EDUSCHOOL INTERNATIONAL JOURNAL OF DATA SCIENCE AND MACHINE LEARNING (EIJDSML)

(Open Access, Double-Blind Peer Reviewed Journal)

ISSN Online: ISSN Print

Attention-Free Transformers: State Space Models as Scalable Alternatives

Juby George

Assistant Professor, Department of Computer Applications, Marian College Kuttikkanam Autonomous, Kerala, India

Article information

Received: 3rd January 2026
Received in revised form: 5th February 2026
Accepted: 7th March 2026
Available online: 18th April 2026

Volume: 1
Issue: 1
DOI: https://doi.org/10.5281/zenodo.19627334

Abstract

The self-attention mechanism underpinning transformer architectures imposes quadratic computational complexity with respect to sequence length, creating fundamental scalability barriers for long-context applications. State space models (SSMs) have emerged as a compelling alternative, offering linear-time sequence modeling through structured recurrent dynamics derived from continuous-time systems theory. This paper provides a comprehensive examination of the theoretical foundations, architectural evolution, and empirical performance of SSM-based architectures. We trace the progression from foundational structured state spaces (S4) through diagonal parameterizations (S4D, S5) and gated variants (H3) to the selective state space paradigm introduced by Mamba. Our analysis covers the HiPPO initialization framework, discretization strategies, hardware-aware algorithm design, and the emerging structured state space duality connecting SSMs to attention. We present extensive comparisons against transformer baselines across language modeling, long-range sequence classification, audio processing, and genomic sequence analysis. We also examine hybrid architectures that combine SSM and attention layers, discussing their advantages and trade-offs. Open challenges including in-context learning limitations, multimodal extension, and hardware co-design are identified and discussed.

Keywords:- State Space Models, Structured State Spaces, Mamba, Self-Attention, Long-Range Dependencies, Sequence Modeling, Linear Complexity, Selective State Spaces.

I. INTRODUCTION

The transformer architecture, introduced by Vaswani et al. [1], has become the dominant paradigm in deep learning, achieving state-of-the-art results across natural language processing [2], computer vision [3], speech recognition [4], and scientific computing [5]. At the core of the transformer lies the self-attention mechanism, which computes pairwise interactions between all elements in a sequence, enabling the model to capture arbitrary long-range dependencies. However, this global computation imposes $O(N^2)$ time and memory complexity with respect to sequence length N, creating fundamental scalability constraints.

For applications requiring long-context reasoning—such as document-level understanding, genomic sequence analysis, high-resolution image processing, and long-form audio generation—the quadratic scaling of attention becomes a critical bottleneck. A sequence of length 100,000 tokens requires 10 billion pairwise computations per attention layer, rendering standard transformers impractical for many real-world tasks [6].

Numerous approaches have been proposed to address this limitation. Efficient attention variants, including sparse attention [7], linear attention [8], and low-rank approximations [9], reduce the computational cost but often sacrifice modeling quality. An alternative paradigm has emerged from control theory and dynamical systems: state space models (SSMs), which process sequences through a fixed-dimensional latent state, achieving linear-time complexity while maintaining the ability to capture long-range dependencies [10].

This paper provides a comprehensive survey of SSM-based architectures as alternatives to self-attention. We organize our discussion around four themes:

- The theoretical foundations of continuous-time state spaces and their discretization for sequence modeling
- The architectural evolution from s4 to mamba and beyond
- Empirical comparisons with transformer baselines across diverse domains
- Open challenges and future research directions. Table 1 summarizes the key ssm architectures discussed in this paper.

Table 1. Chronological Overview of Key State Space Model Architectures

Model	Year	Key Innovation	Complexity	Domain
S4 [10]	2022	HiPPO initialization + FFT conv.	O(N log N)	General sequences
S4D [11]	2022	Diagonal state matrix	O(N)	General sequences
S5 [12]	2023	MIMO parallel scan	O(N)	General sequences
H3 [13]	2023	SSM + multiplicative gating	O(N log N)	Language modeling
Mamba [14]	2023	Selective (input-dependent) SSM	O(N)	Language + general
Mamba-2 [15]	2024	Structured state space duality	O(N)	Language modeling
Jamba [16]	2024	Hybrid Mamba + attention	O(N)	Language modeling

II. THEORETICAL FOUNDATIONS OF STATE SPACE MODELS

A. Continuous-Time State Space Formulation

State space models originate from modern control theory, representing linear time-invariant (LTI) dynamical systems through first-order ordinary differential equations. The continuous-time state space representation takes the canonical form:

$$x'(t) = Ax(t) + Bu(t) \tag{1}$$

$$y(t) = Cx(t) + Du(t) \tag{2}$$

where $x(t) \in \mathbb{R}^N$ is the hidden state, $u(t) \in \mathbb{R}$ is the input signal, $y(t) \in \mathbb{R}$ is the output, and $A \in \mathbb{R}^{NxN}$, $B \in \mathbb{R}^{Nx1}$, $C \in \mathbb{R}^{1xN}$, $D \in \mathbb{R}$ are learned parameter matrices [17].

The state matrix A governs the dynamics of the hidden state and is the most critical component for sequence modeling. The key insight of Gu et al. [10] is that naive parameterization of A leads to vanishing or exploding gradients over long sequences—a problem analogous to that faced by recurrent neural networks (RNNs) [18]. The solution lies in structured initialization of A based on polynomial projection operators.

