

Learning Advanced Self-Attention for Linear Transformers in the Singular Value Domain

Hyowon Wi¹, Jeongwhan Choi² and Noseong Park¹

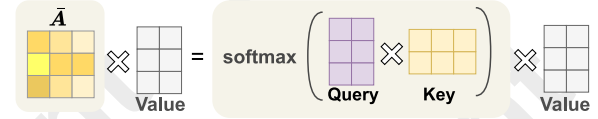
¹Korea Advanced Institute of Science and Technology (KAIST)

²Yonsei University

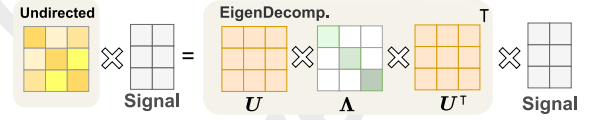
hyowon.wi@kaist.ac.kr, jeongwhan.choi@yonsei.ac.kr, noseong@kaist.ac.kr

Abstract

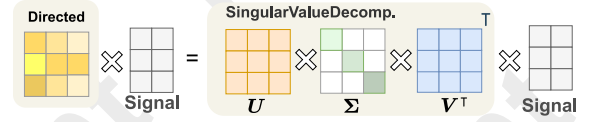
Transformers have demonstrated remarkable performance across diverse domains. The key component of Transformers is self-attention, which learns the relationship between any two tokens in the input sequence. Recent studies have revealed that the self-attention can be understood as a normalized adjacency matrix of a graph. Notably, from the perspective of graph signal processing (GSP), the self-attention can be equivalently defined as a simple graph filter, applying GSP using the value vector as the signal. However, the self-attention is a graph filter defined with only the first order of the polynomial matrix, and acts as a low-pass filter preventing the effective leverage of various frequency information. Consequently, existing self-attention mechanisms are designed in a rather simplified manner. Therefore, we propose a novel method, called **Attentive Graph Filter (AGF)**, interpreting the self-attention as learning the graph filter in the singular value domain from the perspective of graph signal processing for directed graphs with the linear complexity w.r.t. the input length. In our experiments, we demonstrate that AGF achieves state-of-the-art performance on various tasks, including Long Range Arena benchmark and time series classification. Code is available at <https://github.com/hyowonwi/agf>.



(a) Vanilla self-attention in Transformers



(b) Graph signal processing on undirected graph



(c) Graph signal processing on directed graph

Figure 1: Illustration of the vanilla self-attention in Transformers and graph signal processing: (a) shows the softmax operation after the dot-product of query and key vectors, followed by the multiplication with the value vector; (b) and (c) show graph signal processing in undirected and directed graphs, respectively. For undirected graphs, the signal is filtered in the eigenvalue domain, while for directed graphs, the signal is filtered in the *singular value domain*.

1 Introduction

Transformers [Vaswani *et al.*, 2017] have achieved great success in many fields, including computer vision [Touvron *et al.*, 2021; Liu *et al.*, 2021], time series analysis [Li *et al.*, 2019; Wu *et al.*, 2021; Zhou *et al.*, 2021], natural language processing [Nangia and Bowman, 2018; Maas *et al.*, 2011; Radford *et al.*, 2019; Devlin *et al.*, 2019], and many other works [Wang *et al.*, 2020; Shin *et al.*, 2024; Kim *et al.*, 2024; Yu *et al.*, 2024]. Many researchers agree that the self-attention mechanism plays a major role in the powerful performance of Transformers. The self-attention mechanism employs a dot-product operation to calculate the similarity between any two tokens of the input sequence, allowing all other tokens to be attended when updating one token.

Linear Transformers approximate the self-attention.

However, the self-attention requires quadratic complexity over the input length to calculate the cosine similarity between any two tokens. This makes the self-attention difficult to apply to inputs with long lengths. In order to process long sequences, therefore, reducing the complexity of self-attention has become a top-priority goal, leading to the proposal of approximating the self-attention with linear complexity [Zaheer *et al.*, 2020; Beltagy *et al.*, 2020; Wang *et al.*, 2020; Katharopoulos *et al.*, 2020; Choromanski *et al.*, 2020; Shen *et al.*, 2021; Xiong *et al.*, 2021; Qin *et al.*, 2022; Chen *et al.*, 2023]. However, existing self-attention with linear complexity aims to create a matrix that is close to the original self-attention map $\bar{\mathbf{A}}$.

The self-attention is a low-pass filter (see Thm. 1). The self-attention map is a matrix that represents the relationship between every pair of tokens as scores normalized to probability by softmax. Considering each token as a node and the attention score as an edge weight, self-attention is considered as a normalized adjacency matrix from the perspective of the graph [Shi *et al.*, 2022; Wang *et al.*, 2022; Choi *et al.*, 2024]. Therefore, the self-attention plays an important role in the message passing scheme by determining which nodes to exchange information with. Given that the self-attention operates as a normalized adjacency matrix, it is intuitively aligned with graph signal processing (GSP) [Ortega *et al.*, 2018; Marques *et al.*, 2020; Chien *et al.*, 2021; Defferrard *et al.*, 2016], which employs graph structures to analyze and process signals. Fig. 1 illustrates the self-attention in Transformers and signal filtering in GSP. As depicted in Fig. 1 (a), the original self-attention takes the softmax of the output from the dot product of the query vector and the key vector, and then multiplies it by the value vector. Fig. 1 (b) illustrates a general GSP method that applies the graph Fourier transform to the signal, conducts filtering in the spectral domain, and subsequently restores it to the original signal domain. In GSP, signals are filtered through the graph filters, which are generally approximated by a matrix polynomial expansion. The self-attention mechanism in Transformers can be viewed as a graph filter \mathbf{H} defined with only the first order of the polynomial matrix, i.e., $\mathbf{H} = \bar{\mathbf{A}}$. Furthermore, since the self-attention is normalized by softmax and functions as a low-pass filter (see Theorem. 1), the high-frequency information in the value vector is attenuated, preventing the effective leverage of various frequency information. Consequently, existing self-attention mechanisms are designed in a rather simplified manner.

