WavSpA: Wavelet Space Attention for Boosting Transformers' Long Sequence Learning Ability

Yufan Zhuang UC San Diego
Zihan Wang UC San Diego
Fangbo Tao Mindverse
Jingbo Shang UC San Diego

Editors: Marco Fumero, Emanuele Rodolà, Clementine Domine, Francesco Locatello, Gintare Karolina Dziugaite, Mathilde Caron

Abstract

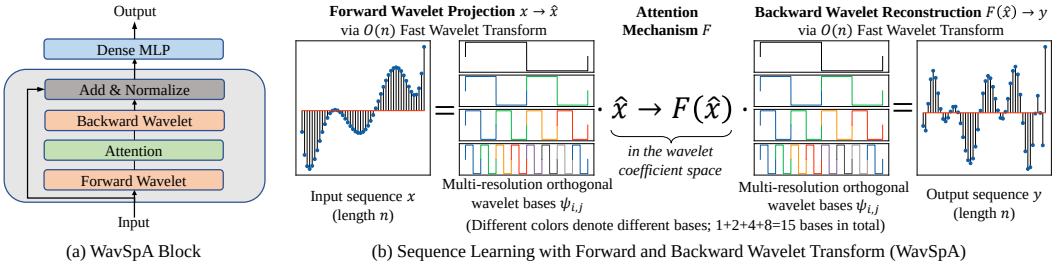
Transformer and its variants are fundamental neural architectures in deep learning. Recent works show that learning attention in the Fourier space can improve the long sequence learning capability of Transformers. We argue that wavelet transform shall be a better choice because it captures both position and frequency information with linear time complexity. Therefore, in this paper, we systematically study the synergy between wavelet transform and Transformers. We propose Wavelet Space Attention (WavSpA) that facilitates attention learning in a learnable wavelet coefficient space which replaces the attention in Transformers by (1) applying forward wavelet transform to project the input sequences to multi-resolution bases, (2) conducting attention learning in the wavelet coefficient space, and (3) reconstructing the representation in input space via backward wavelet transform. Extensive experiments on the Long Range Arena demonstrate that learning attention in the wavelet space using either fixed or adaptive wavelets can consistently improve Transformer's performance and also significantly outperform learning in Fourier space. We further show our method can enhance Transformer's reasoning extrapolation capability over distance on the LEGO chain-of-reasoning task.

1 Introduction

Transformer [39] has become one of the most influential neural architectures in deep learning. Large language models such as ChatGPT [26] have reshaped people's imagination of what an AI model can do in making conversation with humans, solving nontrivial math problems, writing code, and even co-authoring a paper [16]. In image processing, vision transformers have become the backbone for a wide array of applications [9, 29]. Similarly, on source code understanding, Codex [3] can finish people's code given the helper text of the function or just the function name. All of those accomplishments are built upon the foundational Transformer.

Nevertheless, the effective handling of long sequences remains a challenge for Transformers due to the intricate relationships that can exist within such sequences. To address this limitation, recent research has focused on enhancing the Transformers' long-range capabilities through attention learning in transformed sequence spaces. One approach involves low-cost token-mixing, which utilizes forward Fourier transformation to achieve notable accuracy improvements while maintaining quasi-linear time complexity [18]. However, without incorporating a backward transformation, the model might inadvertently mix information from both the input and transformed spaces. To overcome this limitation, researchers have leveraged the forward and backward Fourier transformations to learn large filters with linear weights [31] and non-linearities [11] for vision tasks, exploiting the equivalence between multiplication in the Fourier space and direct convolution in the input space.

Proceedings of the I edition of the Workshop on Unifying Representations in Neural Models (UniReps 2023).



(a) WavSpA Block (b) Sequence Learning with Forward and Backward Wavelet Transform (WavSpA)

Figure 1: An overview of our proposed WavSpA. (a) The only difference between a Transformer block and a WavSpA block is the attention computation. (b) The general flow of computation in WavSpA with learnable forward and backward wavelet transform.

In light of these developments, it is evident that attention learning in transformed sequence spaces holds significant promise for enhancing the effectiveness of Transformers' handling of long-range dependencies. We propose Wavelet Space Attention (WavSpA) that facilitates attention learning in a learnable *wavelet coefficient space*, as shown in Figure 1(a). Specifically, we first apply *forward* wavelet transform to project the input sequence to multi-resolution bases, then conduct attention (e.g., full attention [39], random feature kernel [30]) in the wavelet coefficient space, and finally, reconstruct the representation in input space via *backward* wavelet transform. We implement the transform using Fast Wavelet Transform (FWT) [22] so both transform steps are linear in time, leading to a small overhead.

Performing attention on a sequence in a wavelet-transformed space can offer several advantages. Firstly, it can enhance the representation of the input sequence by capturing relevant features and patterns. By applying the transformation, the sequence is mapped to a new space where certain characteristics might be easier to capture. Attention mechanisms can then be applied in this transformed space to effectively weigh these transformed features, leading to improved representation learning. Secondly, it can enable the attention mechanism to capture different types of relationships between the elements of the sequence, such as associative relationships. By operating in the transformed space, attention can effectively capture the underlying structure of the data and reason over it, leading to improved performance on long sequences. Finally, it is orthogonal to existing work that attempts to replace attention, hence can be combined with any Transformer design.

Besides applying fixed wavelets, we further propose three ways to construct learnable wavelets: direct wavelet parameterization, orthogonal wavelet parameterization, and wavelet lifting. We give detailed explanations of the three schemes and discuss their individual advantages and drawbacks.

We conduct extensive experiments on the Long Range Arena (LRA) benchmark to validate and justify our proposed WavSpA. By combining fixed wavelet space with various representative attention methods, we observed significant performance improvements without introducing additional time complexities. Furthermore, we analyze the performance of WavSpA's three parameterization schemes when coupled with the attention methods, demonstrating even stronger performance boosts. Additionally, our investigation demonstrated that equipping the Transformer with our proposed WavSpA resulted in enhanced reasoning extrapolation capacity, as evidenced by improved performance on the LEGO dataset [47]. These findings highlight the superior long-range understanding capabilities achieved by learning in the wavelet coefficient space compared to the input space or Fourier space.

In summary, our major contributions are as follows.

- We propose WavSpA to facilitate learning in the wavelet space following a forward-backward paradigm which can be paired with various attention methods and boost their long-range understanding capabilities.
- We further propose three adaptive wavelet parameterization schemes (AdaWavSpA, OrthoWavSpA, LiftWavSpA) to maximize the flexibility of wavelet transformation.
- Extensive experiments on the Long-Range Arena benchmark have demonstrated the effectiveness and also justified the design of WavSpA.
- We show WavSpA enhances the reasoning extrapolation capacity to longer sequence lengths.

Reproducibility. Our code is available at `https://github.com/EvanZhuang/wavspa`.

Table 1: Transformed Spaces vs. Original Space (N/A) on the Long Range Arena Text task. We color the number green if it surpasses the baseline (i.e., N/A), red vice versa.

Transformation	Transformer	Linformer	Linear Att.	Longformer	Performer
Original Space (N/A)	64.27	53.94	65.90	62.85	65.40
Fourier - Forward Only [18]	54.65	51.27	65.25	53.51	53.39
Fourier [31, 11]	56.42	**57.06**	71.66	55.36	65.52
Fixed Daubechies-2 Wavelet	**74.82**	55.22	**71.93**	**74.99**	**75.60**

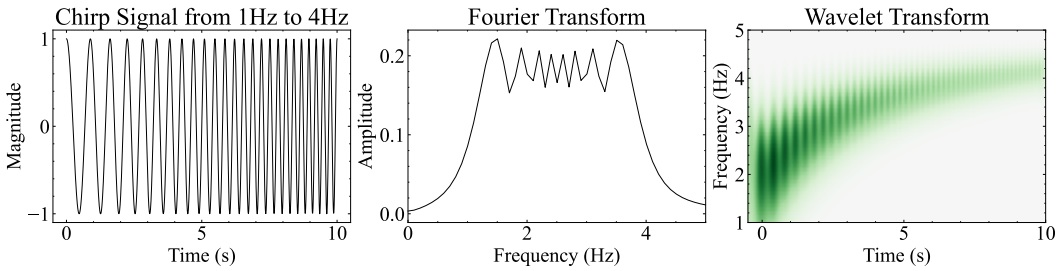


Figure 2: We show a chirp signal from 1Hz to 4Hz, its continuous Fourier transform, and its continuous wavelet transform. From the Fourier transform graph one can only infer the existence of signal in the range of 1-4Hz without time information, while in the wavelet transform graph, both time and frequency information are present and one can tell this is a chirp signal.

