Spectral-Window Hybrid (SWH)

Vladimer Khasia
Independent Researcher
vladimer.khasia.1@gmail.com

January 3, 2026

Abstract

Scaling sequence modeling to extreme contexts requires balancing computational efficiency with representational expressivity. While Transformers provide precise retrieval via the attention mechanism, their quadratic $\mathcal{O}(T^2)$ complexity limits their application to long-horizon tasks. In this work, we propose the **Spectral-Window Hybrid (SWH)**, an architecture that decouples sequence modeling into two *parallel* streams: a global branch utilizing the Convolution Theorem to model long-range decay dynamics in $\mathcal{O}(T \log T)$ time, and a local branch employing sliding-window attention for token interactions within a bounded context. By aggregating these representations, SWH avoids the computational bottleneck of global attention while retaining local precision. We demonstrate that SWH matches the perplexity of standard Transformers on short contexts while enabling efficient linear scaling to extended sequences. The code is available at `https://github.com/VladimerKhasia/SWH`

1 Introduction

Sequence modeling constitutes the backbone of modern Artificial Intelligence, driving progress in Large Language Models (LLMs). The Transformer architecture [7] serves as the standard, primarily due to the effectiveness of the self-attention mechanism in modeling dense, pairwise dependencies. However, standard global self-attention incurs a computational cost of $\mathcal{O}(T^2)$ in both time and memory. While hardware-aware optimizations like FlashAttention [1] reduce activation memory complexity to linear and improve computational throughput, the asymptotic quadratic *compute* scaling remains a bottleneck for contexts spanning millions of tokens.

To address this scaling challenge, research has explored *Structured State Space Models (SSMs)*. Early Linear Time Invariant (LTI) systems, such as S4 [4], leverage continuous-time systems that admit a convolutional representation to achieve linear scaling. More recently, Selective SSMs like Mamba [3] introduced input-dependent dynamics to address the limitations of LTI systems in content-dependent reasoning and in-context learning.

Building on these advancements, recent architectures have moved toward *Hybrid* models that combine the efficiency of recurrence with the precision of attention. Jamba [5] utilizes a sequential hybrid approach, interleaving blocks of Mamba layers with occasional global attention layers to balance throughput with long-range retrieval. Similarly, Griffin [2] combines recurrent blocks (specifically the RG-LRU) with local sliding-window attention, avoiding global attention entirely to maintain fixed-size inference states.

In this work, we explore an alternative arrangement to these sequential hybrids. We propose the **Spectral-Window Hybrid (SWH)**, an architecture that utilizes a *parallel decoupled* approach of global convolution and local attention.

2 Methodology

We propose the **Spectral-Window Hybrid (SWH)** architecture, which decomposes sequence modeling into two parallel processes: (1) a global deterministic Linear Time-Invariant (LTI) system solved via the Convolution Theorem, and (2) a local probabilistic attention mechanism with sliding window memory.

Let $\mathbf{X} \in \mathbb{R}^{B \times T \times D}$ denote the input tensor, where B is the batch size, T is the sequence length, and D is the embedding dimension. The output $\mathbf{Y}$ is computed as the projection of the sum of the spectral and local branches[1]:

$$\mathbf{Y} = (\text{RMSNorm}(\mathbf{Y}_{\text{spec}}) + \mathbf{Y}_{\text{local}})\mathbf{W}_{\text{out}} \tag{1}$$

where $\mathbf{W}_{\text{out}} \in \mathbb{R}^{D \times D}$ is a learnable projection and $\mathbf{Y}_{\text{spec}}, \mathbf{Y}_{\text{local}}$ are the outputs of the respective branches defined below.

2.1 Global Branch: Causal Spectral Convolution

We model global dependencies using a parameterized causal convolution. We define a learnable kernel $\mathbf{K} \in \mathbb{R}^{T \times D}$ parameterized as the impulse response of a damped harmonic oscillator. For a time step $t \in [0, T-1]$ and channel $c \in [0, D-1]$:

$$K_{t,c} = \exp(-|\alpha_c| \cdot t) \cos(\omega_c \cdot t) \tag{2}$$

where $\boldsymbol{\alpha}, \boldsymbol{\omega} \in \mathbb{R}^D$ are learnable decay and frequency parameters. The spectral output is obtained via depth-wise causal convolution:

$$\mathbf{Y}_{\text{spec}} = (\mathbf{X}\mathbf{W}_{\text{conv}}) * \mathbf{K} \tag{3}$$

where $\mathbf{W}_{\text{conv}} \in \mathbb{R}^{D \times D}$ is the input projection.