Our proposed linear Transformer learns an advanced self-attention (see Thm. 2). Although the approximation of linear Transformers is successful, what they are doing is simply a low-pass filtering. Therefore, to increase the expressive power of linear Transformers, we propose a more generalized GSP-based self-attention, called **Attentive Graph Filter (AGF)**. We interpret the value vector of Transformers as a signal and redesign the self-attention as a graph filter. However, the existing self-attention mechanism possesses two problems: i) since the self-attention is based on directed graphs, the graph Fourier transform through eigendecomposition is not always guaranteed, and ii) the attention map changes for every batch, making it too costly to perform the graph Fourier transform every batch. In order to address the first problem, therefore, we design a self-attention layer based on the GSP process in the *singular value domain* (see Fig. 1 (c)). The singular value decomposition (SVD) has been used recently for the GSP in directed graphs and can substitute the eigendecomposition [Maskey *et al.*, 2023]. In order to address the second problem, we directly learn the singular values and vectors instead of explicitly decomposing the self-attention map or any matrix. Since our proposed self-attention layer directly learns in the *singular value domain* by generating singular vectors and values using a neural network, our proposed method has a linear complexity of $O(nd^2)$. Therefore,

our method efficiently handles inputs with long sequences.

Our contributions can be summarized as follows:

1. We propose an advanced self-attention mechanism based on the perspective of signal processing on directed graphs, called **Attentive Graph Filter (AGF)**, motivated that the self-attention is a simple graph filter and acts as a low pass filter in the singular value domain.
2. AGF learns a sophisticated graph filter directly in the singular value domain with linear complexity w.r.t. input length, which incorporates both low and high-frequency information from hidden representations.
3. The experimental results for time series, long sequence modeling and image domains demonstrate that AGF outperforms existing linear Transformers.
4. As a side contribution, we conduct additional experiments to show that AGF effectively mitigates the over-smoothing problem in deep Transformer models, where the hidden representations of tokens to become indistinguishable from one another.

2 Background

2.1 Self-attention in Transformer

A key operation of Transformers is the self-attention which allows them to learn the relationship among tokens. The self-attention mechanism, denoted as SA: $\mathbb{R}^{n \times d} \rightarrow \mathbb{R}^{n \times d}$, can be expressed as follows:

$$\text{SA}(\mathbf{X}) = \text{softmax}\left(\frac{\mathbf{X}\mathbf{W}_q(\mathbf{X}\mathbf{W}_k)^\top}{\sqrt{d}}\right)\mathbf{X}\mathbf{W}_v = \bar{\mathbf{A}}\mathbf{X}\mathbf{W}_v, \quad (1)$$

where $\mathbf{X} \in \mathbb{R}^{n \times d}$ is the input feature and $\bar{\mathbf{A}} \in \mathbb{R}^{n \times n}$ is the self-attention matrix. $\mathbf{W}_q \in \mathbb{R}^{d \times d}$, $\mathbf{W}_k \in \mathbb{R}^{d \times d}$, and $\mathbf{W}_v \in \mathbb{R}^{d \times d}$ are the query, key, and value trainable parameters, respectively, and d is the dimension of token. The self-attention effectively learns the interactions of all token pairs and has shown reliable performance in various tasks.

However, in the case of the existing self-attention, a dot-product is used to calculate the attention score for all token pairs. To construct the self-attention matrix $\bar{\mathbf{A}} \in \mathbb{R}^{n \times n}$, the matrix multiplication with query and key parameters mainly causes a quadratic complexity of $O(n^2d)$. Therefore, it is not suitable if the length of the input sequence is large. This is one of the major computational bottlenecks in Transformers. For instance, BERT [Devlin *et al.*, 2019], one of the state-of-the-art Large Language Model (LLM), needs 16 TPUs for pre-training and 64 TPUs with large models.

2.2 Linear Transformer

To overcome the quadratic computational complexity of the self-attention, efficient Transformer variants have been proposed in recent years. Recent research focuses on reducing the complexity of the self-attention in two streams. The first research line is to replace the softmax operation in the self-attention with other operations. For simplicity, we denote $\text{softmax}(\mathbf{X}\mathbf{W}_{qry}(\mathbf{X}\mathbf{W}_{key})^\top)$ as $\text{softmax}(\mathbf{Q}\mathbf{K}^\top)$. [Wang *et al.*, 2020] introduce projection layers that map

value and key vectors to low dimensions. [Katharopoulos *et al.*, 2020] interprets softmax as kernel function and replace the similarity function with $\text{elu}(\mathbf{x}) + 1$. [Choromanski *et al.*, 2020] approximates the self-attention matrix with random features. [Shen *et al.*, 2021] decomposes the $\text{softmax}(\mathbf{QK}^\top)$ into $\text{softmax}(\mathbf{Q})\text{softmax}(\mathbf{K}^\top)$, which allows to calculate $\text{softmax}(\mathbf{K}^\top)\mathbf{V}$ first, reducing the complexity from $\mathcal{O}(n^2d)$ to $\mathcal{O}(nd^2)$. [Qin *et al.*, 2022] replaces softmax with a linear operator and adopts a cosine-based distance re-weighting mechanism. [Xiong *et al.*, 2021] adopts Nyström method by down sampling the queries and keys in the attention matrix. [Chen *et al.*, 2023] employs an asymmetric kernel SVD motivated by low-rank property of the self-attention. However, these approaches sacrifice the performance to directly reduce quadratic complexity to linear complexity.

The second research line is to introduce sparsity in the self-attention. [Zaheer *et al.*, 2020] introduces a sparse attention mechanism optimized for long document processing, combining local, random, and global attention to reduce computational complexity while maintaining performance. [Kitaev *et al.*, 2020] use locality-sensitive hashing and reversible feed forward network for sparse approximation, while requiring to re-implement the gradient back propagation. [Beltagy *et al.*, 2020] employ the self-attention on both a local context and a global context to introduce sparsity. [Zeng *et al.*, 2021] take a Bernoulli sampling attention mechanism based on locality sensitive hashing. However, since they do not directly reduce the complexity to linear, they also suffer a large performance degradation, while having only limited additional speed-up.

2.3 Graph Convolutional Filter

The graph signal processing (GSP) can be considered as a generalized concept of the discrete signal processing (DSP). In the definition of DSP, the discrete signal with length n is represented by the vector $\mathbf{x} \in \mathbb{R}^n$. Then for the signal filter $\mathbf{g} \in \mathbb{R}^n$ that transforms \mathbf{x} , the convolution operation $\mathbf{x} * \mathbf{g}$ is defined as follows:

$$y_i = \sum_{j=1}^n \mathbf{x}_j \mathbf{g}_{i-j}, \quad (2)$$

where the index i indicates the i -th element of each vector. GSP extends DSP to signal samples indexed by nodes of arbitrary graphs. Then we define the shift-invariant graph convolution filters \mathbf{H} with a polynomial of graph shift operator \mathbf{S} as follows:

$$\mathbf{y} = \mathbf{H}\mathbf{x} = \sum_{k=0}^K w_k \mathbf{S}^k \mathbf{x}, \quad (3)$$

where K is the maximum order of polynomial and $w_k \in [-\infty, \infty]$ is a coefficient. The graph filter is parameterized as the truncated expansion with the order of K . The most commonly used graph shift operators in GSP are adjacency and Laplacian matrices. Note that Eq. (3) applies to any directed or undirected adjacency matrix [Ortega *et al.*, 2018; Marques *et al.*, 2020]. However, Eq. (3) requires non-trivial matrix power computation. Therefore, we rely on SVD to use the more efficient way in Eq. (5).