2 Learning Attention in a Transformed Space

Inspired by recent work, we begin our study with sequence space transformation with Fourier transforms. FNet [18] replaced the attention with solely forward Fourier transform, it performs well empirically but mixing Fourier coefficients with the input of the original data space is not an intuitive approach. Typical space transforms consist of a forward step and a backward step [31, 11]. Hence, we are interested in comparing sequence learning in a forward-only or in a forward-backward mode.

We conduct pilot studies on the Text task of Long Range Arena [35], combining various attention mechanisms with Forward Only Fourier transform or Forward Backward Fourier transform. The results are summarized in Table 1, and experiment details can be found in Section 4. Notably, we observed that learning with the Forward Backward mode consistently outperformed the Forward Only mode. While the Fourier transform occasionally outperformed the original space, its improvement was not consistently observed across all attention mechanisms.

This phenomenon is understandable since Fourier transform maps signals into the frequency domain, resulting in the loss of time information. In the deep learning context, losing time information is analogous to losing positional information. And positional information is vital in many tasks, as it pins down associative relationships amid elements of the sequence. Hence, preserving and leveraging time information becomes vital for effectively capturing the dependencies within the sequence.

Based on such observation, we propose WavSpA that facilitates attention learning in a wavelet coefficient space, detailed methodology explained in Section 3. Wavelet transform is a sequence projection method where both frequency and time information are captured. As an illustration, we show an example of wavelet transform to demonstrate its ability in time-frequency localization compared to the Fourier transform (see Figure 2). Furthermore, the wavelet transform is multi-level where the decomposition levels correspond to low-to-high frequencies. In the deep learning context, low-frequency signal represents global features and high-frequency signal represents local features, which has been shown useful in prior attention methods [1, 46].

This multi-level decomposition capability corresponds to the multi-level nature of long inputs such as human text. As associative relationships in text occur at various levels, starting from individual words within a sentence. For instance, in the sentence"*The cat chased the mouse*" the words "*cat*" and "*mouse*" are associated in terms of their roles in the action.

Associative relationships also extend beyond sentence boundaries. Texts are organized in hierarchical structures, such as paragraphs, sections, and documents, where higher-level associations emerge.

Within a paragraph, sentences are associated, contributing to a coherent idea. In longer texts like news articles, sections and chapters form hierarchical connections, uniting them under common topics.

This hierarchical structure is not unique to text but also exists in other sequential inputs, including source code, formulas, and more. Recognizing and understanding this multi-level hierarchy is crucial as it enables models to capture rich relationships within the sequence, facilitating more advanced extrapolation reasoning capabilities.

To validate our intuition, we perform experiments on the LRA benchmark (Fixed Daubechies-2 Wavelet row of Table 1), the results indicate wavelet transform can deliver consistent performance boosts across a wide range of attention mechanisms. Furthermore, we present a comprehensive comparison of attention learning in Fourier space and Fixed wavelet spaces in Appendix Table 3.

3 WavSpA: Learning Attention in Parametrized Wavelet Space

In this section, we introduce the details of WavSpA. As shown in Figure 1(a), the only difference between a Transformer block and a WavSpA block is the attention computation. The general flow of WavSpA is shown in Figure 1(b), which constitutes the forward wavelet transform, the attention in the middle, and the backward wavelet transform.

We list our notations here — we denote scalars as x, vectors as $\mathbf{x}$, matrices as X; we denote function f's transformation in the coefficient space as $\hat{f}$.

3.1 WavSpA Paradigm

We propose the WavSpA paradigm to conduct attention learning in the wavelet coefficient space between forward and backward transformation. The forward transformation decomposes the input sequence into coefficients of a set of wavelet basis. We then conduct attention in the coefficient space. In the backward transformation, we reconstruct the target representation in the original function space. For fixed wavelet families, we require the forward-backward transformation pair to be invertible and exact, meaning that one can perfectly reconstruct the same input from the derived coefficients. However, this constraint is not always attached to adaptive wavelets.

The general framework is shown below. In practice, we deal with vectors with dimensions of the attention head dimension. Here, we limit ourselves to 1d functions for a clear illustration. Given input and output function $x(t), y(t) : \mathbb{R} \to \mathbb{R}$ on time domain t, wavelet basis $\psi(\omega, t)$ on both frequency and time domain ω, t (e.g, the basis for a Daubechies-2 wavelet), and attention module $\mathrm{Attention}$,

$$\text{(forward)} \quad \hat{x}(\omega) = \sum_i x(t_i)\psi^*(\omega, t_i) \tag{1}$$

$$\text{(attention)} \quad \hat{h}(\omega) = \mathrm{Attention} \circ \hat{x}(\omega) \tag{2}$$

$$\text{(backward)} \quad y(t) = \sum_j \hat{h}(\omega_j)\psi(\omega_j, t) \tag{3}$$

where $\psi^*(\omega, t)$ denotes the complex conjugate of ψ.

Learning carried out in this space will correspond to gathering and processing information in a coarse to fine-grained fashion. Furthermore, wavelet transform enjoys $O(n)$ time complexity [22], an already desirable property compared to Fourier transform's $O(n \log n)$ complexity.

3.2 Direct Wavelet Parameterization - AdaWavSpA

One key benefit of wavelet transformation is its flexibility in choosing the wavelets for its application, for example, Daubechies wavelets [8] are optimized to have the most compact support; symlets [7] are designed to have better symmetric properties. Therefore it is natural to consider parameterization of the wavelet coefficients and make wavelet transformation part of the learning process.

The direct parameterization scheme is the most intuitive approach. We make the wavelet coefficients learnable parameters, and update them during training. The key problem here is maintaining the structure between the scaling coefficients and the wavelet coefficients, i.e. the quadrature mirror filter

(QMF) relationship [7]. We consider parameterizing the scaling coefficients ($\phi^{(n)} \in \mathbb{R}^n$, n denotes wavelet length) and expanding the system according to the QMF relationship to obtain the full set of wavelet coefficients ($\psi^{(n)} \in \mathbb{R}^n$), shown in equation 4.

$$\psi_j^{(n)} = (-1)^j \phi_{-j}^{(n)}, \quad j \in \mathbb{Z} \tag{4}$$

Further strengthening the learning power of adaptive parameterizations, we use different sets (i.e., d sets) of learnable wavelets for individual hidden dimensions of the input $X \in \mathbb{R}^{n,d}$. At the same time, we do not wish the output to have volatile changes when permuting the hidden dimensions. In other words, we want permutation invariance for the hidden dimensions. For that reason we only use 1d wavelet transform over the input's hidden dimension d for parameterized transformations, including this scheme and the following two parameterization schemes,

The direct parameterization scheme satisfies the QMF relationship automatically, but we have no guarantee that the trained wavelet will form an orthogonal wavelet basis. We enjoy more freedom at the cost of using a potentially imperfect projection and reconstruction pair.

3.3 Orthogonal Wavelet Parameterization - OrthoWavSpA

We provide another way to systematically construct parameterized orthogonal wavelets to keep the perfect reconstruction property intact. There exist extensive studies on this topic [38, 19, 32, 25], but many are based on constrained optimization, which is not ideal for our purpose. We present an unconstrained construction that originates from lattice filters, we refer readers to [25] for details of this design. In general, the orthogonal wavelets are constructed iteratively, each time we extend the wavelet by multiplying the current wavelet by an upshifted rotation matrix. The resulting wavelet basis will always be orthogonal, the formula is shown below:

$$\psi^{(n)} = \mathrm{R}(\theta_1) \cdot \mathrm{U} \cdot \ldots \cdot \mathrm{U} \cdot \mathrm{R}(\theta_n) \tag{5}$$

where R is the rotation matrix and U is an upshift matrix.

As an example, we show how to construct a parameterized wavelet $\psi^{(4)}$ of length 4 from a parameterized wavelet of length 2 ($\psi^{(2)} = [\sin\theta_1, \cos\theta_1]$):

$$\begin{bmatrix} \psi_4^{(4)} \\ \psi_3^{(4)} \\ \psi_2^{(4)} \\ \psi_1^{(4)} \end{bmatrix} = \begin{bmatrix} \cos\theta_2 & 0 \\ -\sin\theta_2 & 0 \\ 0 & \sin\theta_2 \\ 0 & \cos\theta_2 \end{bmatrix} \begin{bmatrix} \sin\theta_1 \\ \cos\theta_1 \end{bmatrix} \tag{6}$$

$\theta_1,\ \theta_2$ represent the two rotation angles that we can set as learnable parameters.