2.1.1 Efficient Implementation via FFT

Direct computation of Eq. (3) requires $\mathcal{O}(T^2)$ operations. We utilize the Convolution Theorem for $\mathcal{O}(T \log T)$ complexity. To enforce causality and prevent circular aliasing inherent to the Discrete Fourier Transform (DFT), the signal and kernel must be zero-padded along the time dimension to length $L \geq 2T$. Computations in the spectral domain are performed in single-precision (FP32) to ensure numerical stability.

Let $\mathcal{F}$ and $\mathcal{F}^{-1}$ denote the FFT and inverse FFT. We compute:

$$\mathbf{Y}_{\text{spec}} = \text{Crop}_{0:T-1}\left(\mathcal{F}^{-1}\left(\mathcal{F}(\text{Pad}(\mathbf{X}\mathbf{W}_{\text{conv}}, 2T)) \odot \mathcal{F}(\text{Pad}(\mathbf{K}, 2T))\right)\right) \tag{4}$$

where $\odot$ denotes the Hadamard product broadcast over the batch dimension, and cropping extracts the first T time steps.

2.2 Local Branch: Chunked Sliding-Window Attention

Local token interactions are recovered via a sliding window attention mechanism of size W. To optimize memory access patterns, we formulate this as **Chunked Attention**.

The input sequence is partitioned into $N = \lceil T/W \rceil$ non-overlapping chunks. If T is not divisible by W, the sequence is right-padded with zeros to ensure strict partitioning. Let $\mathbf{X}^{(n)} \in \mathbb{R}^{B \times W \times D}$ denote the n-th chunk. We compute queries $\mathbf{Q}^{(n)}$, keys $\mathbf{K}^{(n)}$, and values $\mathbf{V}^{(n)}$ via linear projections.

2.2.1 Rotary Position Embeddings

To preserve relative positional information, we apply Rotary Positional Embeddings (RoPE) to the **global** query and key sequences prior to partitioning. Let $f_R(\cdot, m)$ be the rotary injection function for absolute position m. For a token at global index t, the transformed states are:

$$\tilde{\mathbf{q}}_t = f_R(\mathbf{q}_t, t), \quad \tilde{\mathbf{k}}_t = f_R(\mathbf{k}_t, t) \tag{5}$$

These global tensors are subsequently split into chunks $\tilde{\mathbf{Q}}^{(n)}, \tilde{\mathbf{K}}^{(n)}$ for attention computation.

[1]Bias terms are omitted in the mathematical formulation for brevity. In our implementation, bias terms are explicitly disabled for stability in all linear projections, with the exception of the spectral branch input projection.

2.2.2 Context Management and Masking

For the n-th chunk, the key-value context consists of the previous chunk (cached) and the current chunk. This defines a sliding window of size $2W$ (effective receptive field):

$$\mathbf{K}_{\text{ctx}}^{(n)} = [\mathbf{K}^{(n-1)}; \mathbf{K}^{(n)}], \quad \mathbf{V}_{\text{ctx}}^{(n)} = [\mathbf{V}^{(n-1)}; \mathbf{V}^{(n)}] \tag{6}$$

where $[\cdot;\cdot]$ denotes concatenation along the time dimension. For $n = 1$, $\mathbf{K}^{(0)}, \mathbf{V}^{(0)}$ are initialized as zero tensors and logically treated as padding.