3 Proposed Method

3.1 Self-attention as a graph filter

The self-attention learns the relationship among all token pairs. From a graph perspective, each token can be interpreted as a graph node and each self-attention score as an edge weight. Therefore, self-attention produces a special case of the normalized adjacency matrix [Shi *et al.*, 2022; Wang *et al.*, 2022] and can be analyzed from the perspective of graph signal processing (GSP). In GSP, the low-/high-frequency components of a signal \mathbf{x} are defined using the Discrete Fourier Transform (DFT) \mathcal{F} and its inverse \mathcal{F}^{-1} . Let $\bar{\mathbf{x}} = \mathcal{F}\mathbf{x}$ denote the spectrum of \mathbf{x} . Then, $\bar{\mathbf{x}}_{\text{lfc}} \in \mathbb{C}^c$ contains the c lowest-frequency components of $\bar{\mathbf{x}}$, and $\bar{\mathbf{x}}_{\text{hfc}} \in \mathbb{C}^{n-c}$ contains the remaining higher-frequency components. The low-frequency components (LFC) of \mathbf{x} are given as $\text{LFC}[\mathbf{x}] = \mathcal{F}^{-1}(\bar{\mathbf{x}}_{\text{lfc}}) \in \mathbb{R}^n$, and the high-frequency components (HFC) are defined as $\text{HFC}[\mathbf{x}] = \mathcal{F}^{-1}(\bar{\mathbf{x}}_{\text{hfc}}) \in \mathbb{R}^n$. Here, the DFT \mathcal{F} projects \mathbf{x} into the frequency domain, and \mathcal{F}^{-1} reconstructs \mathbf{x} from its spectrum. The Fourier basis $\mathbf{f}_j = [e^{2\pi i(j-1) \cdot 0}, e^{2\pi i(j-1) \cdot 1}, \dots, e^{2\pi i(j-1)(n-1)}]^\top / \sqrt{n}$ is used in computing \mathcal{F} , where j denotes the j -th row. In GSP, the adjacency matrix functions as a low-pass filter, using the edge weights to aggregate information from nodes attenuates the high-frequency information of the nodes. In other words, the self-attention also acts as a low-pass filter within Transformers, and it is theoretically demonstrated below.

Theorem 1 (Self-attention is a low-pass filter). *Let $\mathbf{M} = \text{softmax}(\mathbf{Z})$ for any matrix $\mathbf{Z} \in \mathbb{R}^{n \times n}$. Then \mathbf{M} inherently acts as a low pass filter. For all $\mathbf{x} \in \mathbb{R}^n$, in other words, $\lim_{t \rightarrow \infty} \|\text{HFC}[\mathbf{M}^t(\mathbf{x})]\|_2 / \|\text{LFC}[\mathbf{M}^t(\mathbf{x})]\|_2 = 0$*

The proof of Theorem 1 is in [Wi *et al.*, 2025, Appendix D]. As the self-attention is normalized by softmax, the self-attention functions as a low-pass filter. Hence, Transformers are unable to sufficiently leverage a various scale of frequency information, which reduces the expressive power of Transformers.

Inspired by this observation, we redesign a graph filter-based self-attention from the perspective of GSP. As mentioned earlier, since the adjacency matrix can serve as a graph-shift operator, it is reasonable to interpret the self-attention as a graph-shift operator, $\mathbf{S} = \bar{\mathbf{A}}$. Moreover, the self-attention block of the Transformer is equivalent to defining a simple graph filter $\mathbf{H} = \bar{\mathbf{A}}$ and applying GSP to the value vector, treated as a signal. Therefore, in Eq. (3), we can design a more complex graph filter through the polynomial expansion of the self-attention.

Note that when we interpret the self-attention $\bar{\mathbf{A}}$ as a graph, it has the following characteristics: i) The self-attention is an asymmetric directed graph, and ii) all nodes in the graph are connected to each other since the self-attention calculates the relationships among the tokens. Then we can derive that the self-attention is a special case of the symmetrically normalized adjacency (SNA) as $\bar{\mathbf{A}} = \mathbf{D}^{-1}\mathbf{A}$ where \mathbf{A} is an adjacency matrix and \mathbf{D} is a degree matrix of nodes. In particular, SNA is one of the most popular forms for directed GSP [Maskey *et al.*, 2023].

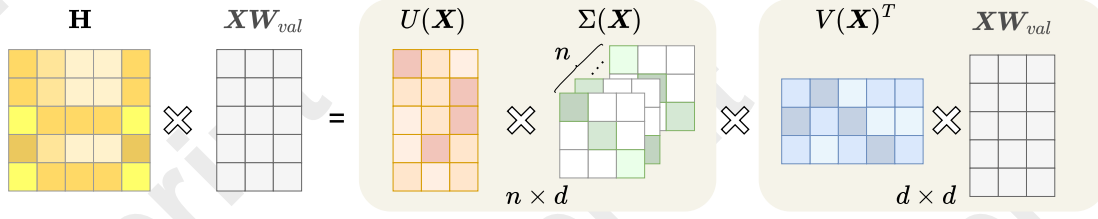


Figure 2: The proposed AGF performs the directed GSP in the singular value domain by learning $U(\mathbf{X})$, $\Sigma(\mathbf{X})$, and $V(\mathbf{X})$ (cf. Eqs. (8) to (10)). The n different sets of singular values in $\Sigma(\mathbf{X})$ are used for token-specific processing. In other words, n different graph filters are used for n different tokens in order to increase the representation learning capability of AGF.