This parameterization scheme offers naturally orthogonal wavelets without the need to customize the loss functions or derive a new optimization process. But on the other hand, this scheme requires more computation than the direct parameterization scheme and the compute cost grows with respect to the wavelet length.

3.4 Wavelet Lifting - LiftWavSpA

The wavelet lifting scheme [34] is developed to become the second-generation wavelet transformation, due to its simplicity and extended flexibility. It is not characterized by transformation via functional convolution, rather it builds its forward and backward transformation from these three steps: 1. *segmentation*: splitting the input into two parts, one widely used segmentation is separating the even and odd parts of the input; 2. *update*: we mix the information from the subsampled segment into the wavelet segment 3. *lifting*: normalizing the subsampled segment and blend the information again.

The simplest design is the so-called Lazy wavelet [34, 33], the forward transformation is shown below for the first level where $(\lambda_{1,:}, \gamma_{1,:})$ represent the subsampled coefficients and wavelet coefficients:

$$\text{(segmentation)} \quad \lambda_{1,k} = x_{2k},\ \forall k \in \mathbb{Z} \tag{7}$$

$$\text{(update)} \quad \gamma_{1,k} = x_{2k+1} - \frac{1}{2}(\lambda_{1,k} + \lambda_{1,k+1}),\ \forall k \in \mathbb{Z} \tag{8}$$

$$\text{(lifting)} \quad \lambda_{1,k} = \lambda_{1,k} + \frac{1}{4}(\gamma_{1,k-1}, \gamma_{1,k}),\ \forall k \in \mathbb{Z} \tag{9}$$

It is assumed that each point in the input is related to its neighbors, hence in equation 8 we mix the information from the even segment to the odd segment. Then to make sure each decomposition level has the same mean and energy, we lift the subsampled coefficients with the wavelet coefficients, mixing the odd segment into the even segment in equation 9. In the Lazy wavelet lifting scheme, the wavelets are inexplicitly parameterized by non-linearities they are later applied to.

A second-level decomposition ($\lambda_{2,:},\ \gamma_{2,:}$) will further decompose the $\lambda_{1,:}$ into finer-grained sequences. And the backward transformation is straightforward: simply reversing the positive and negative signs in the forward steps accordingly will recover the segments.

Wavelet lifting is a simple and straightforward alternative wavelet transformation scheme. The update and lifting step could be subject to arbitrary designs, which entitled this scheme with the most flexibility. However, what comes with this flexibility is the huge search space for finding the optimal lifting, hence we only use the basic Lazy wavelet in our study and leave the rest for future research.

4 Experiments

Our study begins by conducting experiments on the publicly available benchmark Long Range Arena [35]. Our aim is to compare the effectiveness of learning attention in different input spaces: the regular input space, Fourier space with 2D Fourier transformation, and wavelet space with fixed 2D Daubechies-2 wavelet transformation. The results of these experiments demonstrate noteworthy improvements in performance, shown in Table 1.

Furthermore, we proceed to examine the performance of WavSpA's three parameterization schemes when combined with attention methods. The outcomes reveal even more substantial performance gains, as illustrated in Table 2. Based on these findings, we propose a hypothesis that attributes these performance boosts to enhanced reasoning capabilities over distance. To validate our hypothesis, we test it on the LEGO [47] dataset, which is a chain-of-reasoning task.

In addition to our primary investigations, we perform runtime analysis to measure the add-on cost imposed by WavSpA, the result shows that the overhead is small (Appendix A.6). We compare the performance of our adaptive WavSpA when coupled with attention (Transformer-AdaWavSpA) with other efficient transformers on the LRA benchmark in (Appendix Table 7). We provide a proof to show WavSpA maintains Transformer's universal approximation power (Appendix A.3). We also conduct ablation studies to verify the significance of backward reconstruction and the importance of wavelet initialization in the training process (Appendix A.5). In the end, we embark on an exploratory case study to investigate the characteristics of the learned wavelets (Appendix A.7).

4.1 Experimental Design

Long Range Arena (LRA) LRA [35] is designed to compare efficient transformers for their long-range reasoning ability. Since its release which already contains ten different efficient transformers, more and more efficient transformers have chosen it as the primary evaluation target. The datasets require understanding long sequences of mathematical operations, classifying text based on sentiment, matching similar documents, classifying images, and recognizing 2D spacial information. The sequence lengths of the dataset are within the range of 1K-4K.

LEGO LEGO [47] is a reasoning task that encapsulates the problem of following a chain of reasoning. The task itself requires reasoning over a sequence of variables and operators, and figuring out the sign of each variable. A sample input sequence will look like this: $a = +1; b = -a; e = +b; d = -f; c = +d; f = +e;$, and the model will be asked to predict the sign (positive/negative) of each variable. We follow the design of [47], train on the first 14/20 variables, and test on all 20/26 variables for our experiments. We train all models from random initialization.

Wavelet Transformation Details For fixed WavSpA, we use Daubechies-2 wavelet that has length 4 as the default choice and apply 2d wavelet transform with one decomposition level over both the sequence length and hidden dimension. For adaptive WavSpA, we only transform over the sequence length axis because we intend to avoid large permutation variance over the hidden dimensions since we enabled learning distinctive adaptive wavelets over them. In our experiments, for direct wavelet parameterization we initialize from Daubechies-20 wavelet that has length of 40 or Daubechies-8 wavelet that has length of 16, for orthogonal wavelet parameterization we set the wavelet length as

Table 2: Evaluation results of our WavSpA paradigm with the three adaptive parameterization schemes. We denote original space as N/A. Following previous works [20, 43], due to prolonged training on retrieval task, we also report mean test accuracy without this task, denoted as "(w/o r)".

Model	LRA Mean Test Acc				LRA Mean Test Acc (w/o r)			
	N/A	AdaWavSpA	OrthWavSpA	LiftWavSpA	N/A	AdaWavSpA	OrthWavSpA	LiftWavSpA
Transformer	54.39	**70.59**	65.90	59.85	53.62	**68.43**	64.50	60.70
Linformer	49.36	50.72	52.01	**52.12**	48.64	48.12	**49.95**	47.47
Linear Att.	50.67	**64.32**	55.86	56.93	50.06	**62.55**	55.86	57.65
Longformer	53.46	**63.66**	54.96	57.48	52.60	**64.93**	54.96	58.54
Performer	51.41	**65.47**	60.69	56.95	50.81	**64.05**	61.44	58.00

16, for wavelet lifting we conduct three levels of decomposition. The detailed hyper-parameters are reported in Appendix A.4.

Experiment Environment. Our early-stage experiments are conducted on RTX 3090 GPUs and later moved to TPU v2-8s and v3-8s. Our code is written in Jax [2] with the Flax framework [12]. The fixed wavelet transformation implementation is primarily based on Jax Wavelet Toolbox [24] and PyWavelets [17].

4.2 Attention in Fixed WavSpA

Our WavSpA paradigm has a general philosophy of applying attention in the wavelet space and is not limited to a certain type of attention method. We comprehensively evaluate representative attention methods on different space transformations (no transformation, 2d Fourier transformation, and 2d fixed wavelet transformation with Daubechies-2 wavelet). In Table 1, we show that performing full attention, or many other attention approximation operations in a wavelet transformed space as proposed in WavSpA paradigm almost always brings great accuracy improvements. The complete result is shown in Appendix Table 3. Almost all attention methods have increased accuracy when trained in the wavelet space compared to an untransformed space or the Fourier space, except for the Image dataset, where some incur a slight drop in accuracy.

4.3 Attention in Adaptive WavSpA

We demonstrate that the parameterized wavelets can further boost attention methods' performance. In Table 2, we show results for the three parameterization schemes mentioned in Section 3 when each of these schemes is coupled with full attention and several other representative attention methods. The full result is included in Appendix Table 4.

From the experiment results, we observe that direct parameterization almost always provides the highest accuracy elevation, followed by orthogonal parameterization and lifting. This is counter-intuitive: one would think imposing more structures should help the model to learn better wavelets, and in some cases it does, but our experiments show that learning wavelets with the most freedom is the best option most of the time. Does this mean wavelets' nice mathematical properties are not essential at all, and any parameter initialization would work?

