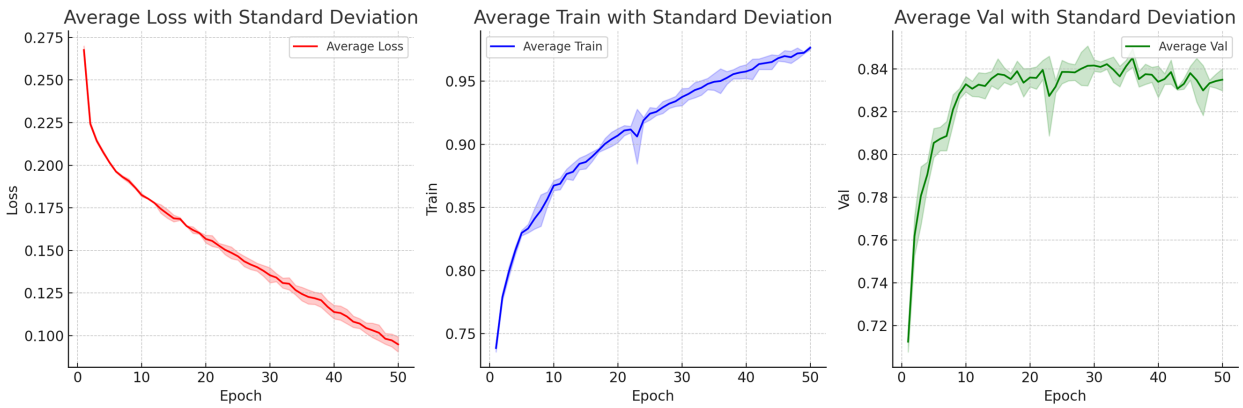

Figure 8: Training curve of the SubFormer-Spec on TOX21 dataset.

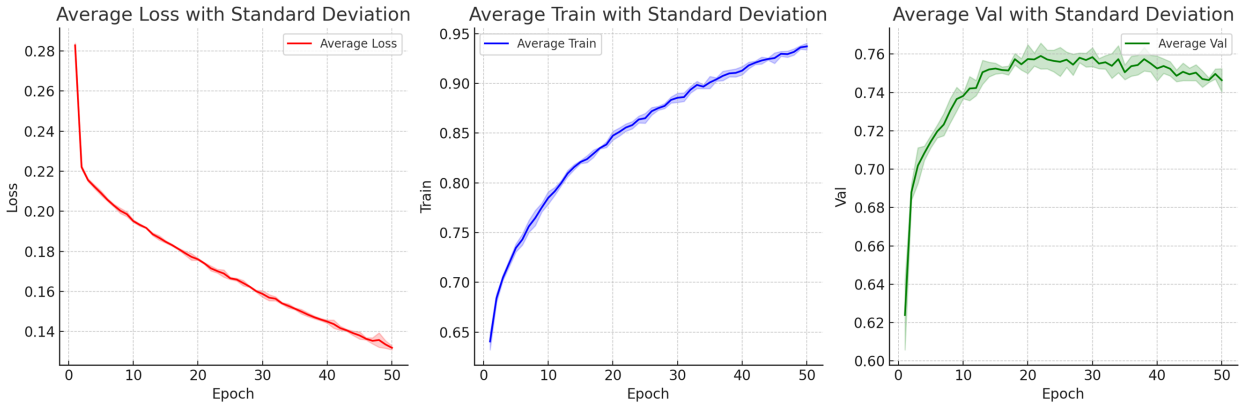

Figure 9: Training curve of the SubFormer-Spec on TOXCAST dataset.