PEER-REVIEWED JOURNAL OF COMPUTER SCIENCE (PRJCS)

ISSN Online: 3139-5082 Open Access Scholarly Journal

State-Space Models and Efficient Long-Sequence Architectures Beyond the Transformer

V Sakhtipriyanka

Assistant Professor, Department of Computer Science, Kongunadu Arts and Science College, Coimbatore, India.

Article information

Received: 6th June 2026
Received in revised form: 27th June 2026
Accepted: 5th July 2026
Available online: 10th July 2026

Volume: 1
Issue: 7
DOI: https://doi.org/10.5281/zenodo.21305118

Abstract

The self-attention mechanism that powers modern Transformers scales quadratically in time and memory with sequence length, making very long context expensive and often impractical. This survey reviews the emerging family of state-space models (SSMs) and related sub-quadratic architectures that aim to preserve sequence-modeling quality while achieving linear or near-linear cost. Beginning with the complexity wall faced by recurrent networks and attention, the article traces the deep SSM line from the HiPPO theory of continuous memory through the Structured State Space sequence model (S4), its diagonal simplifications, and the parallel-scan formulation of S5. It then examines selective SSMs and Mamba, whose input-dependent parameters and hardware-aware parallel scan deliver linear-time training and constant-memory autoregressive inference. Parallel developments in linear attention, RWKV, RetNet, and the exact but memory-efficient FlashAttention kernel are placed in context, along with hybrid designs such as Jamba that interleave attention and SSM blocks, and the state-space duality that unifies both views in Mamba-2. Applications spanning language, audio, genomics, and vision are summarized. The survey closes with an honest account of empirical trade-offs, including the associative-recall gap between recurrent SSMs and attention, and outlines open challenges. Reported numbers are illustrative and labeled as representative rather than definitive.

Keywords: - State-Space Models, Mamba, S4, Long-Sequence Modeling, Linear Attention, Efficient Transformers, Selective Scan, Sequence Architectures.

I. INTRODUCTION

The Transformer architecture has become the default engine for sequence modeling across natural language, audio, biology, and vision [1]. Its central component, self-attention, lets every position in a sequence interact directly with every other position, which captures long-range dependencies without the sequential bottleneck of recurrent networks. That flexibility carries a structural cost: for a sequence of length n, self-attention forms an n by n interaction matrix, so both computation and memory grow as O(n squared). For short passages this cost is negligible, but as applications move toward book-length documents, high-resolution audio, genomic sequences of hundreds of thousands of tokens, and long agentic contexts, the quadratic term dominates and long context becomes expensive or infeasible [2].

The pressure to model longer sequences has driven a large and fragmented literature on efficient alternatives. Early efforts kept the attention formulation but sparsified or approximated it, trading exactness for cost [2]. A more recent and increasingly influential direction abandons the pairwise interaction matrix altogether and returns to a recurrent, state-based view of sequences, now expressed through the mathematics of linear state-space models (SSMs). Architectures in this line, most prominently the Structured State Space model (S4) and its selective successor Mamba, promise linear-time training and constant-memory inference while remaining competitive on long-range benchmarks [3].

It is worth being precise about what long context demands. A model can technically accept a long input yet fail to use it, so the goal is not merely a large window but efficient and effective use of distant information. Two costs matter in practice. Training cost governs how much data of a given length a model can learn from within a fixed budget, and it is dominated by the O(n squared) attention term for long sequences. Inference cost governs deployment, where autoregressive generation must both compute each new token and store enough of the past to condition on it. Transformers pay for the latter through a key-value cache that grows linearly with the context already produced, so memory, not just compute, becomes the limiting resource in long-form generation. State-space models attack both costs at once, which is why they have attracted attention across research and production settings alike.

This survey provides a structured and deliberately honest overview of state-space models and the broader ecosystem of efficient long-sequence architectures. It develops the motivation from complexity first, then follows the deep SSM line from its theoretical foundations to selective SSMs, situates linear attention, RWKV, RetNet, and FlashAttention as complementary approaches, reviews hybrid designs, surveys applications across modalities, and gives a candid comparison of empirical trade-offs. Throughout, any specific throughput or accuracy figures are presented as representative illustrations drawn from the cited literature rather than as precise, hardware-independent guarantees. The aim is a clear map of a fast-moving field, including where SSMs genuinely help and where attention still holds an advantage.