We conduct ablation studies where we initialize the directly parameterized wavelets from Gaussian distribution $\mathrm{N}(\mu = 0, \sigma = 0.02)$, and from damped sinusoidal waves ($x[t] = \frac{\cos(t)}{t+1}$). The results are shown in Appendix Table 11. This showcases the importance of initializing from wavelets even when we impose no constraints on them.

4.4 LEGO Reasoning Task

We hypothesize that the observed performance gain comes from the enhanced reasoning power over distance. We test our hypothesis with the LEGO chain-of-reasoning task. Following the original configuration, the Transformer is a randomly initialized BERT-base model (12-layer, 768-hidden dimensions) and AdaWavSpA represents the same model when wrapped in our framework. We train on the first 14/20 variables and evaluate on the last 6 variables. The learning rate (5e-5), training schedule (200 epochs), and batch size (1024) are all following the original configuration. We perform three runs for each model, the result is shown in Figure 3. It can be observed that the Transformer's extrapolation-over-distance capability is significantly enhanced when coupled with our framework.

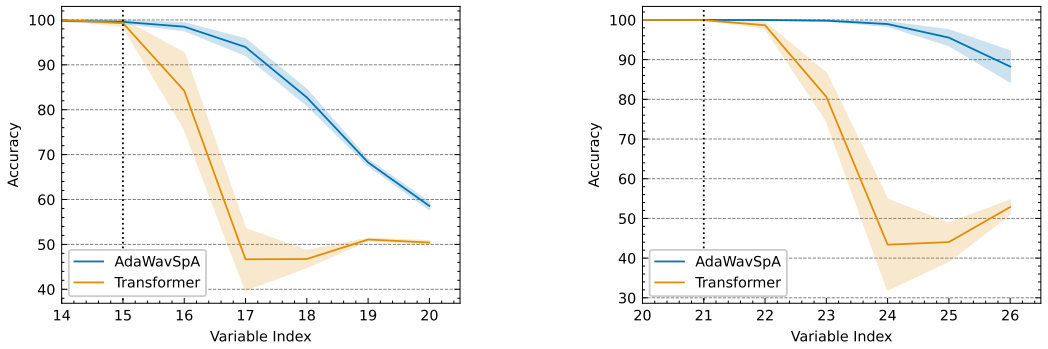


Figure 3: Generalization of Transformer-AdaWavSpA and vanilla Transformer. The WavSpA paradigm improves the reasoning extrapolation to longer sequence lengths. The two figures show the test set results for LEGO with 20 and 26 variables respectively. We report 90% confidence intervals.

5 Related Work

5.1 Attention Methods

There has been plenty of prior work to enable transformers to handle long input more effectively and efficiently. Since the inefficiency comes from the quadratic dependency on sequence length because of the dense attention operation, a large portion of research simulates the attention operation with certain approximations, for example, replacing the dense attention matrix with a sparse version, or assume that it satisfies certain low-rank structures. We briefly review some methods on this topic in this section. For a more detailed survey, we refer the readers to [36].

Sparse Attention. Perhaps the most intuitive solution to alleviate the quadratic cost, Sparse Attention only calculates a portion of the full n^2 attention matrix. Early stage methods include Local Attention [27] and Multi-passage BERT [41] use sliding windows or chunked blocks to speed up computation. Longformer [1] and BigBird [46] further combine global attention, sliding window attention, dilated sliding window attention, and random attention together to form strong sparse attention mechanisms, and BigBird showed that their method is a universal approximator of sequence functions. On the other front, Orthogonal Transformer [13] utilizes an iterative approach to construct an orthogonal vector basis in Euclidean space, then perform windowed attention on grouped tokens after orthogonal projection.

Low-rank Approximation. The self-attention matrix, at the center of transformer, has been found to display low-rank behaviors after pre-training. Linformer [40] performed spectrum analysis on the pre-trained attention matrix, and the results indicate that the top 128 singular values composite 88%-96% of the entire 512 singular values across attention heads and layers. Based on this observation, Linformer added low-rank projection matrices in attention to approximate the original attention matrix. On a similar notion, Drone [4] extended the low-rank approximation scope to all matrices in transformer via data-driven optimal compression.

Kernel Methods. The kernel methods approximate the whole self-attention by replacing the softmax with a kernel function that can be decomposed to avoid the explicit calculation of the $O(n^2)$ matrix multiplication. Linear Transformer [15] proposed a non-negative elu feature mapping as the substitution for the softmax, they further pointed out the connection between their formulation and RNNs, and argued that transformers and RNNs can be unified under the same umbrella. Building on top of this, Random Feature Attention [28] and Performer [5] utilized random feature approximation of the attention, one highlights the importance of normalization before random projection while the other one emphasizes the benefits of positive & orthogonal random features.

Token Mixing. Token Mixing methods are another version of efficient transformer building blocks. Different from the methods discussed above, they do not approximate attention, but rather conduct a new way of enabling communication between tokens. Hard-coded Gaussian attention [44] showed the possibility that a random token mixing strategy can work well in transformer encoders, as opposed to delicate (pre-)trained attention heads. Token Mixing is a new view towards attention learning as

these methods do not perform self-attention at all. FNet [18] pushed this idea further by providing an efficient method to mix the tokens with Fourier forward transformation.

Among these methods, our WavSpA utilizes wavelet transform, thus, is slightly similar to Token Mixing. However, our work should be seen as a new approach to boost transformers, which mixes the idea of a sequence space transform that communicates between tokens and attention methods that can benefit from the new space.

5.2 State Space Models

Different from all the attention methods, state space models (SSM) such as S4 [10] construct long-term memories utilizing orthogonal polynomial projection. They update and maintain the hidden states according to a differential equation, and output the states using linear projection. They have shown outstanding performance on LRA and other long-range tasks. It is also flexible in choosing the family of orthogonal polynomials, but for each polynomial family (Laguerre, Legendre) and each measure (uniform, truncated), significant effort is required to derive the explicit SSM formula. Similarly, MEGA [21] utilized the exponential moving average mechanism to construct its hidden space for recording long-range dependencies and has shown promising results. Our WavSpA is orthogonal towards the SSMs since our target is to boost attention methods' performance on long-range tasks as a sequence space transformation paradigm.

5.3 Sequence Space Transformation in ML

In the field of machine learning, sequence space transformations have gained widespread usage. One particularly common transformation is the Fast Fourier Transform (FFT) [6], which is frequently employed due to its ability to speed up convolution operations. In the context of our research objective, previous works such as AFNO [11] and GFNet [31] have explored the learning of global filters for images by incorporating block-wise MLPs in the Fourier transformation process. This approach can be seen as akin to utilizing a convolutional layer with large filters. However, it is important to note that these methods were primarily designed for learning global filters. Through our comprehensive analysis (Table 3) and ablation study (Table 9), we have demonstrated that such architectures are inadequate for capturing long-range associative relationships.

Fast wavelet transform[22] has been used for neural network compression [42], speech recognition [37], and time series analysis [23]. Recently in computer vision, WaveMix [14] proposed to mix the input images with forward wavelet transform. We note that our work differs from theirs by learning the attention in the coefficient space amid forward and backward wavelet transform.

6 Conclusions and Future Work

In this paper, we propose to learn attention in the wavelet coefficient space. Specifically, the inputs are first forward transformed into the wavelet space, then the attention is learned, and finally, we reconstruct the transformed sequence back in the input space. When coupled with attention methods, learning in wavelet space can boost their performance on long-range understanding tasks while enjoying no extra cost in time complexity. We further propose three ways to learn adaptive wavelets for WavSpA: direct parameterization, orthogonal parameterization, and wavelet lifting. We discuss their advantages and drawbacks and evaluate them empirically. The experiments show adaptive wavelets can provide an even stronger lift to attention methods. In the end, we conduct study on a chain-of-reasoning task, to show the improved long-range learning capability may come from the enhanced reasoning extrapolation power.

Through this work, we have focused on performing attention in the transformed wavelet space, either via fixed wavelet transformation or adaptive wavelet transformation. Is there an optimal way to construct learnable wavelets? And if so what should be the leading criterion for such optimality? These are both interesting questions we leave for the future.

References

[1] Iz Beltagy, Matthew E Peters, and Arman Cohan. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020.

[2] James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018.

[3] Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

[4] Patrick Chen, Hsiang-Fu Yu, Inderjit Dhillon, and Cho-Jui Hsieh. Drone: Data-aware low-rank compression for large nlp models. *Advances in neural information processing systems*, 34:29321–29334, 2021.