3.2 Polynomial graph filter

When approximating the graph filter with a matrix polynomial, $k - 1$ matrix multiplications are required to calculate up to the k -th order (cf. Eq. (3)), which requires a large computational cost. Therefore, to reduce the computational complexity, we avoid matrix multiplications by directly learning the graph filter in the spectral domain. In the case of an undirected graph, a filter can be learned in the spectral domain by performing the graph Fourier transform through eigendecomposition. In general, however, the eigendecomposition is not guaranteed for directed graphs. The GSP through SVD, therefore, is often used [Maskey *et al.*, 2023] if i) a directed graph $\bar{\mathbf{A}}$ is SNA and ii) its singular values are non-negative and within the unit circle, i.e., $\|\bar{\mathbf{A}}\| \leq 1$. For $\bar{\mathbf{A}}$ and its SVD $\bar{\mathbf{A}} = \mathbf{U}\Sigma\mathbf{V}^T$, an α -power of the symmetrically normalized adjacency is defined as:

$$\bar{\mathbf{A}}^\alpha := \mathbf{U}\Sigma^\alpha\mathbf{V}^T, \quad (4)$$

where $\alpha \in \mathbb{R}$ [Maskey *et al.*, 2023]. Therefore, we can define the graph filter \mathbf{H} as follows:

$$\mathbf{y} = \mathbf{H}\mathbf{x} = g_\theta(\bar{\mathbf{A}})\mathbf{x} = \mathbf{U}g_\theta(\Sigma)\mathbf{V}^T\mathbf{x} = \mathbf{U}\left(\sum_{k=0}^K \theta_k \Sigma^k\right)\mathbf{V}^T\mathbf{x}, \quad (5)$$

where $\theta \in \mathbb{R}^n$ is a vector for singular value coefficients. Therefore, a spectral filter can be defined as a truncated expansion with K -th order polynomials. In other words, unlike directly performing the matrix polynomial as in Eq. (3), the computational cost is significantly reduced by K times element-wise multiplying of the singular values, which are represented as a diagonal matrix.

However, the polynomial expansion in Eq. (5) is parameterized with monomial basis, which is unstable in terms of its convergence since the set of bases is non-orthogonal. Therefore, for stable convergence, a filter can be designed using an orthogonal basis. Note that we have the flexibility to apply any basis when using the polynomial expansion for learning graph filters. In this work, we adopt the Jacobi expansion [Askey and Wilson, 1985], one of the most commonly used polynomial bases. Furthermore, Jacobi basis is a generalized form of classical polynomial bases such as Chebyshev [Defferrard *et al.*, 2016] and Legendre [McCarthy *et al.*, 1993], offering strong expressiveness in the graph filter design. Detailed formulas are provided in [Wi *et al.*, 2025, Appendix F]. Therefore, we can define the graph polynomial

filter as follows:

$$g_\theta(\Sigma) = \sum_{k=0}^K \theta_k T_k(\Sigma), \quad (6)$$

where $T_k(\cdot)$ is a specific polynomial basis of order k .

3.3 Attentive Graph Filter

In order to use Eq. (6), however, we need to decompose the adjacency matrix $\bar{\mathbf{A}}$, which incurs non-trivial computation. Therefore, we propose to directly learn a graph filter in the singular value domain (instead of learning an adjacency matrix, i.e., a self-attention matrix, and decomposing it). Therefore, as shown in Fig. 2, we propose our attentive graph filter (AGF) as follows:

$$\text{AGF}(\mathbf{X}) = \mathbf{H}\mathbf{X}\mathbf{W}_v = \mathbf{U}(\mathbf{X})\Sigma(\mathbf{X})\mathbf{V}(\mathbf{X})^T\mathbf{X}\mathbf{W}_v, \quad (7)$$

$$\mathbf{U}(\mathbf{X}) = \rho(\mathbf{X}\mathbf{W}_U) \in \mathbb{R}^{n \times d}, \quad (8)$$

$$\Sigma(\mathbf{X}) = \sum_{k=0}^K \theta_k T_k(\text{diag}(\sigma(\mathbf{X}\mathbf{W}_\Sigma))) \in \mathbb{R}^{n \times d \times d}, \quad (9)$$

$$\mathbf{V}(\mathbf{X})^T = \rho((\mathbf{X}\mathbf{W}_V)^T) \in \mathbb{R}^{d \times n}, \quad (10)$$

where $\mathbf{W}_U, \mathbf{W}_\Sigma, \mathbf{W}_V \in \mathbb{R}^{d \times d}$ are learnable matrices, ρ is a softmax, and σ is a sigmoid. Our proposed model does not apply SVD directly on the computed self-attention or other matrices. Instead, the learnable singular values $\sigma(\mathbf{X}\mathbf{W}_\Sigma)$ and orthogonally regularized singular vectors $\mathbf{U}(\mathbf{X})$ and $\mathbf{V}(\mathbf{X})$ are generated by neural network. The singular values are then filtered by the graph filter, denoted as $\Sigma(\mathbf{X})$. To ensure that the elements of the singular value matrix are non-negative and within the unit circle, the sigmoid function is applied to the matrix. Moreover, we observe that the softmax of singular vectors enhances the stability of learning.

We construct our graph filter using the generated singular values, leveraging the Jacobi expansion as an orthogonal polynomial basis. If the trainable coefficients θ_k is allowed to take negative values and learned adaptively, the graph filter can pass low/high-frequency components of the value vector. Therefore, AGF functions as a graph filter that leverages various frequency information from the value vector. Furthermore, unlike the adjacency matrix that remains unchanged in GCNs, the self-attention matrix changes with each batch. To enhance the capacity for addressing these dynamics, AGF incorporates a token-specific graph filter, characterized by n

different sets of singular values. This allows to leverage the token-specific frequency information in the singular value domain, increasing the capability to handle complex dynamics in hidden representation.

3.4 Objective Function

In the definition of SVD, $U(\mathbf{X})$ is column orthogonal, $V(\mathbf{X})$ is row orthogonal, and $\Sigma(\mathbf{X})$ is a rectangular diagonal matrix with non-negative real numbers. When we train the proposed model, strictly constraining $U(\mathbf{X})$ and $V(\mathbf{X})$ to be orthogonal requires a high computational cost. Instead, we add a regularization on them since these matrices generated by neural network can be trained to be orthogonal as follows:

$$\mathcal{L}_{ortho} = \frac{1}{n^2} (\| (U(\mathbf{X})^\top U(\mathbf{X}) - \mathbf{I} \| + \| (V(\mathbf{X})V(\mathbf{X})^\top - \mathbf{I} \|), \quad (11)$$

where $\mathbf{I} \in \mathbb{R}^{d \times d}$ is an identity matrix. Therefore, our joint learning objective \mathcal{L} is as follows:

$$\mathcal{L} = \mathcal{L}_{transformer} + \gamma \mathcal{L}_{ortho}, \quad (12)$$

where $\mathcal{L}_{transformer}$ is an original objective function for Transformers. The hyperparameters γ controls the trade-off between the loss and the regularization.