II. BACKGROUND: RNNS, TRANSFORMERS, AND THE COMPLEXITY WALL

Sequence models map an ordered input to an ordered output while accounting for dependencies among positions. Recurrent neural networks (RNNs) do this by maintaining a hidden state that is updated one step at a time. Because the state has fixed size, an RNN processes a sequence in O(n) time and O(1) memory per step at inference, which is attractive for streaming and long contexts. The price is a strictly sequential recurrence that resists parallel training, together with optimization difficulties such as vanishing and exploding gradients that historically limited the effective memory horizon of vanilla RNNs and even gated variants.

A. Self-Attention and Its Cost

The Transformer replaced recurrence with self-attention, in which queries, keys, and values are computed for every position and each query attends to all keys through a softmax-weighted sum [1]. This removes the sequential dependency during training, so an entire sequence can be processed in parallel, and it gives every position direct access to every other position. The mechanism is expressive and highly parallelizable, which is a large part of why Transformers scale so well on modern accelerators. The unavoidable consequence is the n by n attention matrix: time and memory both scale as O(n squared) with sequence length, and autoregressive decoding must cache keys and values whose size grows linearly in the context already generated [2].

The quadratic term is not merely a constant-factor nuisance. Doubling the context quadruples attention compute, so moving from a few thousand to a few hundred thousand tokens raises the attention cost by orders of magnitude. The growing key-value cache also becomes a memory bottleneck during generation. These pressures motivated the family of efficient Transformers, including sparse-attention models such as Longformer that restrict each token to a local window plus a few global tokens [4], low-rank approximations such as Linformer that project keys and values to a smaller dimension [5], and kernel-based approximations such as the Performer that rewrite softmax attention using random features [6].

B. Linear Attention as a Recurrent View

A pivotal observation is that attention with a suitable kernel feature map can be reordered so that keys and values are aggregated before multiplication by queries. This linear-attention formulation replaces the explicit n by n matrix with a running summary and reduces cost to O(n) in sequence length, while exposing an equivalence between such attention and a linear recurrent network [7]. Linear attention thus foreshadows the state-space viewpoint: a bounded recurrent state can, in principle, summarize an unbounded past. The open question that the rest of this survey addresses is how to design that state and its update so the model both trains efficiently in parallel and retains enough of the past to remain competitive with full attention.

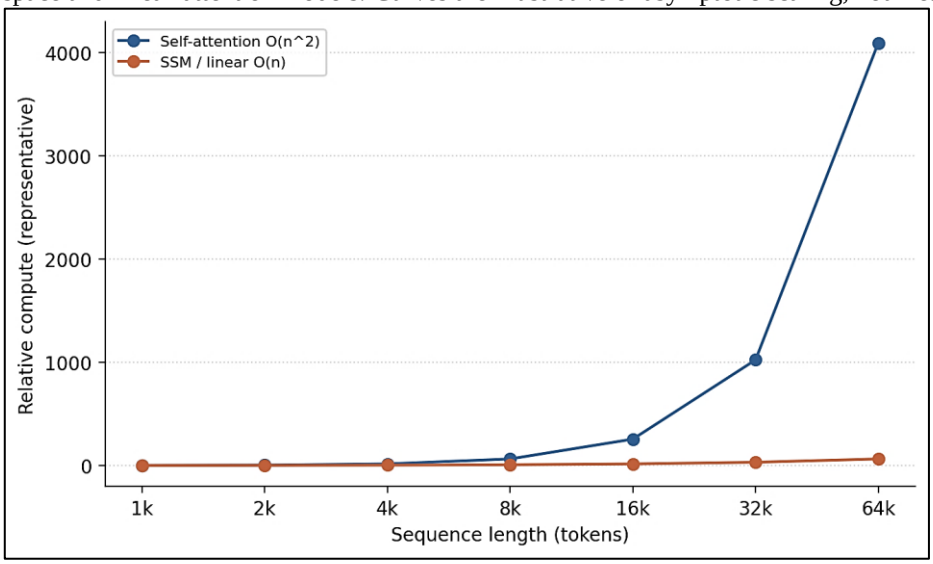
This bounded-state idea also exposes the fundamental tension that runs through the entire field. Full attention keeps a lossless record of the past, every key and value, and pays for it with quadratic compute and a linearly growing cache. A recurrent state, whether in an RNN, a linear-attention model, or an SSM, keeps a lossy but constant-size summary and pays a much smaller and bounded cost. Everything that follows can be read as a search for state representations and update rules that make this compression as painless as possible, so that little of the information that a given task needs is discarded. Different architectures place the compromise at different points, and no single choice is optimal for every workload.

Table 1. Representative Complexity and Inference Cost by Architecture Class (n is Sequence Length; Illustrative, Leading-Order Terms Only).