[5] Krzysztof Marcin Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Quincy Davis, Afroz Mohiuddin, Lukasz Kaiser, et al. Rethinking attention with performers. In *International Conference on Learning Representations*, 2020.

[6] James W Cooley and John W Tukey. An algorithm for the machine calculation of complex fourier series. *Mathematics of computation*, 19(90):297–301, 1965.

[7] Ingrid Daubechies. Orthonormal bases of compactly supported wavelets. *Communications on pure and applied mathematics*, 41(7):909–996, 1988.

[8] Ingrid Daubechies. *Ten lectures on wavelets*. SIAM, 1992.

[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] Albert Gu, Karan Goel, and Christopher Re. Efficiently modeling long sequences with structured state spaces. In *International Conference on Learning Representations*, 2021.

[11] John Guibas, Morteza Mardani, Zongyi Li, Andrew Tao, Anima Anandkumar, and Bryan Catanzaro. Adaptive fourier neural operators: Efficient token mixers for transformers. *arXiv preprint arXiv:2111.13587*, 2021.

[12] Jonathan Heek, Anselm Levskaya, Avital Oliver, Marvin Ritter, Bertrand Rondepierre, Andreas Steiner, and Marc van Zee. Flax: A neural network library and ecosystem for JAX, 2020.

[13] Huaibo Huang, Xiaoqiang Zhou, and Ran He. Orthogonal transformer: An efficient vision transformer backbone with token orthogonalization. *Advances in Neural Information Processing Systems*, 35:14596–14607, 2022.

[14] Pranav Jeevan and Amit Sethi. Wavemix: Resource-efficient token mixing for images. *arXiv preprint arXiv:2203.03689*, 2022.

[15] Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are rnns: Fast autoregressive transformers with linear attention. In *International Conference on Machine Learning*, pages 5156–5165. PMLR, 2020.

[16] Tiffany H Kung, Morgan Cheatham, Arielle Medinilla, ChatGPT, Czarina Sillos, Lorie De Leon, Camille Elepano, Marie Madriaga, Rimel Aggabao, Giezel Diaz-Candido, et al. Performance of chatgpt on usmle: Potential for ai-assisted medical education using large language models. *medRxiv*, pages 2022–12, 2022.

[17] Gregory R. Lee, Ralf Gommers, Filip Waselewski, Kai Wohlfahrt, and Aaron O'Leary. Pywavelets: A python package for wavelet analysis. *Journal of Open Source Software*, 4(36):1237, 2019.

[18] James Lee-Thorp, Joshua Ainslie, Ilya Eckstein, and Santiago Ontanon. Fnet: Mixing tokens with fourier transforms. *arXiv preprint arXiv:2105.03824*, 2021.

[19] Jean-Marc Lina and Michel Mayrand. Parametrizations for daubechies wavelets. *Physical Review E*, 48(6):R4160, 1993.

[20] Xuezhe Ma, Xiang Kong, Sinong Wang, Chunting Zhou, Jonathan May, Hao Ma, and Luke Zettlemoyer. Luna: Linear unified nested attention. *Advances in Neural Information Processing Systems*, 34:2441–2453, 2021.

[21] Xuezhe Ma, Chunting Zhou, Xiang Kong, Junxian He, Liangke Gui, Graham Neubig, Jonathan May, and Luke Zettlemoyer. Mega: moving average equipped gated attention. *arXiv preprint arXiv:2209.10655*, 2022.

[22] Stephane G Mallat. A theory for multiresolution signal decomposition: the wavelet representation. *IEEE transactions on pattern analysis and machine intelligence*, 11(7):674–693, 1989.

[23] Gabriel Michau, Gaetan Frusque, and Olga Fink. Fully learnable deep wavelet transform for unsupervised monitoring of high-frequency time series. *Proceedings of the National Academy of Sciences*, 119(8):e2106598119, 2022.

[24] Moritz Wolter. *Frequency Domain Methods in Recurrent Neural Networks for Sequential Data Processing*. PhD thesis, Rheinische Friedrich-Wilhelms-Universität Bonn, July 2021.

[25] Nicola Neretti and Nathan Intrator. An adaptive approach to wavelet filters design. In *Proceedings of the 12th IEEE Workshop on Neural Networks for Signal Processing*, pages 317–326. IEEE, 2002.

[26] OpenAI. Chatgpt: Optimizing language models for dialogue, Jan 2023.

[27] Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Lukasz Kaiser, Noam Shazeer, Alexander Ku, and Dustin Tran. Image transformer. In *International conference on machine learning*, pages 4055–4064. PMLR, 2018.

[28] Hao Peng, Nikolaos Pappas, Dani Yogatama, Roy Schwartz, Noah Smith, and Lingpeng Kong. Random feature attention. In *International Conference on Learning Representations*, 2020.

[29] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021.

[30] Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. *Advances in neural information processing systems*, 20, 2007.

[31] Yongming Rao, Wenliang Zhao, Zheng Zhu, Jiwen Lu, and Jie Zhou. Global filter networks for image classification. *Advances in Neural Information Processing Systems*, 34:980–993, 2021.

[32] Peter Rieder, Jurgen Gotze, JS Nossek, and C Sidney Burrus. Parameterization of orthogonal wavelet transforms and their implementation. *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, 45(2):217–226, 1998.

[33] Wim Sweldens. Wavelets and the lifting scheme: A 5 minute tour. *ZAMM-Zeitschrift fur Angewandte Mathematik und Mechanik*, 76(2):41–44, 1996.

[34] Wim Sweldens. The lifting scheme: A construction of second generation wavelets. *SIAM journal on mathematical analysis*, 29(2):511–546, 1998.

[35] 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. In *International Conference on Learning Representations*, 2020.

[36] Yi Tay, Mostafa Dehghani, Dara Bahri, and Donald Metzler. Efficient transformers: A survey. *ACM Computing Surveys (CSUR)*, 2020.

[37] Z Tufekci and John N Gowdy. Feature extraction using discrete wavelet transform for speech recognition. In *Proceedings of the IEEE SoutheastCon 2000.'Preparing for The New Millennium'(Cat. No. 00CH37105)*, pages 116–123. IEEE, 2000.

[38] Palghat P Vaidyanathan and P-Q Hoang. Lattice structures for optimal design and robust implementation of two-channel perfect-reconstruction qmf banks. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 36(1):81–94, 1988.

[39] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[40] Sinong Wang, Belinda Z Li, Madian Khabsa, Han Fang, and Hao Ma. Linformer: Self-attention with linear complexity. *arXiv preprint arXiv:2006.04768*, 2020.

[41] Zhiguo Wang, Patrick Ng, Xiaofei Ma, Ramesh Nallapati, and Bing Xiang. Multi-passage bert: A globally normalized bert model for open-domain question answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5878–5882, 2019.

[42] Moritz Wolter, Shaohui Lin, and Angela Yao. Neural network compression via learnable wavelet transforms. In *International Conference on Artificial Neural Networks*, pages 39–51. Springer, 2020.

[43] 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 *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 14138–14148, 2021.

[44] Weiqiu You, Simeng Sun, and Mohit Iyyer. Hard-coded gaussian attention for neural machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7689–7700, 2020.

[45] Chulhee Yun, Srinadh Bhojanapalli, Ankit Singh Rawat, Sashank Reddi, and Sanjiv Kumar. Are transformers universal approximators of sequence-to-sequence functions? In *International Conference on Learning Representations*, 2019.

[46] 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. *Advances in Neural Information Processing Systems*, 33:17283–17297, 2020.

[47] Yi Zhang, Arturs Backurs, Sébastien Bubeck, Ronen Eldan, Suriya Gunasekar, and Tal Wagner. Unveiling transformers with lego: a synthetic reasoning task. *arXiv preprint arXiv:2206.04301*, 2022.

A Appendix

A.1 Wavelet Transform

Fourier transform decomposes the entire function into global sinusoidal waves. It tells people what *frequencies* are there and in what *magnitude*, but no information is given about *when* that frequency started or ended. See Figure 2 for an illustration on a chirp signal. This limits the capability to understand the local structures of the input and to conduct learning on top of it, which is crucial to many machine learning tasks.