B. HiPPO: High-Order Polynomial Projection Operators

The HiPPO (High-order Polynomial Projection Operators) framework [19] provides a principled initialization for the state matrix A by deriving matrices that optimally project the input history onto a basis of orthogonal polynomials. Specifically, the HiPPO-LegS (Legendre scaled) matrix compresses the entire input history into a fixed-dimensional state by maintaining an optimal polynomial approximation of all past inputs under an exponentially decaying measure. The HiPPO-LegS matrix has the specific form:

$$A_{nk} = -\sqrt{2n+1}\sqrt{2k+1} \quad \text{if } n > k, \tag{3}$$

$$A_{nk} = -(n+1) \quad \text{if } n = k, \tag{4}$$

$$A_{nk} = 0 \quad \text{if } n < k. \tag{5}$$

This lower-triangular structure ensures that the state retains a compressed representation of the entire input history with provably bounded approximation error, enabling SSMs to capture dependencies over thousands

of time steps [19]. The HiPPO initialization was shown to be critical for S4's breakthrough performance on the Long Range Arena benchmark, where standard RNN initializations completely failed [10].

C. Discretization and Computational Modes

To process discrete sequences, the continuous-time SSM must be discretized with a step size Δ. The bilinear (Tustin) transform is the standard discretization method, converting the continuous parameters (A, B) to discrete parameters:

$$\bar{A} = \left(I - \frac{\Delta}{2A}\right)^{-1}\left(I + \frac{\Delta}{2A}\right) \tag{6}$$

$$\bar{B} = \left(I - \frac{\Delta}{2A}\right)^{-1}\Delta B \tag{7}$$

The zero-order hold (ZOH) discretization provides an alternative:

$$\bar{A} = \exp(\Delta A) \tag{8}$$

$$\bar{B} = (\Delta A)^{-1}(\exp(\Delta A) - I)\Delta B \text{ [10]} \tag{9}$$

A fundamental advantage of SSMs is their dual computational mode. During training, the recurrence:

$$x_k = \bar{A}x_{k-1} + \bar{B}u_k \tag{10}$$

can be unrolled into a global convolution y = K * u, where the kernel K = ($C\bar{B}$, $C\bar{A}\bar{B}$, $C\bar{A}^2\bar{B}$, ...) is precomputed via fast Fourier transform (FFT) in O(N log N) time. During inference, the model operates as a recurrence with O(1) cost per step and O(N) state size, enabling efficient autoregressive generation without the key-value cache that transformers require [10].