The attention logits $\mathbf{S}^{(n)} \in \mathbb{R}^{B \times H \times W \times 2W}$ are computed as:

$$\mathbf{S}^{(n)} = \frac{\tilde{\mathbf{Q}}^{(n)} (\tilde{\mathbf{K}}_{\text{ctx}}^{(n)})^\top}{\sqrt{D/H}} + \mathbf{M}^{(n)} \tag{7}$$

where H is the number of heads and $\mathbf{M}^{(n)}$ is the **Block-Causal Mask**. For a generic chunk $n > 1$, the previous chunk ($j \in [1, W]$) is fully visible, while the current chunk ($j \in [W+1, 2W]$) requires causal masking. For the first chunk ($n = 1$), the previous context is invalid and must be masked out to prevent probability leakage. The mask is defined as:

$$M_{i,j}^{(n)} = \begin{cases} -\infty & \text{if } n = 1 \text{ and } j \leq W \quad \text{(Invalid Context)} \\ 0 & \text{if } n > 1 \text{ and } j \leq W \quad \text{(Valid Context)} \\ 0 & \text{if } W < j \leq W + i \quad \text{(Current Chunk, Causal)} \\ -\infty & \text{otherwise} \end{cases} \tag{8}$$

The final local output is obtained by projecting the concatenated heads:

$$\mathbf{Y}_{\text{local}} = \text{Concat}_{n=1}^{N} \left(\text{Softmax}(\mathbf{S}^{(n)}) \mathbf{V}_{\text{ctx}}^{(n)} \right) \mathbf{W}_{\text{local}} \tag{9}$$

where $\mathbf{W}_{\text{local}} \in \mathbb{R}^{D \times D}$ is the output projection matrix.

2.3 Algorithm

The forward pass implementation is detailed in Algorithm 1.

2.4 Complexity Analysis

We compare the asymptotic complexity of SWH against the standard Transformer (Baseline). Let T be the sequence length and D be the hidden dimension.

As shown in Table 1, the SWH architecture eliminates the quadratic bottleneck $\mathcal{O}(T^2)$. The spectral branch scales log-linearly due to the FFT, while the window attention scales linearly with T given a constant W.

3 Experiments

We evaluate the proposed Spectral-Window Hybrid (SWH) architecture against a standard Transformer baseline. Our evaluation proceeds in two phases: (1) a preliminary synthetic evaluation to verify fundamental sequence modeling capabilities and extrapolation potential, and (2) a large-scale language modeling objective to assess performance on real-world data. Finally, we analyze the computational efficiency regarding latency and memory usage.

All experiments were conducted on Kaggle using two NVIDIA T4 GPUs.

Algorithm 1: Spectral-Window Hybrid (SWH) Forward Pass

Input : $\mathbf{X} \in \mathbb{R}^{B \times T \times D}$, Decay $\boldsymbol{\alpha}$, Freq $\boldsymbol{\omega}$, Window W
Output: $\mathbf{Y} \in \mathbb{R}^{B \times T \times D}$