Architecture class	Train time / length	Train memory	Autoregressive step	State / cache growth
RNN / gated RNN	O(n) sequential	O(1) per step	O(1)	Fixed hidden state

Full self-attention	O(n squared)	O(n squared)	O(n)	KV cache grows with n
Sparse / windowed attention	O(n w)	O(n w)	O(w)	Local window w
Linear attention	O(n)	O(1) state	O(1)	Fixed feature state
Structured SSM (S4/S5)	O(n log n) or O(n)	O(1) state	O(1)	Fixed latent state
Selective SSM (Mamba)	O(n)	O(1) state	O(1)	Fixed latent state

Fig. 1: Schematic, representative growth of relative compute cost with sequence length for quadratic self-attention versus linear-cost state-space and linear-attention models. Curves are illustrative of asymptotic scaling, not measured timings.



III. STRUCTURED STATE-SPACE MODELS: FROM HIPPO TO S4

Classical state-space models describe how a latent state evolves under an input signal and produces an output. In continuous time a linear SSM is written with a state matrix A, input matrix B, and output matrix C, so the hidden state derivative depends linearly on the current state and input, and the output is a linear readout of the state. Discretizing this system with a learnable step size yields a linear recurrence that can be unrolled over a sequence, and, crucially, the same system can be written as a convolution of the input with a fixed kernel. This dual recurrent-and-convolutional nature is the mechanical foundation of the deep SSM line: the recurrent form gives cheap streaming inference, while the convolutional form gives parallel training.

A. HiPPO and Principled Memory

A naive SSM with random matrices does not remember much. The HiPPO framework supplied the missing theory by deriving specific state matrices that optimally compress a signal's history onto a basis of orthogonal polynomials, so that the fixed-size state approximates the entire past under a chosen measure [8]. HiPPO reframed memory as online function approximation and provided a principled initialization that lets a linear recurrence retain information over very long horizons. This result turned SSMs from a numerically fragile idea into a practical building block and directly enabled the architectures that followed.

B. S4 and Structured State Spaces

The Structured State Space sequence model (S4) made deep SSMs efficient and stable enough to compete on long-range tasks [3]. Its key contributions were a HiPPO-based initialization for long memory and a structured parameterization of the state matrix, expressed as a normal-plus-low-rank form, that allows the convolution kernel to be computed efficiently without materializing the full recurrence. S4 delivered strong results on the Long Range Arena benchmark suite, which was designed specifically to stress models on sequences of thousands to tens of thousands of steps and on which standard Transformers had struggled [9]. S4 demonstrated that a carefully constructed linear recurrence could match or exceed attention on tasks demanding very long-range reasoning, at substantially lower asymptotic cost.

C. Diagonal and Simplified Variants

The dual nature of the SSM is worth emphasizing because it is the source of the family's efficiency. During training, when the full sequence is available, the linear time-invariant recurrence can be unrolled as a global convolution and computed in parallel using fast transforms, so there is no sequential bottleneck. During inference, the same parameters define a step-by-step recurrence with a fixed-size hidden state, so generation is cheap and constant in memory. A single set of learned parameters therefore supports two complementary computation modes, and the model pays the parallel-training cost only once while enjoying recurrent inference thereafter. This is precisely the combination that RNNs and Transformers each achieved only in part.

S4's normal-plus-low-rank structure is powerful but intricate to implement. Subsequent work showed that much of the benefit survives with a purely diagonal state matrix. Diagonal State Spaces (DSS) demonstrated that a diagonal parameterization is simpler and nearly as effective [10], and a follow-up analysis of the parameterization and initialization of diagonal SSMs, often called S4D, clarified which choices matter and provided practical recipes for stable diagonal models [11]. In parallel, S5 simplified the layer further by using a single multi-input multi-output SSM computed with an associative parallel scan rather than a bank of independent convolutions, which streamlined the architecture while preserving long-range performance [12]. Together these variants made SSM layers easier to build, more numerically robust, and better matched to hardware, setting the stage for input-dependent selection.

IV. SELECTIVE STATE SPACES AND MAMBA

A limitation shared by S4, S5, and their diagonal cousins is that their dynamics are linear time-invariant: the matrices A, B, and C do not depend on the input, so the same recurrence is applied at every position. This makes them fast, because the recurrence reduces to a convolution, but it also makes them content-agnostic. A time-invariant SSM cannot easily decide to remember a particular token, ignore filler, or reset its state at a boundary, which limits its ability to perform context-dependent operations such as selective copying or associative recall.