3.5 Time and Space Complexities of AGF

Since our AGF is based on the concept of SVD, it is not restricted by softmax for calculating attention scores. Therefore, $U(\mathbf{X})$, $\Sigma(\mathbf{X})$, and $V(\mathbf{X})$ generated by neural network can be freely multiplied according to the combination law of matrix multiplication. First, since $\Sigma(\mathbf{X})$ is a diagonal matrix, by performing element-wise multiplication with $U(\mathbf{X})$ and the diagonal elements of $\Sigma(\mathbf{X})$, $(n \times d)$ matrix is calculated with a time complexity of $\mathcal{O}(nd)$. Next, by multiplying $V(\mathbf{X})$ and the value vector, $(d \times d)$ matrix is calculated with a time complexity of $\mathcal{O}(nd^2)$. Finally, by multiplying the outputs of steps 1 and 2, the final output is $(n \times d)$ matrix with a time complexity of $\mathcal{O}(nd^2)$. Therefore, the time complexity is $\mathcal{O}(nd^2)$ and the space complexity is $\mathcal{O}(nd + d^2)$.

3.6 Properties of AGF

How to use high-frequency information. In GSP, the characteristics of the graph filter are determined by the learned coefficients θ_k of the signal. These coefficients allow the graph filter to function as a low-pass, high-pass, or combined-pass filter, depending on the specific needs of each task [Defferrard *et al.*, 2016; Marques *et al.*, 2020; Chien *et al.*, 2021], demonstrated by following theorem:

Theorem 2 (Adapted from [Chien *et al.*, 2021]). *Assume that the graph G is connected. If $\theta_k \geq 0$ for $\forall k \in \{0, 1, \dots, K\}$, $\sum_{k=0}^K \theta_k = 1$ and $\exists k' > 0$ such that $\theta_{k'} > 0$, then $g_\theta(\cdot)$ is a low-pass graph filter. Also, if $\theta_k = (-\alpha)^k$, $\alpha \in (0, 1)$ and K is large enough, then $g_\theta(\cdot)$ is a high-pass graph filter.*

The proof is in [Wi *et al.*, 2025, Appendix E]. Theorem 2 indicates that if the coefficient θ_k of a graph filter can have negative values, and learned adaptively, the graph filter will pass low and high frequency signals appropriately. This flexibility is crucial for effectively processing signals with varying

frequency components. Similarly, AGF operates as a filter that modulates frequency information in the singular value domain through the generated singular values and singular vectors. This approach enables AGF to dynamically adjust the frequency components of the signal, providing a more tailored and efficient filtering process. Therefore, unlike conventional Transformers, AGF can appropriately incorporate both low and high frequencies for each task, thereby enhancing the expressive power and adaptability of Transformers.

Comparison with existing linear self-attention methods.

We explain that while our AGF addresses the computational inefficiencies inherent in the vanilla self-attention like existing linear self-attention studies, we take a different approach from them. Instead of using explicit SVDs, our AGF reinterprets self-attention through a GSP lens, using the learnable SVD to learn graph filters directly from the spectral domain of directed graphs. Linformer [Wang *et al.*, 2020], the most prominent representative of linear self-attention, approximates the vanilla self-attention through dimensionality reduction, and Nyströmformer [Xiong *et al.*, 2021], which reduces to linear complexity with a kernel decomposition method, also efficiently approximates the full self-attention matrix with the Nyström method. Singularformer [Wu *et al.*, 2023], a closely related approach, uses a parameterized SVD and linearize the calculation of self-attention. However, like existing linear Transformers, it approximates the original self-attention, which is inherently a low-pass filter. Thus, to the best of our knowledge, existing linear self-attention methods focus on approximating the self-attention and reducing it to linear complexity, whereas our AGF approximates a graph filter rather than the self-attention. This allows AGF to use the token-specific graph filter to improve model representation within the singular value domain.

4 Experiments

4.1 Time Series Classification

Experimental settings. To evaluate the performance of AGF, we employ UEA Time Series Classification Archive [Bagnall *et al.*, 2018] which is the benchmark on temporal sequences. Strictly following [Wu *et al.*, 2022], we report accuracy for 10 multivariate datasets preprocessed according to [Zerveas *et al.*, 2021]. We adopt 2-layer Transformer as backbone with 512 hidden dimension on 8 heads and 64 embedding dimension of self-attention. The experiments are conducted on 1 GPU of NVIDIA RTX 3090. The detailed descriptions are in [Wi *et al.*, 2025, Appendix H.1].

Experimental results. Table 1 summarizes the test accuracy of AGF and the state-of-the-art linear Transformer models on the UEA time series classification task. We observe that AGF achieves an average accuracy of 75.1, outperforming the vanilla Transformer and other linear Transformers by large margins across various datasets. This performance gap underscores the effectiveness of our approach in leveraging advanced graph filter-based self-attention to enhance the expressive power of Transformers.

	EC	FD	HW	HB	JV	PEMS-SF	SRSCP1	SRSCP2	SAD	UWGL	Avg
Transformer	32.7	67.3	32.0	76.1	98.7	82.1	92.2	53.9	98.4	85.6	71.9
LinearTransformer	31.9	67.0	34.7	76.6	99.2	82.1	92.5	56.7	98.0	85.0	72.4
Reformer	31.9	68.6	27.4	77.1	97.8	82.7	90.4	56.7	97.0	85.6	71.5
Longformer	32.3	62.6	39.6	78.0	98.9	83.8	90.1	55.6	94.4	87.5	72.0
Performer	31.2	67.0	32.1	75.6	98.1	80.9	91.5	56.7	98.4	85.3	71.9
YOSO-E	31.2	67.3	30.9	76.5	98.6	85.2	91.1	53.9	98.9	88.4	72.2
Cosformer	32.3	64.8	28.9	77.1	98.3	83.2	91.1	55.0	98.4	85.6	71.5
SOFT	33.5	67.1	34.7	75.6	99.2	80.9	91.8	55.6	98.8	85.0	72.2
Flowformer	33.8	67.6	33.8	77.6	98.9	83.8	92.5	56.1	98.8	86.6	73.0
Primalformer	33.1	67.1	29.6	76.1	98.3	89.6	92.5	57.2	100.0	86.3	73.0
AGF	36.1	69.9	33.5	79.0	99.5	91.3	93.5	58.9	100.0	89.4	75.1

Table 1: Performance comparison on UEA time series classification. Abbreviations are as follows: EthanolConcentration (EC), FaceDetection (FD), HandWriting (HW), HearBeat (HB), JapaneseVowels (JV), PEMS-SF, SelfRegulation SCP1 (SRSCP1), SelfRegulation SCP2 (SRSCP2), SpokenArabicDigits (SAD), and UWaveGesture Library (UWGL).