/* Branch 1: Causal Spectral Convolution */
$\mathbf{U} \leftarrow \mathbf{X}\mathbf{W}_{\text{conv}}$;
$K_{t,c} \leftarrow e^{-|\alpha_c|t} \cos(\omega_c t)$ for $t \in [0, T-1]$;
/* Zero-pad to $2T$ for linear convolution via FFT (FP32) */
$\hat{\mathbf{U}} \leftarrow \texttt{Pad}(\mathbf{U}, 2T)$; $\hat{\mathbf{K}} \leftarrow \texttt{Pad}(\mathbf{K}, 2T)$;
$\mathbf{H}_{\text{spec}} \leftarrow \texttt{irFFT}\big(\texttt{rFFT}(\hat{\mathbf{U}}) \odot \texttt{rFFT}(\hat{\mathbf{K}})\big)$; /* $\odot$ is broadcast over B */
$\mathbf{Y}_{\text{spec}} \leftarrow \mathbf{H}_{\text{spec}}[0 : T-1]$; /* Crop to length T */
/* Branch 2: Chunked Sliding-Window Attention */
$\mathbf{Q}, \mathbf{K}, \mathbf{V} \leftarrow \mathbf{X}\mathbf{W}_Q, \mathbf{X}\mathbf{W}_K, \mathbf{X}\mathbf{W}_V$;
$\mathbf{Q}, \mathbf{K} \leftarrow \texttt{ApplyRoPE}(\mathbf{Q}, \mathbf{K})$; /* Global Position Injection */
/* Handle padding if T is not divisible by W */
$P \leftarrow (W - (T \pmod W)) \pmod W$;
$\mathbf{Q}, \mathbf{K}, \mathbf{V} \leftarrow \texttt{Pad}(\mathbf{Q}, \mathbf{K}, \mathbf{V}, P)$;
Split $\mathbf{Q}, \mathbf{K}, \mathbf{V}$ into N chunks of size W;
Initialize buffers $\mathbf{K}_{\text{prev}}, \mathbf{V}_{\text{prev}} \leftarrow \mathbf{0}^{B \times W \times D}$;
$\mathbf{Y}_{\text{att}} \leftarrow []$;
for $n \leftarrow 1$ **to** N **do**
  $\mathbf{K}_{\text{ctx}} \leftarrow \texttt{Concat}(\mathbf{K}_{\text{prev}}, \mathbf{K}^{(n)})$; $\mathbf{V}_{\text{ctx}} \leftarrow \texttt{Concat}(\mathbf{V}_{\text{prev}}, \mathbf{V}^{(n)})$;
  Calculate Scores $\mathbf{A}^{(n)}$ via Eq. (7);
  Apply Block-Causal Mask $\mathbf{M}^{(n)}$ defined in Eq. (8);
  if $n = 1$ **then**
    Mask region $[0 : W]$ with $-\infty$; /* Ignore padding */
  end
  $\mathbf{O}^{(n)} \leftarrow \texttt{Softmax}(\mathbf{A}^{(n)})\mathbf{V}_{\text{ctx}}$;
  Append $\mathbf{O}^{(n)}$ to $\mathbf{Y}_{\text{att}}$;
  $\mathbf{K}_{\text{prev}} \leftarrow \mathbf{K}^{(n)}$; $\mathbf{V}_{\text{prev}} \leftarrow \mathbf{V}^{(n)}$;
end
$\mathbf{Y}_{\text{local}} \leftarrow \texttt{Concat}(\mathbf{Y}_{\text{att}})$;
if $P > 0$ **then**
  $\mathbf{Y}_{\text{local}} \leftarrow \texttt{Crop}(\mathbf{Y}_{\text{local}}, 0, T-1)$; /* Remove padding */
end
$\mathbf{Y}_{\text{local}} \leftarrow \mathbf{Y}_{\text{local}}\mathbf{W}_{\text{local}}$; /* Output Projection */
/* Fusion */
$\mathbf{Y} \leftarrow (\texttt{RMSNorm}(\mathbf{Y}_{\text{spec}}) + \mathbf{Y}_{\text{local}})\mathbf{W}_{\text{out}}$;
return $\mathbf{Y}$

Method	Time Complexity	Space Complexity
Standard Attention	$\mathcal{O}(T^2 D)$	$\mathcal{O}(T^2 + TD)$
Spectral Branch	$\mathcal{O}(T \log T \cdot D)$	$\mathcal{O}(TD)$
Window Attention	$\mathcal{O}(T \cdot W \cdot D)$	$\mathcal{O}(T \cdot W \cdot H)$
SWH (Total)	$\mathcal{O}(\mathbf{T}(\log \mathbf{T} + \mathbf{W})\mathbf{D})$	$\mathcal{O}(\mathbf{TD})$

Table 1: Complexity comparison. W is fixed window size ($W \ll T$).

3.1 Preliminary Synthetic Evaluation

Before scaling to natural language, we assessed the method's ability to capture essential algorithmic primitives. We compared a basic SWH method against a standard causal Attention (both with $D = 128$, $L = 2$, $H = 4$) on five diagnostic tasks. The models were trained for 3,000 steps. The tasks included:

- **Associative Recall:** Retrieving a value associated with a specific key.
- **Induction Head:** A core mechanism for in-context learning.
- **Sorting:** Outputting the sorted version of an integer sequence.
- **Length Generalization (LenGen):** Trained on sequence length $T = 32$, evaluated on $T = 128$.
- **Needle-in-a-Haystack:** Retrieving a specific token pair buried in a long sequence (evaluated at $T = 256$ after training on $T = 32$).