Wavelet transform is designed to solve this issue. We give a basic introduction here, we refer interested readers to [8] for a more thorough explanation. Wavelet transform employs a function $\psi(x), x \in \mathbb{R}$, called mother wavelet, to generate a family of translated and dilated wavelets (see Figure 1(b)):

$$\psi_{i,j}(x) = 2^{\frac{i}{2}}\psi(2^i x - j), \quad i, j \in \mathbb{Z} \tag{10}$$

where scale i controls the resolution of the wavelet and j controls the position of the wavelet. With a larger i the wavelet will be squeezed shorter in space, hence the normalization factor $2^{\frac{i}{2}}$ to ensure the same L^2 norm for all wavelets. The wavelet family $\psi_{i,j}(x)$ is orthogonal on this dyadic grid.

To be a valid mother wavelet $\psi(x)$, the only requirement is admissibility:

$$\int_{\mathbb{R}} \psi(x)\,\mathrm{d}x = 0 \tag{11}$$

In other words, the sum of function value should be 0.

Given any square integrable function $f \in \mathbb{L}^2(\mathbb{R})$ (i.e., $\int |f(x)|^2\,\mathrm{d}x < \infty$) and wavelet functions $\psi_{i,j}$, the wavelet transform pair is defined as:

$$\hat{f}(i,j) = \int_{\mathbb{R}} f(x)\psi^*_{i,j}(x)\,\mathrm{d}x = \sum_t f(x_t)\psi^*_{i,j}(x_t) \tag{12}$$

$$f(x) = \sum_{i=-\infty}^{+\infty}\sum_{j=-\infty}^{+\infty} \hat{f}(i,j)\psi_{i,j}(x) \tag{13}$$

where $\psi^*_{i,j}(x)$ denotes the complex conjugate of $\psi_{i,j}(x)$.

Intuitively, in wavelet transform, we are scanning $f(x)$ with a microscope that has two knobs. One knob is the location j, the other one is the frequency (i.e., 2^i). We will be able to oversee the local structure of the input and calibrate it accordingly with parameterized functions in WavSpA paradigm.

To generalize beyond $\mathbb{L}^2(\mathbb{R})$ and avoid using an infinite number of wavelets, we must introduce another function ϕ, called scaling function with a similar admissibility and orthogonality constraint:

$$\int_{-\infty}^{+\infty} \phi(x)\,\mathrm{d}x = 1, \quad \phi_{i,j}(x) = 2^{\frac{i}{2}}\phi(2^i x - j), \tag{14}$$
$$\text{s.t.} \quad \langle \phi_{i,j}, \psi_{i',j'}\rangle = 0,\ i' > i,\ \forall j, j'$$

$\phi_{i,j}$ is designed to cover the scale up to i, hence the orthogonality requirement. The decomposition of $f(x)$ therefore becomes:

$$f(x) = \sum_{j=-\infty}^{+\infty} \langle \phi_{0,j}, f\rangle \phi_{0,j}(x) + \sum_{i=0}^{+\infty}\sum_{j=-\infty}^{+\infty} \langle \psi_{i,j}, f\rangle \psi_{i,j}(x) \tag{15}$$

Note that although i still goes to $+\infty$ in (15), i usually has an upper limit in practice since it is impossible to work with infinite frequency.

In the d-dimensional case, we do not have a general orthogonal discrete $\mathbb{R}^d$ wavelet, unlike the continuous case. However, we can still perform discrete wavelet transform over each spatial dimension of the input, and we'd still be able to perfectly project and reconstruct the original function. To be more specific, for sequential inputs $X \in \mathbb{R}^{n,d}$ of length n and hidden dimension d, we will apply 2d wavelet transform over both the length and hidden dimension to generate the wavelet coefficients.

A.2 Universal Approximation Power

Background about Attention Let $X \in \mathbb{R}^{n\times d}$ denotes the input sequence of length n and hidden dimension d. A dense self-attention is shown below:

$$\mathrm{Attention}(X) = \mathrm{Softmax}(\frac{QK^\top}{\sqrt{d}})V \tag{16}$$

where $Q = XW_q$, $K = XW_k$, $V = XW_v$ with $W_q, W_k, W_v \in \mathbb{R}^{d\times m}$ stand for the query, key, and value, respectively. The attention head size is denoted by m.

In this subsection, we show that WavSpA can maintain the universal approximation power on seq-to-seq functions for Transformer and its variants. We illustrate this idea with proof for a slightly modified Performer [5] under WavSpA. The goal is to show that for any f in $\mathcal{F}$, $\forall p \in [1, +\infty), \forall \epsilon > 0$, we can find a $\bar{f}$ in the class of Performer-WavSpA, such that:

$$d_p(f, \bar{f}) = \left(\int_{\mathbb{R}^{n\times d}} \|f(\mathbf{X}) - \bar{f}(\mathbf{X})\|_p^p\,\mathrm{d}\mathbf{X}\right)^{\frac{1}{p}} \leq \epsilon$$

We define the Performer-WavSpA class that has positional encoding, h heads, head size s, hidden dimension r as $\mathcal{W}^{h,s,r}$ with FAVOR+ kernel with an additional normalization on the input.

Theorem A.1. *$\forall p \in [1, +\infty),\ \epsilon > 0$, and for any $f \in \mathcal{F}$, we can find a Performer-WavSpA network $\mathsf{w} \in \mathcal{W}^{2,1,4}$, such that $d_p(f, \mathsf{w}) \leq \epsilon$.*

The sketch of the proof is simple: since we have required the transformation pair to be invertible and exact, so for any seq-to-seq function, we can universally approximate it in the wavelet space and it is equivalent to having universal approximation power in the original space. The detailed proof of Theorem A.1 is shown below.

A.3 Proof for Theorem A.1

We define the function class $\mathcal{F}$ to be the set of all countinous functions that map a compact domain in $\mathbb{R}^{n\times d}$ to $\mathbb{R}^{n\times d}$.

We start from making the connection between random feature kernel and regular transformer block:

Lemma A.2. *(Asymptotic Result for FAVOR+) The following is true for independent random w_i,*

$$\begin{aligned}
&\mathrm{MSE}(\hat{\mathrm{SM}}(x, y)) \\
&= \frac{1}{m} \exp\left(\|x + y\|^2\right) \mathrm{SM}^2(x, y)(1 - \exp\left(-\|x + y\|^2\right)) \\
\Rightarrow &\lim_{\mathrm{SM}(x,y)\to 0} \mathrm{MSE}(\hat{\mathrm{SM}}(x, y)) \to 0
\end{aligned}$$

where SM *denotes the softmax kernel,* $\hat{\mathrm{SM}}$ *denotes the random feature kernel, and* MSE *stands for mean-squared error.*

The proof of this lemma can be found at [5, Lemma 2]. It tells us the the MSE error is upper bounded to a constant since x, y is normalized beforehand, and vanishes to 0 as the original softmax kernel value tends to 0 and the number of random features m tends to $+\infty$.

Next we use the main theorem of [45]. We denote the transformer network class that has positional encoding, h heads, head size s, and hidden dimension r as $\mathcal{T}^{h,s,r}$.

Lemma A.3. *$\forall p \in [1, +\infty),\ \epsilon > 0$, and for any $f \in \mathcal{F}$, we can find a Transformer network $g \in \mathcal{T}^{2,1,4}$, such that $d_p(f, g) \leq \epsilon$.*

The proof of Lemma A.3 constitutes of several steps, of which the first step is to approximate any function $f \in \mathcal{F}$ as a piece-wise constant function $\tilde{f}$. Since f is continuous, the piece-wise constant approximation can be of arbitrary accuracy. Next they find a modified transformer $\tilde{g}$ with hardmax operator and a special class of activations. Finally they show that the transformer block g is able to approximate g. The functional distance is then bounded by:

$$d_p(f, g) \leq d_p(f, \tilde{f}) + d_p(\tilde{f}, \tilde{g}) + d_p(\tilde{g}, g) \leq \epsilon$$

We show that with slight modification, the proof will work for Performer-WavSpA, and can be generalized to the WavSpA paradigm under certain constraints.

The proof is outlined below: For $\forall f \in \mathcal{F}$, its wavelet transform $\hat{f}$ (we will also use f_w to denote this, see (12) for details) is still continuous. Hence, the discretization claim remains valid. We can then effectively approximate the self-attention transformer block with the FAVOR+ block up to $\frac{\epsilon}{4}$ difference by controlling the number of random features m. In the end, the backward reconstruction is exact, the distance bound becomes when we control the other three terms to be less than $\frac{1}{4}\epsilon$ as well:

$$\begin{aligned}
&d_p(f, \mathsf{w}) \\
&\leq d_p(f_w, \tilde{f}_w) + d_p(\tilde{f}_w, \tilde{g}) + d_p(\tilde{g}, g) + d_p(g, \mathsf{w}) \\
&\leq \epsilon
\end{aligned}$$

□

A.4 LRA Configuration Details

We tried to follow all hyperparameters as suggested for each of the attention approximations with exceptions on Linformer and Linear Trans. in Image and Pathfinder. For them, we experimented with five additional configurations as shown in Table 5.