	ListOps	Text	Retrieval	Image	Pathfinder	Avg
Transformer	37.1	65.0	79.4	38.2	74.2	58.8
Reformer	19.1	64.9	78.6	43.3	69.4	55.1
Performer	18.8	63.8	78.6	37.1	69.9	53.6
Singularformer	18.7	61.8	76.7	35.3	55.8	49.7
Linformer	37.3	55.9	79.4	37.8	67.6	55.6
Nyströmformer	37.2	65.5	79.6	41.6	70.9	59.0
Longformer	37.2	64.6	81.0	39.1	73.0	59.0
YOSO-E	37.3	64.7	81.2	39.8	72.9	59.2
Primalformer	37.3	61.2	77.8	43.0	68.3	57.5
AGF	38.0	64.7	81.4	42.4	74.0	60.1

Table 2: Performance comparison on LRA benchmark

4.2 Long Range Arena Benchmark

Experimental settings. We evaluate AGF on Long Range Arena (LRA) [Tay *et al.*, 2020] benchmark under long-sequence scenarios. Following [Xiong *et al.*, 2021], we train 2 layer Transformer with 128 hidden dimension, 2 heads, and 64 embedding dimension with mean pooling. The experiments are conducted on 1 GPU of NVIDIA RTX 3090. The details are in [Wi *et al.*, 2025, Appendix H.2].

Experimental results. We report the top-1 test accuracy on LRA benchmark in Table 2. Our model demonstrates the highest average performance, achieving a score of 60.1 — an improvement of 1.3 points over the vanilla Transformer. In contrast, SingularFormer, a close approach that parameterizes SVD, only functions as a low-pass filter and thus fails to achieve optimal performance. Compared with YOSO-E, a state-of-the-art linear-complexity Transformer, AGF improves the performance by a substantial margin.

4.3 Sensitivity Analyses

We conduct sensitivity studies on K and γ . Other sensitivity studies are reported in [Wi *et al.*, 2025, Appendix K].

Sensitivity study on K . We test our model by varying K on UEA time series classification, and the results are shown in Table 3. As K increases, the performance improves. However, beyond a certain threshold, increasing K results in saturation and diminished performance. Therefore, choosing an appropriate K has a significant impact on performance.

K	EC	FD	JV	PEMS-SF	SRSCP1	UWGL
3	32.3	68.2	98.9	86.7	91.1	84.1
4	31.6	68.8	99.5	89.6	92.2	84.1
6	36.1	68.3	98.9	83.8	91.1	84.4
9	30.8	69.9	99.2	83.8	93.5	86.2
10	32.3	67.5	98.9	87.3	91.1	89.4

Table 3: Effect of K on UEA classification

γ	ListOps	Text	Retrieval	Image	Pathfinder
1×10^{-1}	38.0	64.3	81.4	40.8	73.1
1×10^{-2}	36.9	64.5	79.8	42.4	73.3
1×10^{-3}	37.2	64.7	79.5	42.0	74.0
1×10^{-4}	37.0	64.2	79.4	41.0	74.0

Table 4: Effect of γ on LRA benchmark

Sensitivity study on γ . Table 4 summarizes the impact of γ on LRA benchmark. The optimal level of regularization applied to learnable singular vectors varies depending on the dataset, and we demonstrate that imposing a certain degree of regularization can enhance training stability.

4.4 Empirical Runtime

Table 5 summarize the results of the runtime and peak memory usage during the training phase. AGF consistently improves the efficiency of both time and space complexity compared to the vanilla Transformer. Specifically, for Text dataset, which have extremely long input sequences, the efficiency of AGF stands out even more. When compared with other linear complexity Transformers, our AGF shows comparable efficiency with longer sequences.

4.5 Ablation Studies

We conduct various ablation studies, and the additional results on $T(\cdot)$ and ρ are in [Wi *et al.*, 2025, Appendix L].

Effect on the graph filter. To analyze the impact of graph filter, we conduct an ablation study on the following variants: i) $\mathbf{H}_{UV\tau}$ refers to the graph filters with parameterized singular vectors and the singular values are fixed as one; ii)

	ListOps(2K)	Text(4K)	Retrieval(4K)	Image(1K)	Pathfinder(1K)	Average
Transformer	194.5/5.50	694.8/21.24	1333.7/18.72	334.5/5.88	405.5/5.88	592.6/11.44
Nystromformer	68.3/0.89	52.3/0.48	187.5/1.93	227.9/1.93	232.6/3.29	153.7/1.70
Performer	90.3/1.67	55.9/0.84	230.7/3.34	296.7/3.34	344.8/6.28	203.7/3.09
Reformer	94.1/1.64	58.1/0.82	244.2/3.29	309.1/3.29	370.7/6.09	215.2/3.03
PrimalFormer	56.5/0.69	93.6/1.37	185.3/2.99	142.9/1.39	180.0/1.52	131.7/1.59
AGF	60.8/0.88	48.4/0.51	252.3/3.95	183.3/2.15	209.3/1.89	150.8/1.90

Table 5: Running time (s/1K-steps) and the peak training memory usage (GB) on LRA benchmark

	EC	FD	HW	HB	PEMS-SF	UWGL
$\mathbf{H}_{UV\tau}$	29.7	66.6	28.2	76.6	87.3	83.8
\mathbf{H}_{SVD}	33.1	67.1	27.1	75.1	88.4	85.9
AGF	36.1	69.9	33.5	79.0	91.3	89.4

Table 6: Ablation study on the graph filter

Model	ImageNet-100	ImageNet-1K
DeiT-small	80.6	79.8
+ AGF	81.3	80.3

Table 7: Comparison of performance for DeiT-small trained on ImageNet-100 and ImageNet-1K

\mathbf{H}_{SVD} initializes the singular values to one, allowing them to be learnable from $\mathbf{H}_{UV\tau}$; and iii) AGF refers to the proposed method. Table 6 shows the result of the effect of the graph filter, and in general, these ablation models leads to suboptimal performance. However, AGF processes the generated signal through the graph filter, allowing the model to use various scales of frequency information. The graph filter enhances the capacity of the model, resulting in optimal performance and demonstrating the effectiveness of AGF.