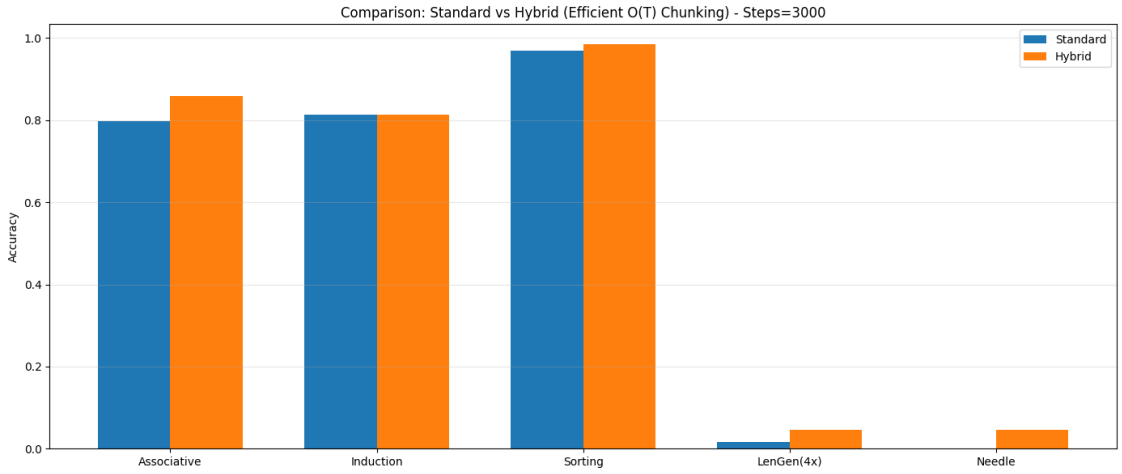

Figure 1: **Synthetic Task Performance.** Comparison of Standard Attention vs. SWH on five algorithmic primitives. SWH demonstrates superior generalization on out-of-distribution lengths (LenGen, Needle) while maintaining strong performance on in-distribution tasks.

The results (Table 2 and Figure 1) indicate that SWH matches the baseline on in-distribution tasks (Associative, Induction, Sorting). Notably, SWH exhibits superior capability in length extrapolation. While the baseline fails completely on the Needle task (0.00 accuracy) when the sequence length increases beyond training limits, SWH maintains non-zero performance (0.05), attributed to the continuous nature of the spectral decay and the localized sliding window.

Table 2: Accuracy on Synthetic Primitives (Training Steps=3000). SWH matches the baseline on standard tasks and shows superior capabilities in length generalization.

Method	Associative	Induction	Sorting	LenGen (4x)	Needle
Standard Transformer	0.80	0.81	0.97	0.02	0.00
SWH (Ours)	**0.86**	**0.81**	**0.98**	**0.05**	**0.05**

3.2 Language Modeling on FineWeb-Edu

We trained a 125M parameter class model on the `FineWeb-Edu` dataset (HuggingFaceFW/fineweb-edu sample-10BT [6]). Both the Baseline and SWH configurations utilized $L = 12$ layers, $D = 768$, and $H = 12$

heads. Training was conducted with a block size of 1024, a batch size of 64 (accumulated), and a learning rate of 6×10^{-4} with cosine decay.

Figure 2 illustrates the validation loss over 16,000 steps. The SWH architecture achieves a consistently lower perplexity compared to the Transformer baseline. The hybrid nature of the architecture allows it to capture both long-range global contexts (via the spectral branch) and sharp local dependencies (via the window attention), resulting in improved convergence efficiency.