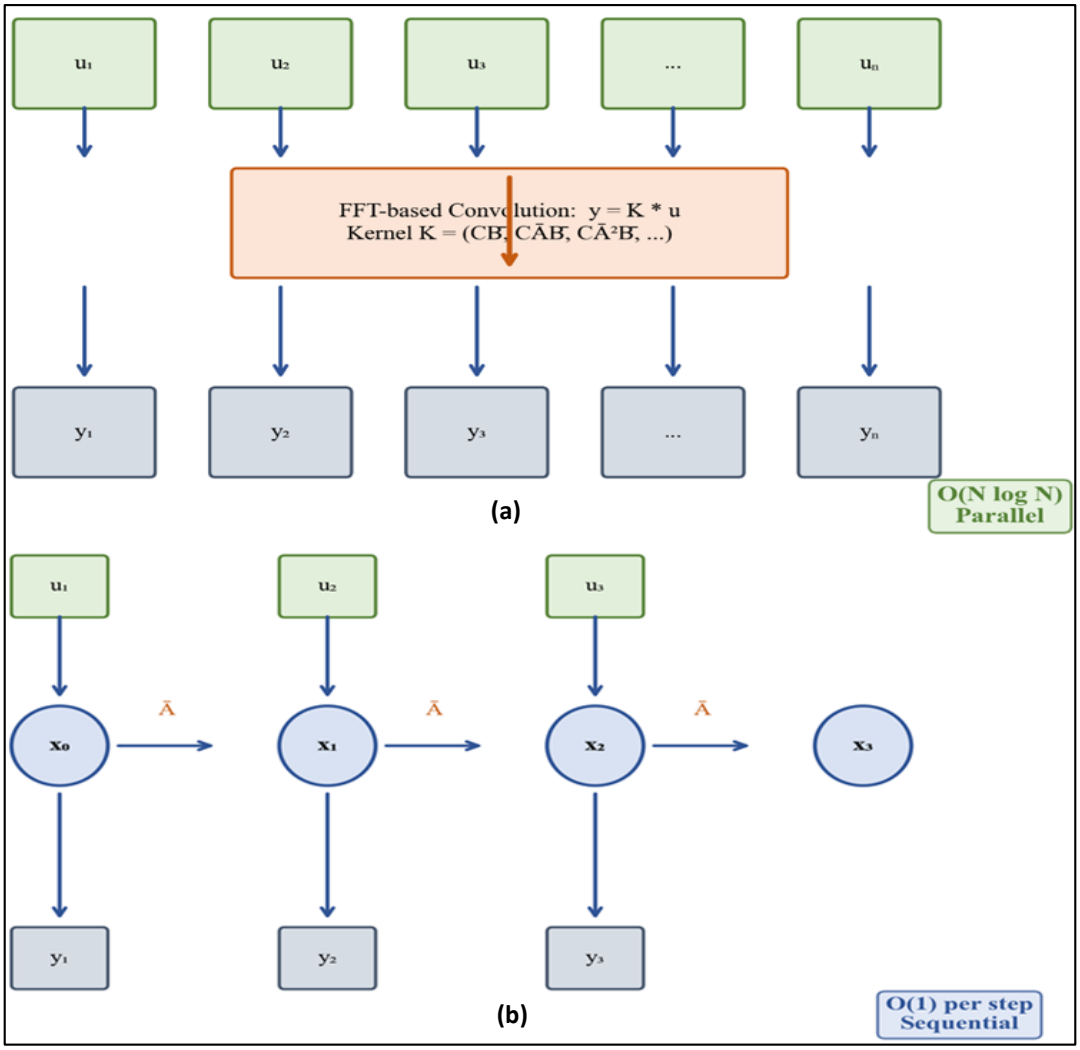

Fig 1: Dual computational modes of SSMs: (a) Convolutional Mode (Training) (b) Recurrent Mode (Inference)

Convolutional mode (parallel, O(N log N)) for training and recurrent mode (sequential, O(1) per step) for inference. Adapted from [10].

III. ARCHITECTURAL EVOLUTION OF STATE SPACE MODELS

A. S4: Structured State Spaces for Sequences

The Structured State Space Sequence model (S4) [10] was the first SSM architecture to achieve competitive performance with transformers on a broad range of sequence modeling tasks. S4's key contributions include:

- The use of hippo initialization for the state matrix A
- A normal plus low-rank (NPLR) parameterization that enables efficient computation of the convolutional kernel via the Cauchy kernel formula
- A deep architecture stacking multiple SSM layers with nonlinear activations between them.

S4 achieved a landmark average score of 86.09% on the Long Range Arena (LRA) benchmark [20], dramatically outperforming all prior efficient transformer variants. Particularly notable was its performance on the Path-X task (sequence length 16,384), where all transformer-based methods had failed to exceed random chance, while S4 achieved 94.20% accuracy. This result demonstrated the fundamental advantage of SSMs for extremely long-range dependency modeling.

B. Simplified Parameterizations: S4D and DSS

While S4's NPLR parameterization was mathematically elegant, it was complex to implement and optimize. Gu et al. [11] showed that restricting the state matrix to a diagonal form (S4D) preserves most of S4's modeling capability while dramatically simplifying the architecture. With a diagonal A, the convolutional kernel computation reduces to independent geometric series, eliminating the need for the Cauchy kernel and reducing the implementation to a few lines of code.The Diagonal State Space (DSS) model [21] independently arrived at a similar simplification, demonstrating that diagonal SSMs with careful initialization achieve comparable performance to full S4 on most LRA tasks. S4D further showed that initializing the diagonal entries as the eigenvalues of the HiPPO matrix (which lie in the left half of the complex plane, ensuring stability) is sufficient for strong performance. These simplifications made SSMs significantly more accessible to the research community.

C. S5: Parallel Scanning with MIMO SSMs

Smith et al. [12] introduced S5, which extends the SSM framework to multi-input, multi-output (MIMO) state spaces with a single shared state matrix across all input channels. Unlike S4, which uses independent single-input single-output (SISO) SSMs for each channel and relies on FFT-based convolution, S5 leverages the parallel scan algorithm [22] to compute the recurrence efficiently on modern hardware.The parallel scan operates on the discretized recurrence $x_k = \bar{A}x_{k-1} + \bar{B}u_k$ by decomposing it into a binary associative operation that can be computed in O(log N) parallel steps using O(N) processors. This approach avoids the FFT entirely and maps naturally to GPU architectures. S5 matched S4's performance on LRA while being simpler to implement and more hardware-friendly [12].

D. H3: Bridging SSMs and Language Modeling

Despite strong performance on synthetic benchmarks, early SSMs underperformed transformers on language modeling tasks. Fu et al. [13] identified two key capabilities where SSMs lagged: associative recall (remembering which token appeared after a given token) and comparison (determining relationships between distant tokens). H3 (Hungry Hungry Hippos) addressed these gaps by combining two SSM layers with a multiplicative gating mechanism inspired by linear attention.Specifically, H3 computes: $y = \text{SSM\_shift}(x) \odot \text{SSM\_diag}(x)$, where SSM_shift performs a position-shifting operation (analogous to the Q-K interaction in attention) and SSM_diag performs a diagonal recurrence (analogous to the value projection). This gated structure enables H3 to match transformer perplexity on OpenWebText at the 125M and 355M parameter scales, closing the gap that had prevented SSM adoption for language modeling [13].

IV. THE MAMBA PARADIGM: SELECTIVE STATE SPACES

A. Limitations of Time-Invariant SSMs

All preceding SSM architectures (S4, S4D, S5, H3) employed linear time-invariant (LTI) dynamics: the state matrices A, B, C are fixed parameters independent of the input. While the LTI property enables efficient convolutional computation, it fundamentally limits the model's ability to perform content-based reasoning. An

LTI system processes the tokens 'the cat sat' and 'the dog sat' with identical dynamics, unable to condition its state transitions on the specific input tokens [14].

Gu and Dao [14] formalized this limitation through the lens of information compression. An effective sequence model must selectively propagate relevant information while filtering irrelevant content. LTI models lack a mechanism for such selection—they treat all inputs uniformly, relying entirely on post-hoc nonlinear mixing between layers to achieve content-dependent processing.