4.6 Additional Experiments on Deep Transformer

Experimental settings. We conduct additional experiments for image classification task with ImageNet-100 [Rusakovsky *et al.*, 2015] and ImageNet-1K [Deng *et al.*, 2009] datasets and report top-1 accuracy. We adopt DeiT-small as the backbone, and trained from scratch with 300 epochs [Touvron *et al.*, 2021] with 2 GPU of NVIDIA RTX 3090. The detailed descriptions are in [Wi *et al.*, 2025, Appendix H.3].

Experimental results. Table 7 shows the top-1 accuracy on ImageNet-100 and ImageNet-1k. Our AGF effectively learns the representation in deep layers model, which has 12 layers. Notably, plugging AGF improves the performance marginally, from 80.6 to 81.3 trained on ImageNet-100 datasets and from 79.8 to 80.3 on ImageNet-1K.

Analysis on mitigating over-smoothing problem. Deep Transformers, like GCNs, suffers from over-smoothing problem, where hidden representations become similar and indistinguishable to the last layer [Kipf and Welling, 2017; Veličković *et al.*, 2018; Oono and Suzuki, 2020; Rusch *et al.*, 2023]. We previously demonstrated that the self-attention in Transformers acts as a low-pass filter attenuat-

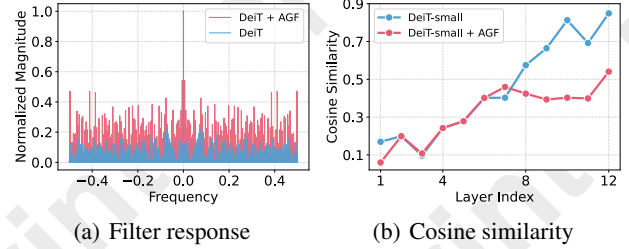


Figure 3: Filter response and cosine similarity on ImageNet-1k for DeiT-small and DeiT-small + AGF

ing high-frequency information, which is a major cause of over-smoothing [Wang *et al.*, 2022; Shi *et al.*, 2022; Choi *et al.*, 2024]. AGF mitigates this issue by effectively leveraging various scale frequency information through directly filtering signals in the singular value domain. Fig. 3 (a) illustrates the frequency information in both the vanilla DeiT (i.e., $\mathbf{H} = \bar{\mathbf{A}}$) and DeiT + AGF (i.e., $\mathbf{H} = \mathbf{U}(\mathbf{X})\Sigma(\mathbf{X})\mathbf{V}(\mathbf{X})^T$). Unlike the vanilla model, AGF better captures high-frequency information. Additionally, Fig. 3 (b) shows the cosine similarity among hidden vectors at each layer. While the cosine similarity in DeiT increases to nearly 0.9 as layers deepen, it is moderated to nearly 0.5 in DeiT + AGF. Thus, AGF prevents over-smoothing in deep Transformers by effectively leveraging diverse frequency information.

5 Conclusions

We presented AGF, which interprets the self-attention as learning graph filters in the singular value domain from the perspective of directed graph signal processing. Since the self-attention matrix can be interpreted as a directed graph, we designed a more expressive self-attention using signals directly in the singular value domain. By learning the coefficients for various polynomial bases, AGF uses diverse frequencies. Our experiments showed that AGF outperforms baselines across various tasks, and the training time and GPU usage of AGF are comparable to baseline models with linear complexity. As a side contribution, AGF mitigates the over-smoothing problem in deep Transformers.

Since our comparison scope is focused on linear Transformers, a limitation is the exploration and comparison regarding the recent state-space models [Gu *et al.*, 2021; Gu and Dao, 2023]. Exploring the potential of our method in enhancing state-space models is an intriguing avenue for future work.

Acknowledgments

This work was partly supported by Institute for Information & Communications Technology Planning & Evaluation (IITP) grants funded by the Korea government (MSIT) (No. RS-2022-II220113, Developing a Sustainable Collaborative Multi-modal Lifelong Learning Framework, and 80% No. RS-2024-00457882, AI Research Hub Project, 10%), and Samsung Electronics Co., Ltd. (No. G01240136, KAIST Semiconductor Research Fund (2nd), 10%)

References

- [Askey and Wilson, 1985] Richard Askey and James Arthur Wilson. *Some basic hypergeometric orthogonal polynomials that generalize Jacobi polynomials*, volume 319. American Mathematical Soc., 1985.
- [Bagnall et al., 2018] Anthony Bagnall, Hoang Anh Dau, Jason Lines, Michael Flynn, James Large, Aaron Bostrom, Paul Southam, and Eamonn Keogh. The ueda multivariate time series classification archive, 2018. *arXiv preprint arXiv:1811.00075*, 2018.
- [Beltagy et al., 2020] Iz Beltagy, Matthew E Peters, and Arman Cohan. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020.
- [Chen et al., 2023] Yingyi Chen, Qinghua Tao, Francesco Tonin, and Johan AK Suykens. Primal-attention: Self-attention through asymmetric kernel svd in primal representation. *arXiv preprint arXiv:2305.19798*, 2023.
- [Chien et al., 2021] Eli Chien, Jianhao Peng, Pan Li, and Olga Milenkovic. Adaptive universal generalized PageRank graph neural network. In *ICLR*, 2021.
- [Choi et al., 2024] Jeongwhan Choi, Hyowon Wi, Jayoung Kim, Yehjin Shin, Kookjin Lee, Nathaniel Trask, and Noseong Park. Graph convolutions enrich the self-attention in transformers! *NeurIPS*, 2024.
- [Choromanski et al., 2020] Krzysztof Choromanski, Valerii Likhoshesterov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Lukasz Kaiser, et al. Rethinking attention with performers. *arXiv preprint arXiv:2009.14794*, 2020.
- [Defferrard et al., 2016] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *NeurIPS*, 2016.
- [Deng et al., 2009] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, 2009.
- [Devlin et al., 2019] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*, 2019.
- [Gu and Dao, 2023] Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*, 2023.
- [Gu et al., 2021] Albert Gu, Karan Goel, and Christopher Ré. Efficiently modeling long sequences with structured state spaces. *arXiv preprint arXiv:2111.00396*, 2021.
- [Katharopoulos et al., 2020] Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are rnns: Fast autoregressive transformers with linear attention. In *ICML*. PMLR, 2020.
- [Kim et al., 2024] Jayoung Kim, Yehjin Shin, Jeongwhan Choi, Hyowon Wi, and Noseong Park. Polynomial-based self-attention for table representation learning. In *ICML*, 2024.
- [Kipf and Welling, 2017] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *ICLR*, 2017.
- [Kitaev et al., 2020] Nikita Kitaev, Łukasz Kaiser, and Anselm Levskaya. Reformer: The efficient transformer. *arXiv preprint arXiv:2001.04451*, 2020.
- [Li et al., 2019] Shiyang Li, Xiaoyong Jin, Yao Xuan, Xiyu Zhou, Wenhui Chen, Yu-Xiang Wang, and Xifeng Yan. Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting. *NeurIPS*, 2019.
- [Liu et al., 2021] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, 2021.
- [Maas et al., 2011] Andrew Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. Learning word vectors for sentiment analysis. In *Proceedings of the annual meeting of the association for computational linguistics: Human language technologies*, 2011.
- [Marques et al., 2020] Antonio G Marques, Santiago Segarra, and Gonzalo Mateos. Signal processing on directed graphs: The role of edge directionality when processing and learning from network data. *IEEE Signal Processing Magazine*, 37(6), 2020.
- [Maskey et al., 2023] Sohir Maskey, Raffaele Paolino, Aras Bacho, and Gitta Kutyniok. A fractional graph laplacian approach to oversmoothing. In *NeurIPS*, 2023.
- [McCarthy et al., 1993] PC McCarthy, JE Sayre, and BLR Shawyer. Generalized legendre polynomials. *Journal of mathematical analysis and applications*, 177(2), 1993.
- [Nangia and Bowman, 2018] Nikita Nangia and Samuel R Bowman. Listops: A diagnostic dataset for latent tree learning. *arXiv preprint arXiv:1804.06028*, 2018.
- [Oono and Suzuki, 2020] Kenta Oono and Taiji Suzuki. Graph neural networks exponentially lose expressive power for node classification. In *ICLR*, 2020.
- [Ortega et al., 2018] Antonio Ortega, Pascal Frossard, Jelena Kovačević, José MF Moura, and Pierre Vandergheynst. Graph signal processing: Overview, challenges, and applications. *IEEE*, 106(5), 2018.
- [Qin et al., 2022] Zhen Qin, Weixuan Sun, Hui Deng, Dongxu Li, Yunshen Wei, Baohong Lv, Junjie Yan, Lingpeng Kong, and Yiran Zhong. cosformer: Rethinking softmax in attention. *arXiv preprint arXiv:2202.08791*, 2022.