Table 3: Performance comparison of Transformers on Long Range Arena: Transformed spaces vs. Original space. We use $\mathcal{F}/\mathcal{W}$ to denote the Fourier space learning similar to AFNO [11] and GFNet [31] or wavelet space (which adds an $O(n \log n)/O(n)$ complexity cost). We color the number green if it surpasses the baseline, red vice versa. Wavelet space ($\mathcal{W}$) demonstrated superior performance in 21 out of 25 architecture/task combinations compared to Fourier space. † We reran Linformer & Linear Attention for all (N/A, $\mathcal{F}, \mathcal{W}$) with the same additional five sets of hyperparameters because of convergence issues.‡ We note that we are unable to reproduce a score close to the original Linformer performance on Pathfinder. § This is the normalized version of Performer as described in Section A.2.

Transformer Variants		ListOps			Text			Retrieval			Image			Pathfinder		
		N/A	$\mathcal{F}$	$\mathcal{W}$	N/A	$\mathcal{F}$	$\mathcal{W}$	N/A	$\mathcal{F}$	$\mathcal{W}$	N/A	$\mathcal{F}$	$\mathcal{W}$	N/A	$\mathcal{F}$	$\mathcal{W}$
Full	$O(n^2)$	36.37	17.80	**37.15**	64.27	56.42	**74.82**	57.46	51.78	**72.43**	42.44	31.41	42.29	71.40	50.55	**78.25**
Linformer	$O(n)$	35.70	36.15	**37.65**	53.94	**57.06**	55.22	52.27	55.93	**65.85**	38.47†	34.89†	**39.17**†	66.44†‡	61.76†	**70.21**†
Linear Att.	$O(n)$	16.13	**37.65**	37.55	65.90	71.66	**71.93**	53.09	**72.71**	70.71	42.32†	**51.07**†	40.83†	75.91†	70.45†	**76.43**†
Longformer	$O(n)$	35.63	18.95	**36.65**	62.85	55.36	**74.99**	56.89	52.52	**66.21**	42.22	29.12	37.10	69.71	50.38	**78.15**
Performer§	$O(n)$	18.01	37.15	**38.20**	65.40	65.52	**75.60**	53.82	60.56	**78.56**	42.77	9.99	**42.98**	77.05	50.49	**79.17**

Table 4: Evaluation results for the three adaptive parameterization schemes, we denote direct/orthogonal parameterization, and wavelet lifting as Ada/Ortho/Lift-WavSpA.

Models	ListOps	Text	Retrieval	Image	Pathfinder	Avg	Avg (w/o r)
Transformer	36.37	64.27	57.46	42.44	71.40	54.39	53.62
AdaWavSpA	**55.40**	81.60	**79.27**	**55.58**	81.12	**70.59**	**68.43**
OrthoWavSpA	45.95	**81.63**	71.52	49.29	81.13	65.90	64.50
LiftWavSpA	42.95	75.63	56.45	42.48	**81.73**	59.85	60.70
Longformer	35.63	62.85	56.89	42.22	69.71	53.46	52.60
AdaWavSpA	**49.30**	**79.73**	58.57	**50.84**	**79.48**	**63.66**	**64.93**
OrthoWavSpA	39.45	78.41	**79.93**	49.93	79.47	54.96	54.96
LiftWavSpA	39.40	78.00	53.27	40.95	75.80	57.48	58.54
Linformer	35.70	53.94	52.27	38.47	66.44	49.36	48.64
AdaWavSpA	37.15	54.75	61.09	34.93	65.66	50.72	48.12
OrthoWavSpA	**38.05**	**56.93**	60.25	**39.45**	65.35	52.01	**49.95**
LiftWavSpA	37.30	54.43	**70.73**	34.66	63.49	**52.12**	47.47
Linear Att.	16.13	65.90	53.09	42.32	75.91	50.67	50.06
AdaWavSpA	38.90	76.82	**71.38**	**54.81**	**79.68**	**64.32**	**62.55**
OrthoWavSpA	**39.55**	**79.45**	69.65	49.93	78.09	55.86	55.86
LiftWavSpA	38.35	73.39	54.06	44.39	74.46	56.93	57.65
Performer	18.01	65.40	53.82	42.77	77.05	51.41	50.81
AdaWavSpA	**46.05**	**80.93**	**71.16**	**52.06**	77.17	**65.47**	**64.05**
OrthoWavSpA	39.80	79.10	57.67	48.78	**78.09**	60.69	61.44
LiftWavSpA	39.85	75.96	52.75	39.97	76.20	56.95	58.00

For all fixed wavelet transform conducted in this work, we use Daubechies-2 [8] as the basis and we set the level of decomposition to 1.

For Performer, the number of random features in the random feature kernel is set as 256 for all text tasks (ListOps, Text, Retrivial), 512 for all image tasks (Image, Pathfinder).

We use the same set of hyperparameters for all the attention methods on individual tasks, the detailed setting is shown in Table 6. We adjust the training length to stabilize the adaptive wavelets, the training will take over min steps and then will terminate with patience ($= 10\% *$ max step). We also find out that on the image task, it is better to use both the first and last output as the readout (CLS2) for parameterized wavelet transformation.

A.5 Ablation Study

We conduct an ablation study for WavSpA, as shown in Table 9. For (Linear), We limit the transformation in wavelet space to be linear. For (Fourier), we use the Fourier transform as the transformation

Table 5: Additional hyperparameter configurations tried for Linformer and Linear Att. in Image and Pathfinder

Hyperparameter	Config_1	Config_2	Config_3	Config_4	Config_5
Layers	1	1	2	2	2
Embedding Dim.	128	128	128	256	256
Attention Dim.	64	64	64	64	64
MLP Dim.	128	128	256	1024	512
Attention Heads	8	8	2	4	4
Dropout	0.2	0.1	0.1	0.1	0.2
Attention Dropout	0.1	0.1	0.1	0.1	0.1

Table 6: Hyperparameter configurations for parameterized WavSpA experiments.

Hyperparameter	ListOps	Text	Retrieval	Image	Pathfinder
Batch Size	400	128	64	64	512
Max Step	80k	50k	50k	200k	500k
Min Step	5k	20k	20k	20k	20k
Layers	8	6	6	8	1
Embedding Dim.	128	256	256	128	128
Attention Dim.	64	256	128	64	64
MLP Dim.	128	1024	256	128	128
Attention Heads	1	1	1	1	8
AdaWavSpA WLen.	40	16	40	40	40
OrthoWavSpA WLen.	16	16	16	16	16
LiftWavSpA Lev.	3	3	3	3	3
Dropout	0.1	0.1	0.1	0.1	0.2
Attention Dropout	0.1	0.1	0.1	0.1	0.1
Readout	CLS	CLS	CLS	CLS2	MEAN

mechanism for WavSpA. For (Forward Fourier), we only use the forward Fourier transform without backward transform. It can be observed that performance dropped significantly in all cases, indicating the necessity of non-linearity in wavelet space and forward-backward wavelet transform.

For fixed WavSpA, we also try out different wavelet families and decomposition level when paired with Performer, results shown in Table 10.

We further test the necessity of wavelets in direct wavelet parameterization scheme. We tried two other initializations on ListOps task when coupled with full attention, one with random Gaussian initialization $\mathrm{N}(\mu = 0, \sigma = 0.02)$, the other one with damped sinusoidal wave initialization. It can be observed from Table 11 that both alternative initializations induced significant performance deterioration.

A.6 Model Runtime Analysis

Since our framework involves two additional steps in attention computation, the forward and backward wavelet transform, it is important to measure the add-on cost of these two transformations. We show the training & inference latency, and the number of additional parameters used in adaptive wavelets in Table 8. The run-time data is collected for 2,000 steps on the Text dataset with sequence length being 4,096. All WavSpA variants pay a small overhead (linear to sequence length) on the wavelet transformations but also gain an advantage in efficiency due to the halved lengths in each decomposition level. For example, with wavelet lifting, we observe a 40% less latency due to the higher decomposition level (L=3) and simpler decomposition scheme. To illustrate the decomposition effect, an input of length 4096 with three decomposition levels will be transformed into four sequences with lengths 2048, 1024, 512, 512, thus delivering a speed-up ($2048^2 + 1024^2 + 512^2 + 512^2 \ll 4096^2$).