B. The Selection Mechanism

Mamba [14] introduces input-dependent (selective) state space dynamics by making the parameters B, C, and the discretization step Δ functions of the input: B(x) = Linear_B(x), C(x) = Linear_C(x), and Δ(x) = softplus(Linear_Δ(x)). This seemingly simple modification has profound consequences: the model can now gate information flow based on input content, selectively remembering or forgetting information at each time step.

The selection mechanism enables Mamba to solve tasks that are provably impossible for LTI models, such as the selective copying task (copying only specific tokens from an input sequence based on a selection criterion) and the induction head task (completing the pattern 'A B ... A → B' by recognizing and reproducing in-context patterns). These capabilities are fundamental building blocks for in-context learning [14].

C. Hardware-Aware Algorithm Design

Making SSM parameters input-dependent breaks the convolutional mode—the kernel K can no longer be precomputed since it changes for each input. Naively, this would force sequential computation in O(N) time. Mamba overcomes this through a hardware-aware parallel scan algorithm inspired by FlashAttention [23].

The key insight is that the bottleneck is not arithmetic operations but memory bandwidth between GPU high-bandwidth memory (HBM) and on-chip SRAM. Mamba's algorithm:

- Loads the SSM parameters from HBM to SRAM
- Computes the discretization and parallel scan entirely in SRAM
- Writes only the final outputs back to HBM. This kernel fusion eliminates intermediate memory reads/writes, achieving up to 3× faster wall-clock speed than equivalent-quality transformers at inference for sequences of length 2K–64K [14].

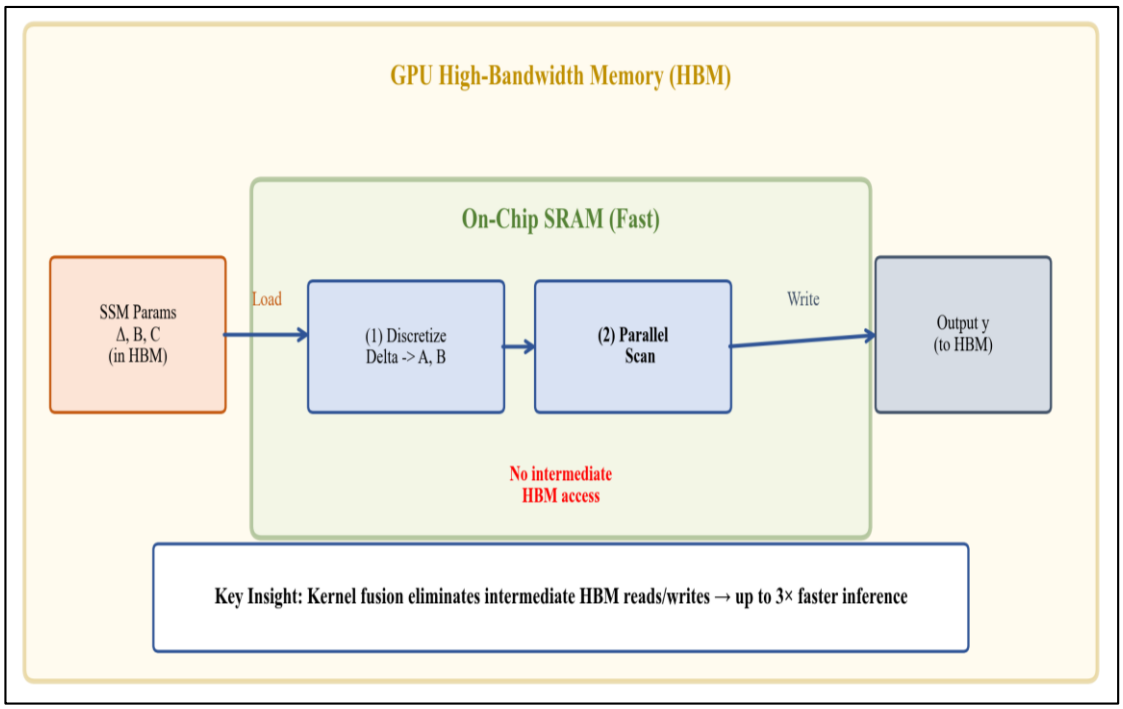

Fig 2: Hardware-Aware Selective Scan Algorithm

Hardware-aware selective scan algorithm showing data flow between HBM and SRAM, with kernel fusion eliminating intermediate materialization. Adapted from [14].

D. Architecture and Scaling Results

The Mamba architecture replaces the transformer block's attention + MLP structure with a single gated block containing:

- A linear projection expanding the input dimension;
- A 1d convolution for local context;
- The selective ssm;
- A silu nonlinearity and multiplicative gate; and
- A linear projection back to the model dimension. This design eliminates the need for separate attention and mlp components, reducing parameter count and simplifying the architecture.

Scaling experiments from 130M to 1.3B parameters on the Pile dataset [24] demonstrate that Mamba matches or exceeds Transformer++ (transformer with modern improvements including RMSNorm, SwiGLU, and rotary embeddings) in perplexity while achieving significantly better throughput. At 1.3B parameters, Mamba achieves the same perplexity as a Transformer++ trained on 50% more tokens, suggesting superior data efficiency [14].