- [Radford *et al.*, 2019] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.
- [Rusch *et al.*, 2023] T. Konstantin Rusch, Michael M. Bronstein, and Siddhartha Mishra. A survey on oversmoothing in graph neural networks. *arXiv preprint arXiv: 2303.10993*, 2023.
- [Russakovsky *et al.*, 2015] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *IJCV*, 115, 2015.
- [Shen *et al.*, 2021] Zhuoran Shen, Mingyuan Zhang, Haiyu Zhao, Shuai Yi, and Hongsheng Li. Efficient attention: Attention with linear complexities. In *ICCV*, 2021.
- [Shi *et al.*, 2022] Han Shi, Jiahui Gao, Hang Xu, Xiaodan Liang, Zhenguo Li, Lingpeng Kong, Stephen Lee, and James T Kwok. Revisiting over-smoothing in bert from the perspective of graph. In *ICLR*, 2022.
- [Shin *et al.*, 2024] Yehjin Shin, Jeongwhan Choi, Hyowon Wi, and Noseong Park. An attentive inductive bias for sequential recommendation beyond the self-attention. In *AAAI*, 2024.
- [Tay *et al.*, 2020] Yi Tay, Mostafa Dehghani, Samira Abnar, Yikang Shen, Dara Bahri, Philip Pham, Jinfeng Rao, Liu Yang, Sebastian Ruder, and Donald Metzler. Long range arena: A benchmark for efficient transformers. *arXiv preprint arXiv:2011.04006*, 2020.
- [Touvron *et al.*, 2021] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *ICML*, 2021.
- [Vaswani *et al.*, 2017] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, volume 30, 2017.
- [Veličković *et al.*, 2018] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph Attention Networks. In *ICLR*, 2018.
- [Wang *et al.*, 2020] Sinong Wang, Belinda Z Li, Madian Khabsa, Han Fang, and Hao Ma. Linformer: Self-attention with linear complexity. *arXiv preprint arXiv:2006.04768*, 2020.
- [Wang *et al.*, 2022] Peihao Wang, Wenqing Zheng, Tianlong Chen, and Zhangyang Wang. Anti-oversmoothing in deep vision transformers via the fourier domain analysis: From theory to practice. In *ICLR*, 2022.
- [Wi *et al.*, 2025] Hyowon Wi, Jeongwhan Choi, and Noseong Park. Learning advanced self-attention for linear transformers in the singular value domain. *arXiv preprint arXiv:2505.08516*, 2025.
- [Wu *et al.*, 2021] Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. *NeurIPS*, 34, 2021.
- [Wu *et al.*, 2022] Haixu Wu, Jialong Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Flowformer: Linearizing transformers with conservation flows. In *ECCV*, 2022.
- [Wu *et al.*, 2023] Yifan Wu, Shichao Kan, Min Zeng, and Min Li. Singularformer: Learning to decompose self-attention to linearize the complexity of transformer. In *IJCAI*, 2023.
- [Xiong *et al.*, 2021] Yunyang Xiong, Zhanpeng Zeng, Rudrasis Chakraborty, Mingxing Tan, Glenn Fung, Yin Li, and Vikas Singh. Nyströmformer: A nyström-based algorithm for approximating self-attention. In *AAAI*, 2021.
- [Yu *et al.*, 2024] Youn-Yeol Yu, Jeongwhan Choi, Woojin Cho, Kookjin Lee, Nayong Kim, Kiseok Chang, ChangSeung Woo, Ilho Kim, SeokWoo Lee, Joon Young Yang, Sooyoung Yoon, and Noseong Park. Learning flexible body collision dynamics with hierarchical contact mesh transformer. In *ICLR*, 2024.
- [Zaheer *et al.*, 2020] Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. Big bird: Transformers for longer sequences. *NeurIPS*, 33, 2020.
- [Zeng *et al.*, 2021] Zhanpeng Zeng, Yunyang Xiong, Sathya Ravi, Shailesh Acharya, Glenn M Fung, and Vikas Singh. You only sample (almost) once: Linear cost self-attention via bernoulli sampling. In *ICML*. PMLR, 2021.
- [Zerveas *et al.*, 2021] George Zerveas, Srideepika Jayaraman, Dhaval Patel, Anuradha Bhamidipaty, and Carsten Eickhoff. A transformer-based framework for multivariate time series representation learning. In *SIGKDD*, 2021.
- [Zhou *et al.*, 2021] Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *AAAI*, volume 35, 2021.