Table 7: Evaluation results on Long-Range Arena benchmark. We show both the average accuracy (Avg) and average accuracy without Retrieval (Avg (w/o r)) since LUNA 256, Nyströmformer, and our WavSpA coupled with full attention and direct wavelet parameterization (Transformer-AdaWavSpA) all use prolonged training steps on Retrieval.

Model	ListOps	Text	Retrieval	Image	Pathfinder	Avg	Avg (w/o r)
Transformer	36.37	64.27	57.46	42.44	71.40	54.39	53.62
Local Attention	15.82	52.98	53.39	41.46	66.63	46.06	44.22
Sparse Trans.	17.07	63.58	59.59	44.24	71.71	51.24	49.15
Longformer	35.63	62.85	56.89	42.22	69.71	53.46	52.60
Linformer	35.70	53.94	52.27	38.56	76.34	51.36	51.14
Reformer	37.27	56.10	53.40	38.07	68.50	50.67	49.99
Sinkhorn Trans.	33.67	61.20	53.83	41.23	67.45	51.39	50.89
Synthesizer	36.99	61.68	54.67	41.61	69.45	52.88	52.43
BigBird	36.05	64.02	59.29	40.83	74.87	55.01	53.94
Linear Trans.	16.13	65.90	53.09	42.34	75.30	50.55	49.92
Performer	18.01	65.40	53.82	42.77	77.05	51.41	50.81
FNet	35.33	65.11	59.61	38.67	77.80	55.30	54.23
LUNA 256	37.98	65.78	**79.56**	47.86	78.55	61.95	57.54
Nyströmformer	37.15	65.52	**79.56**	41.58	70.94	58.95	53.80
Transformer-AdaWavSpA	**55.40**	**81.60**	79.27	**55.58**	**81.12**	**70.59**	**68.42**

Table 8: Training, inference latency, and the number of additional wavelet parameters in parametrized wavelet transform for fixed and adaptive WavSpA, where d denotes the hidden dimension.

Schemes	Transformer	Fixed Daubechies-2	AdaWavSpA	OrthoWavSpA	LiftWavSpA
Avg Train Latency	100.00%	102.56%	112.43%	119.97%	63.52%
Avg Inference Latency	100.00%	103.03%	112.16%	119.38%	66.61%
Add. Wavelet Parameters	0	0	Wavelet Length * d	(Wavelet Length / 2) * d	0

Table 9: Ablation study on Long-Range Arena benchmark.

Model	ListOps	Text	Retrieval	Image	Pathfinder	Avg	Avg (w/r)
WavSpA	38.20	75.60	78.56	42.98	79.17	62.90	58.99
Linear	37.70	55.36	55.27	15.75	50.58	42.93	39.84
Fourier	36.85	65.52	60.56	9.99	50.49	44.68	40.71
Forward Fourier	37.15	64.91	65.98	37.84	53.39	51.85	48.32

Table 10: Ablation study for different wavelet families & decomposition levels for fixed Performer-WavSpA on Long-Range Arena benchmark.

Config	ListOps	Text	Retrieval	Image	Pathfinder
Daubechies-2, L=1	38.20	75.60	78.56	42.98	79.17
Daubeuchies-3, L=1	37.85	76.86	73.62	42.30	78.30
Daubeuchies-3, L=2	37.40	76.84	75.43	41.20	50.52
Coiflet-1, L=1	36.85	76.72	75.43	41.42	50.58
Coiflet-1, L=2	37.65	75.97	75.29	43.2	77.49
Symlet-2, L=1	37.50	75.07	75.59	42.85	49.85
Symlet-2, L=2	37.45	75.63	74.51	41.03	77.39
Symlet-2, L=3	37.55	75.39	76.24	40.88	76.84

Table 11: Ablation study for Daubechies wavelet initialization, Gaussian initialization, and damped sinusoidal wave initialization. All experiments are using full attention as the non-linearity.

Initialization	ListOps
Daubechies init.	55.4
Gaussian init.	45.9
Damped Cos init.	44.65

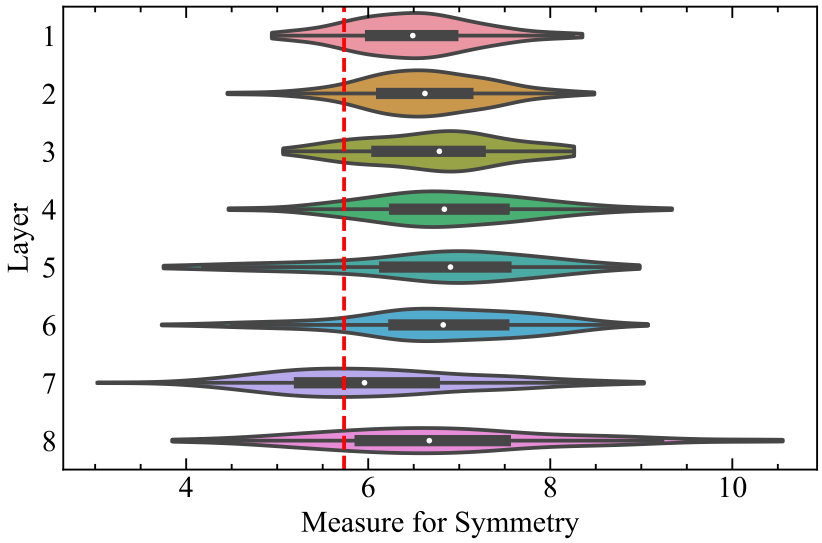


Figure 4: We show the measure for symmetry for all the layers of the Transformer-AdaWavSpA trained on ListOps task. The density plot of each layer shows the distribution for all wavelets of each individual hidden dimension (in this case 128 wavelets). The red dotted vertical line denotes the measure for the Daubechies-20 wavelet, which is the wavelets' initialization value.

A.7 Exploratory Study on Learned Wavelets

Since we have trained the adaptive wavelet in the end-to-end fashion, we are naturally drawn to this question: what kind of wavelets has been learned? We use the best-performing direct parameterization scheme as an example, to empirically examine the learned wavelets.

We initiate our analysis from one commonly studied property of wavelets that characterizes the phase response of the wavelet [32] – symmetry. A closer-to-symmetric wavelet (such as symlet) will have a closer-to-linear phase response, and a less symmetric wavelet (such as Daubechies) will have a more distorted phase response. We measure the symmetry by calculating the ℓ_1 norm between the unit-normalized wavelet and the wavelet's transpose. A perfectly symmetric wavelet will have 0 on this measure.

From Figure 4, we observe that the variance in symmetry grows larger when going from shallow to deep layers. Also on average, the learned wavelets are almost always less symmetric compared to the Daubechies-20 wavelet. These results indicate that it is important to turn wavelet transformation into an adaptive process since the optimal wavelet design varies across the layers and the hidden dimensions.

Table 12: Main results on code understanding tasks, numbers represent accuracy. Code-BERT and C-BERT are BERT-base sized code language models designed for code understanding tasks. We use their published results as a strong baseline, and we are able to surpass the baselines when coupling a much smaller BERT-medium sized model with AdaWavSpA and 32-time less pretraining steps, using the same pretraining corpus. We trained another vanilla BERT-medium following the same procedure as the baseline without WavSpA.

Model	Code-Defection	D2A-Function
Code-BERT/C-BERT	62.08	60.2
BERT-Medium	59.69	59.73
WavSpA-BERT-Medium	**63.47**	**63.75**

Table 13: Sequence length for each dataset's train split. The task of vulnerability detection requires reasoning over the entire piece of code snippet to deduct the label.

Dataset	Code-Defection	D2A-Function
Average Length	1277.49	1038.86
Median Length	561	717

Table 14: Hyperparameter list for code understanding pretraining.

Hyperparameter	BERT-medium	WavSpA-BERT-medium
Layers	8	8
Embedding Dim.	512	512
Attention Dim.	2048	2048
Attention Heads	8	8
Dropout	0.1	0.1
Attention Dropout	0.1	0.1
Batch Size	64	64
Init. Learning Rate	5e-5	5e-5
Warmup Steps	10,000	10,000
Total Steps	100,000	100,000
Adam β_1	0.9	0.9
Adam β_2	0.98	0.98
Weight Decay	0.01	0.01
Wavelet Length	-	16
Wavelet Level	-	1
Wavelet Parametrization	-	Adaptive