Table 2. Language Modeling Performance and Efficiency Comparison on the Pile

Model	Parameters	Pile Perplexity	Throughput (tok/s)	Memory (GB)
Transformer++	1.3B	8.56	16K	18.2
Mamba	1.3B	8.42	48K	9.8
Mamba	790M	8.69	72K	6.4
RWKV-4 [25]	1.5B	8.83	38K	11.1
RetNet [26]	1.3B	8.77	32K	12.5

V. STRUCTURED STATE SPACE DUALITY: MAMBA-2

Dao and Gu [15] established a theoretical framework called structured state space duality (SSD), which reveals that the selective SSM computation is mathematically equivalent to a structured form of attention with a specific semiseparable matrix mask. Specifically, the output of a selective SSM can be expressed as $y = M \odot (QK^T)V$, where $Q = C$, $K = B$, $V = x$, and M is a causal mask derived from the cumulative product of the state matrix eigenvalues.

This duality enables Mamba-2 to leverage both SSM-style recurrent computation and attention-style parallel computation, choosing the most efficient mode based on sequence length and hardware. For short sequences, the attention-like mode using matrix multiplications on tensor cores is faster; for long sequences, the SSM recurrence is more efficient. Mamba-2 achieves 2–8× faster training than Mamba-1 while maintaining comparable modeling quality [15].

The SSD framework also enables multi-head state spaces, analogous to multi-head attention, where multiple independent SSM heads with different state matrices process the same input in parallel. This further improves expressiveness and aligns the SSM framework with established transformer design principles [15].

VI. HYBRID ARCHITECTURES: COMBINING SSMS WITH ATTENTION

Despite Mamba's strong performance, pure SSM architectures exhibit limitations on tasks requiring precise information retrieval from context, such as multi-hop question answering and associative recall over very long contexts [16]. This has motivated hybrid architectures that interleave SSM layers with sparse attention layers, combining the efficiency of SSMs for most computation with the precision of attention for retrieval-heavy operations.

Jamba [16], developed by AI21 Labs, alternates Mamba layers with transformer layers in a ratio of approximately 7:1 (seven Mamba layers per one attention layer), additionally incorporating MoE layers for capacity. Jamba supports a 256K-token context window while fitting within a single 80GB GPU, achieving competitive performance with Mixtral 8×7B and Llama-2 70B across standard benchmarks.

Zamba [27] and Griffin [28] explore alternative hybridization strategies. Griffin combines gated linear recurrences with local attention windows, demonstrating that even a small proportion of attention layers (as few as 1 in 6) recovers most of the quality gap between pure SSMs and pure transformers. These results suggest that attention may serve a specialized role—precise token retrieval—while SSMs handle the bulk of sequential information processing.