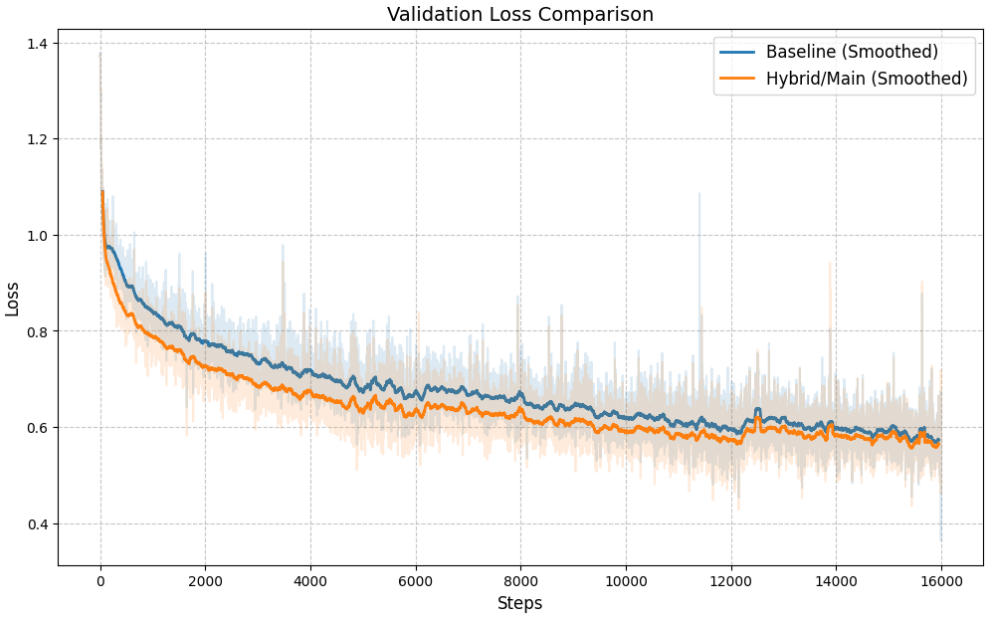

Figure 2: **Validation Loss on FineWeb-Edu.** The SWH model (Orange) consistently achieves lower loss compared to the Standard Transformer Baseline (Blue). Smoothed curves (solid lines) are overlaid on raw loss values (transparent).

3.3 Computational Efficiency

A primary motivation for SWH is breaking the quadratic bottleneck of standard attention. We benchmarked the inference latency and VRAM usage across sequence lengths $T \in [512, 1024, 2048, 4096]$.

As shown in Figure 3, the Standard Transformer exhibits quadratic scaling $\mathcal{O}(T^2)$, resulting in a latency of > 1.4s and memory saturation at $T = 4096$. In contrast, SWH scales linearly. At $T = 4096$, SWH reduces latency by approximately 60% (0.6s) and maintains constant memory usage due to the fixed-size window attention and the $\mathcal{O}(T)$ memory cost of the spectral convolution.

4 Conclusion

In this paper, we introduced the Spectral-Window Hybrid (SWH), a sequence modeling architecture designed to reconcile the efficiency of global convolutions with the precision of local attention. By decomposing the modeling task into parallel spectral and windowed-spatial components, SWH eliminates the $\mathcal{O}(T^2)$ complexity of standard Transformers, achieving near-linear scaling via FFT-based convolutions.

This architecture offers a scalable pathway for long-context modeling, providing the inference speed of linear models with the local representational power of Transformers.

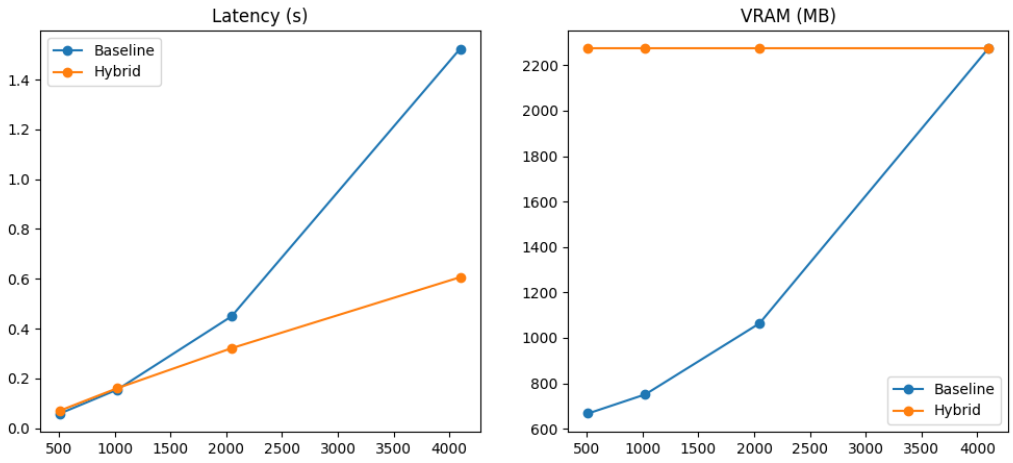

Figure 3: **Efficiency Analysis.** Left: Inference latency (seconds) vs. sequence length. Right: VRAM usage (MB) vs. sequence length. SWH displays linear scaling, whereas the baseline exhibits quadratic growth, making SWH significantly more efficient for long sequences.

A Experimental Details

A.1 Hyperparameters

Table 3 details the hyperparameters used for the main Language Modeling experiment described in Section 3.2.

Table 3: Hyperparameters for FineWeb-Edu Training.

Parameter	**Value**
Parameter Count	≈ 125M
Layers (L)	12
Hidden Dimension (D)	768
Heads (H)	12
Block Size (T)	1024
Vocab Size	50304
Optimizer	AdamW
Learning Rate	6×10^{-4}
Weight Decay	0.1
Batch Size (Global)	64
Gradient Accumulation	8 steps
Precision	Mixed (BF16/FP16)
FFT Precision	FP32

A.2 Synthetic Task Definitions

The synthetic tasks used in Section 3.1 are defined as follows:

- **Associative Recall:** Given a sequence of key-value pairs (e.g., $k_1, v_1, k_2, v_2, \dots$), the model must predict v_i when prompted with k_i at the end of the sequence.
- **Induction Head:** The model must complete a pattern $A \dots B \dots A \to B$. A token A appears, followed eventually by B. When A appears again later, the model must predict B.
- **Sorting:** The input is a sequence of random integers drawn from the vocabulary. The target is the same sequence sorted in ascending order.
- **Length Generalization (LenGen):** The model is trained on standard associative recall with sequence length T_{train}. It is evaluated on sequences of length $T_{\text{test}} = 4 \times T_{\text{train}}$.
- **Needle-in-a-Haystack:** A specific key-value pair ("the needle") is inserted at a random position within a long sequence ("the haystack") of noise tokens. The model is queried for the value at the very end.

References

[1] Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher R'e. Flashattention: Fast and memory-efficient exact attention with io-awareness. *ArXiv*, abs/2205.14135, 2022.

[2] Soham De, Samuel L. Smith, Anushan Fernando, Aleksandar Botev, George Cristian-Muraru, Albert Gu, Ruba Haroun, Leonard Berrada, Yutian Chen, Srivatsan Srinivasan, Guillaume Desjardins, Arnaud Doucet, David Budden, Yee Whye Teh, Razvan Pascanu, Nando de Freitas, and Caglar Gulcehre. Griffin: Mixing gated linear recurrences with local attention for efficient language models. *ArXiv*, abs/2402.19427, 2024.

[3] Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *ArXiv*, abs/2312.00752, 2023.

[4] Albert Gu, Karan Goel, and Christopher R'e. Efficiently modeling long sequences with structured state spaces. *ArXiv*, abs/2111.00396, 2021.

[5] Opher Lieber, Barak Lenz, Hofit Bata, Gal Cohen, Jhonathan Osin, Itay Dalmedigos, Erez Safahi, Shaked Haim Meirom, Yonatan Belinkov, Shai Shalev-Shwartz, Omri Abend, Raz Alon, Tomer Asida, Amir Bergman, Roman Glozman, Michael Gokhman, Avshalom Manevich, Nir Ratner, Noam Rozen, Erez Shwartz, Mor Zusman, and Yoav Shoham. Jamba: A hybrid transformer-mamba language model. *ArXiv*, abs/2403.19887, 2024.

[6] Anton Lozhkov, Loubna Ben Allal, Leandro von Werra, and Thomas Wolf. Fineweb-edu: the finest collection of educational content, 2024.

[7] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Neural Information Processing Systems*, 2017.