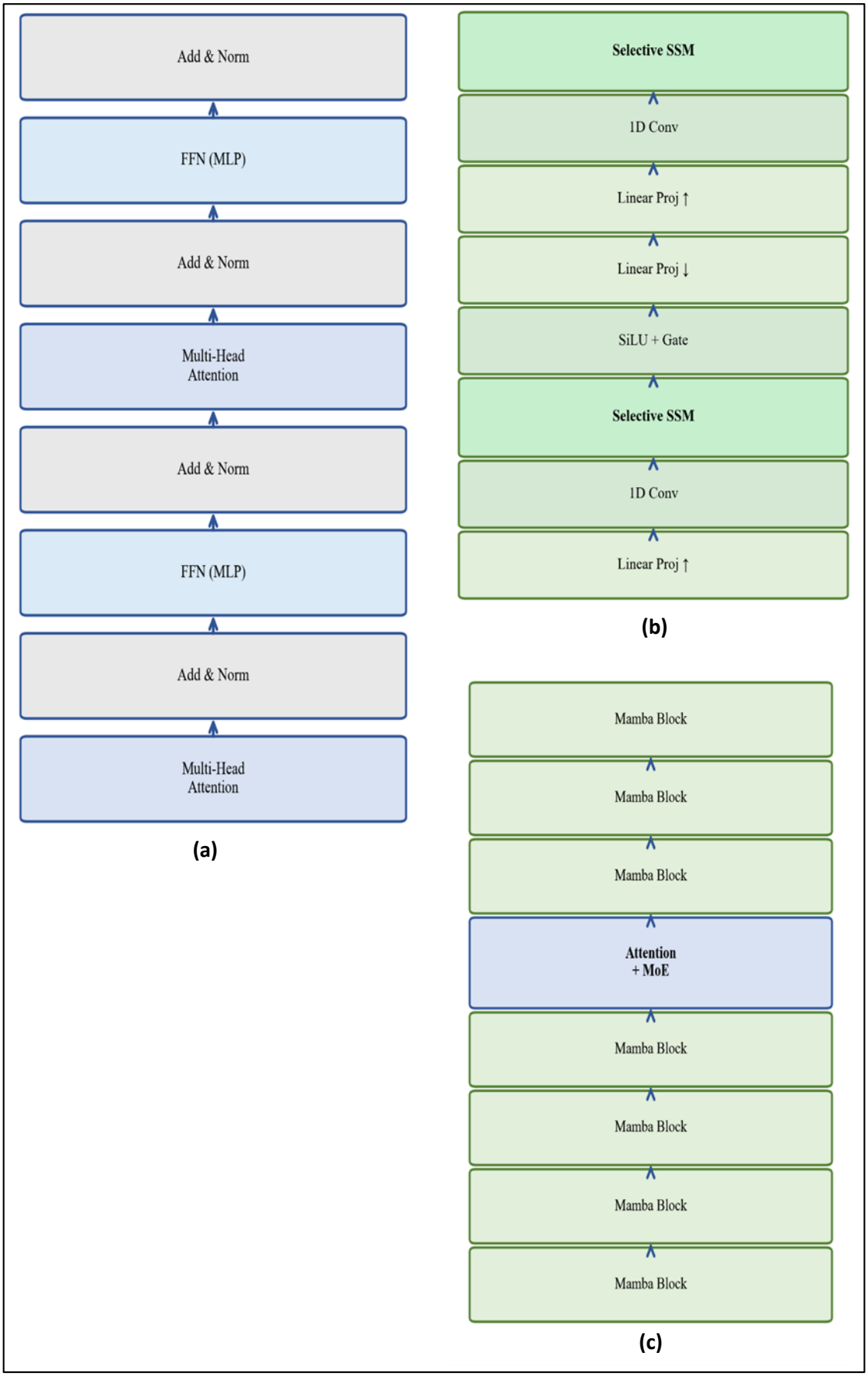

Fig 3: Architecture comparison: (a) Pure Transformer- O(N2) per layer, (b) Pure Mamba- O(N) per layer, (c) Hybrid Jamba- ~7:1 Mamba:Attn O(N) dominaant

Pure Transformer, pure Mamba, and hybrid Jamba architectures showing the interleaving of SSM and attention layers. Adapted from [16].

VII. APPLICATIONS BEYOND LANGUAGE MODELING

A. Genomic Sequence Analysis

Genomic sequences present extreme long-range dependency challenges, with regulatory elements influencing gene expression across millions of base pairs. HyenaDNA [29] applied SSM-based architectures to single-nucleotide-resolution genomic modeling, processing sequences up to 1 million tokens in length—far beyond the reach of transformer-based genomics models. The model achieved state-of-the-art performance on species classification and regulatory element prediction tasks while requiring 100× less computation than comparable attention-based approaches.

B. Computer Vision

Vision Mamba (Vim) [30] and VMamba [31] adapt the Mamba architecture for image recognition by scanning image patches in bidirectional or four-directional patterns. These models achieve competitive ImageNet classification accuracy with DeiT and Swin Transformer while offering linear scaling with image resolution, making them particularly attractive for high-resolution medical imaging and remote sensing applications.

C. Audio and Speech Processing

SSMs are natural candidates for audio processing, where sequences can span hundreds of thousands of time steps at standard sampling rates. SaShiMi [32] (State Space Model for Audio) demonstrated that S4-based architectures generate higher-quality raw audio waveforms than WaveNet and diffusion-based models while requiring fewer parameters and less computation. The model's ability to capture long-range temporal structure—such as musical themes spanning several seconds—highlights the advantages of SSMs for continuous signal processing.

Table 3. SSM Applications Across Diverse Domains

Domain	Model	Sequence Length	Key Result
Genomics	HyenaDNA [29]	1M tokens	SotA species classification
Vision	Vim [30]	Variable	Competitive with DeiT on ImageNet
Audio	SaShiMi [32]	160K samples	Superior to WaveNet generation
Video	VideoMamba [33]	Variable	SotA video understanding
Time Series	S4-TS [34]	Variable	SotA long-term forecasting

VIII. COMPARISON WITH ALTERNATIVE EFFICIENT ARCHITECTURES

SSMs are not the only approach to efficient sequence modeling. Linear attention models, such as RetNet [26] and TransNormerLLM [35], replace softmax attention with linear dot-product kernels, achieving O(N) complexity through the associativity of matrix multiplication. RWKV [25] combines elements of RNNs and attention through a time-mixing mechanism that can be computed either recurrently or in parallel. These approaches share SSMs' linear complexity advantage but differ in their theoretical foundations and empirical characteristics.

Empirical comparisons reveal nuanced trade-offs. On standard language modeling benchmarks at scales up to 7B parameters, Mamba, RWKV, and RetNet achieve broadly similar perplexity, with Mamba showing slight advantages [14]. However, on tasks requiring strong in-context learning (few-shot prompting, instruction following), transformers retain an edge, likely due to their explicit pairwise attention computation [36]. The gap narrows with hybrid approaches, suggesting that a small amount of attention is sufficient for strong in-context learning.

In terms of inference efficiency, SSMs and linear attention models share the advantage of constant per-step cost during autoregressive generation, eliminating the growing KV-cache that limits transformer inference for long sequences. At sequence length 64K, Mamba achieves approximately 5× higher throughput than FlashAttention-2-based transformers [14], making it particularly attractive for real-time and edge deployment scenarios.

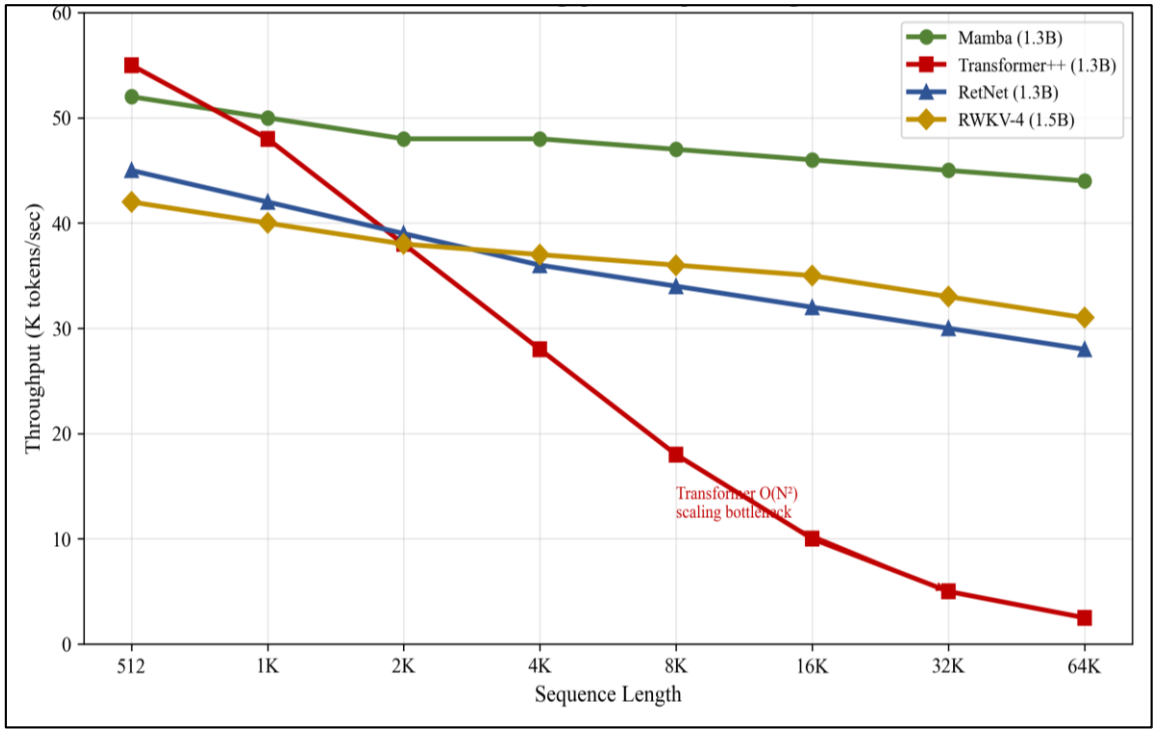

Fig 4: Inference Throughput vs. Sequence Length

Throughput comparison (tokens/second) across sequence lengths for Mamba, Transformer (FlashAttention-2), RetNet, and RWKV at the 1.3B parameter scale. Adapted from [14], [25], [26].

IX. OPEN CHALLENGES AND FUTURE DIRECTIONS

A. In-Context Learning and Reasoning

While Mamba demonstrates strong next-token prediction, its in-context learning capabilities—the ability to adapt behavior based on examples provided in the prompt—remain weaker than comparably sized transformers [36]. The fixed-dimensional state acts as an information bottleneck: a state of dimension N can store at most O(N) bits of information about the context, whereas attention computes over the full context. Addressing this limitation through expanded state dimensions, auxiliary retrieval mechanisms, or hybrid approaches is a critical research priority.

B. Hardware Co-Design

Current SSM implementations achieve efficiency through software-level optimizations (kernel fusion, memory management), but hardware architectures remain optimized for matrix multiplications—the dominant operation in transformers. The parallel scan operation central to SSMs has different computational characteristics that could benefit from specialized hardware support. Co-designing SSM algorithms with hardware accelerators represents an opportunity for substantial further efficiency gains [37].

C. Theoretical Understanding

Despite empirical success, the theoretical expressiveness of selective SSMs relative to transformers remains incompletely characterized. Recent work by Merrill et al. [38] shows that log-precision SSMs can simulate bounded-depth threshold circuits, while transformers simulate bounded-depth Boolean circuits—suggesting comparable but not identical computational classes. Developing tighter characterizations of SSM expressiveness and understanding which tasks fundamentally require attention-like computation are important theoretical questions.

D. Multimodal and Multi-Task Extensions

Extending SSMs to multimodal settings—processing interleaved text, images, audio, and video—is an active area of research. The linear complexity of SSMs makes them attractive for processing long multimodal sequences (e.g., video with aligned audio and text), but the optimal fusion strategies for combining SSM-processed features from different modalities remain unexplored. Similarly, the application of SSMs to encoder-decoder architectures for tasks like machine translation and summarization requires further investigation.

X. CONCLUSION

State space models have emerged as a principled and increasingly practical alternative to self-attention for sequence modeling. From the foundational S4 architecture through the selective state space paradigm of Mamba, SSMs have progressively closed the quality gap with transformers while maintaining fundamentally superior computational scaling. The structured state space duality framework unifies SSMs and attention under a common mathematical umbrella, enabling hybrid architectures that combine the strengths of both approaches.

The empirical evidence reviewed in this paper demonstrates that SSMs match transformer quality across language modeling, genomics, vision, and audio processing, while offering 3–5× throughput improvements for long sequences. Hybrid architectures like Jamba show that interleaving a small number of attention layers with SSM layers can recover the remaining quality gap at minimal computational cost.

Looking ahead, we identify in-context learning limitations, hardware co-design opportunities, theoretical expressiveness characterization, and multimodal extension as the most pressing open challenges. As SSM architectures mature and hardware support improves, we anticipate that the boundary between SSM and attention-based models will continue to blur, leading to a new generation of sequence models that adaptively select the most efficient computational primitive for each component of a task. The rapid pace of progress in this field—from S4's initial demonstration in 2022 to Mamba-2's state space duality framework in 2024—suggests that this convergence may occur sooner than expected.

REFERENCES

[1] Vaswani *et al*., "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, 2017, pp. 5998–6008.

[2] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. NAACL-HLT*, 2019, pp. 4171–4186.

[3] Dosovitskiy *et al*., "An image is worth 16×16 words: Transformers for image recognition at scale," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2021.

[4] Gulati *et al*., "Conformer: Convolution-augmented transformer for speech recognition," in *Proc. Interspeech*, 2020, pp. 5036–5040.

[5] J. Jumper *et al*., "Highly accurate protein structure prediction with AlphaFold," *Nature*, vol. 596, no. 7873, pp. 583–589, Aug. 2021.

[6] Y. Tay, M. Dehghani, D. Bahri, and D. Metzler, "Efficient transformers: A survey," *ACM Comput. Surv.*, vol. 55, no. 6, pp. 1–28, 2022.

[7] R. Child, S. Gray, A. Radford, and I. Sutskever, "Generating long sequences with sparse transformers," *arXiv preprint arXiv:1904.10509*, Apr. 2019.

[8] Katharopoulos, A. Vyas, N. Pappas, and F. Fleuret, "Transformers are RNNs: Fast autoregressive transformers with linear attention," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2020, pp. 5156–5165.

[9] K. Choromanski *et al*., "Rethinking attention with Performers," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2021.

[10] Gu, K. Goel, and C. Ré, "Efficiently modeling long sequences with structured state spaces," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2022.

[11] Gu, A. Gupta, K. Goel, and C. Ré, "On the parameterization and initialization of diagonal state space models," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, 2022.

[12] J. T. H. Smith, A. Warrington, and S. Linderman, "Simplified state space layers for sequence modeling," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2023.

[13] D. Y. Fu *et al*., "Hungry hungry hippos: Towards language modeling with state space models," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2023.

[14] Gu and T. Dao, "Mamba: Linear-time sequence modeling with selective state spaces," *arXiv preprint arXiv:2312.00752*, Dec. 2023.

[15] T. Dao and A. Gu, "Transformers are SSMs: Generalized models and efficient algorithms through structured state space duality," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2024.

[16] O. Lieber *et al*., "Jamba: A hybrid transformer-Mamba language model," *arXiv preprint arXiv:2403.19887*, Mar. 2024.

[17] K. Ogata, *Modern Control Engineering*, 5th ed. Upper Saddle River, NJ, USA: Prentice Hall, 2010.

[18] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE Trans. Neural Netw.*, vol. 5, no. 2, pp. 157–166, Mar. 1994.

[19] Gu, T. Dao, S. Ermon, A. Rudra, and C. Ré, "HiPPO: Recurrent memory with optimal polynomial projections," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, 2020, pp. 1474–1487.

[20] Y. Tay *et al*., "Long range arena: A benchmark for efficient transformers," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2021.

[21] Gupta, A. Gu, and J. Berant, "Diagonal state spaces are as effective as structured state spaces," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, 2022.

[22] G. E. Blelloch, "Prefix sums and their applications," Carnegie Mellon University, Tech. Rep. CMU-CS-90-190, 1990.

[23] T. Dao, D. Y. Fu, S. Ermon, A. Rudra, and C. Ré, "FlashAttention: Fast and memory-efficient exact attention with IO-awareness," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, 2022.

[24] L. Gao *et al*., “The Pile: An 800GB dataset of diverse text for language modeling,” *arXiv preprint arXiv:2101.00027*, Jan. 2021.
[25] Peng *et al*., “RWKV: Reinventing RNNs for the transformer era,” in *Proc. Conf. Empirical Methods Nat. Lang. Process. (EMNLP)*, 2023.
[26] Y. Sun *et al*., “Retentive network: A successor to transformer for large language models,” *arXiv preprint arXiv:2307.08621*, Jul. 2023.
[27] P. Glorioso *et al*., “Zamba: A compact 7B SSM hybrid model,” *arXiv preprint arXiv:2405.16712*, May 2024.
[28] S. De *et al*., “Griffin: Mixing gated linear recurrences with local attention for efficient language models,” *arXiv preprint arXiv:2402.19427*, Feb. 2024.
[29] E. Nguyen *et al*., “HyenaDNA: Long-range genomic sequence modeling at single nucleotide resolution,” in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, 2023.
[30] L. Zhu, B. Liao, Q. Zhang, X. Wang, W. Liu, and X. Wang, “Vision Mamba: Efficient visual representation learning with bidirectional state space model,” in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2024.
[31] Y. Liu et al., "VMamba: Visual state space model," arXiv preprint arXiv:2401.10166, Jan. 2024.
[32] K. Goel et al., "It's raw! Audio generation with state-space models," in Proc. Int. Conf. Mach. Learn. (ICML), 2022.
[33] K. Li et al., "VideoMamba: State space model for efficient video understanding," in Proc. Eur. Conf. Comput. Vis. (ECCV), 2024.
[34] Z. Zhou, L. Ma, and T. Liu, "Effectively modeling time series with state space models," in Proc. Int. Conf. Learn. Represent. (ICLR), 2023.
[35] Z. Qin et al., "TransNormerLLM: A faster and better large language model with improved TransNormer," arXiv preprint arXiv:2307.14995, Jul. 2023.
[36] H. Park et al., "Can Mamba learn how to learn? A comparative study on in-context learning tasks," in Proc. Int. Conf. Mach. Learn. (ICML), 2024.
[37] V. Rajapakse et al., "Hardware-efficient sequence modeling: A survey," IEEE Trans. Circuits Syst., 2024.
[38] W. Merrill, D. Sabané, and A. Petty, "The illusion of state in state-space models," in Proc. Int. Conf. Mach. Learn. (ICML), 2024.