A. Selection Through Input-Dependent Parameters

Mamba addresses this by making the SSM selective: the step size and the input and output projections become functions of the current input, so the model can modulate how information flows into and out of the state based on content [13]. This selection mechanism gives the recurrence a gating-like ability to propagate relevant tokens and discard irrelevant ones, recovering a capability that time-invariant SSMs lacked and that had been a strength of attention. The cost is that input-dependent parameters break the convolutional view, because there is no longer a single fixed kernel to convolve with the sequence.

B. Hardware-Aware Parallel Scan

To keep training efficient without the convolution shortcut, Mamba computes the now time-varying recurrence with a hardware-aware parallel scan implemented as a fused kernel [13]. The scan keeps the expanded state in fast on-chip memory, avoids writing large intermediate tensors to slower memory, and recomputes what is needed during the backward pass, so training remains linear in sequence length and efficient on modern accelerators. The design is explicitly co-created with the memory hierarchy of the target hardware, echoing the kernel-level philosophy that also underlies FlashAttention.

C. Linear-Time Inference and Language Results

Because Mamba is fundamentally a recurrence, autoregressive inference proceeds one step at a time with a fixed-size state and no growing key-value cache, giving constant memory per step and throughput that does not degrade as the generated context lengthens [13]. Reported language-modeling results placed Mamba on par with or ahead of comparably sized Transformers on several benchmarks while offering markedly higher inference throughput at long context, and the architecture removed attention entirely from its blocks. These results are best read as representative of the setting studied; independent evaluations later refined the picture, as discussed in the trade-offs section.

Conceptually, selection is what separates Mamba from the earlier deep SSM line and aligns it with gated recurrent networks and attention. A time-invariant SSM applies the same filter to every input, which is ideal for signals with stationary structure but poor for language, where the relevance of a token depends sharply on context. By letting the effective time constant vary with the input, Mamba can hold a value in state across many irrelevant tokens and then release it, or it can rapidly forget once a boundary is crossed. This dynamic control is the mechanism behind its improved performance on synthetic selective-copying and induction tasks that defeat time-invariant SSMs, and it is why selection, rather than raw state size alone, is treated as the core contribution.

V. RELATED EFFICIENT APPROACHES AND HYBRID ARCHITECTURES

SSMs are one branch of a broader effort to escape quadratic attention, and several parallel lines both compete and combine with them. Understanding these neighbors clarifies what is genuinely novel about selective SSMs and what is shared across the efficient-sequence landscape.

A. Linear Attention, RWKV, and RetNet

Linear-attention models replace softmax with a kernel feature map so that attention can be computed as a recurrence over a fixed-size state, achieving linear cost and constant-memory decoding [7]. RWKV builds a full language-model architecture in this spirit, combining a linearly scaling, RNN-like recurrence for inference with a formulation that can be trained in parallel like a Transformer, and it demonstrated that such models scale to billions of parameters [14]. The Retentive Network (RetNet) introduces a retention mechanism that admits three equivalent computation modes, a parallel form for training, a recurrent form for efficient inference, and a chunkwise form that interpolates between them, and reports strong throughput and memory advantages at longer sequences [15]. These architectures and SSMs share the same core bargain: a bounded recurrent state in place of an unbounded attention matrix.

B. Long-Convolution Models

A related family replaces attention with long, implicitly parameterized convolutions. H3 augmented SSM layers with multiplicative gating to close much of the gap with attention on language while remaining sub-quadratic [16], and Hyena generalized this into a hierarchy of long convolutions and data-controlled gating that scales sub-quadratically and approaches attention quality on several tasks [17]. These models are conceptually close to SSMs, since a long convolution and a linear recurrence are two views of the same operation, and they helped motivate the selection idea later realized in Mamba.

C. FlashAttention as a Complementary Exact Method

Not every efficiency gain requires abandoning attention. FlashAttention keeps exact softmax attention but reorganizes its computation to be aware of the GPU memory hierarchy, tiling the computation and avoiding materialization of the full attention matrix in slow memory, which yields large speedups and memory savings without any approximation [18]. FlashAttention does not change the O(n squared) asymptotic compute, but it dramatically lowers the constant factor and the memory footprint, so it remains the practical baseline that sub-quadratic models must beat and is often used inside the attention layers of hybrid models.

D. Hybrids and State-Space Duality

The most pragmatic recent designs mix attention and SSM layers. Jamba interleaves Transformer and Mamba blocks and adds mixture-of-experts layers, aiming to combine the recall strength of attention with the throughput and long-context memory efficiency of SSMs, and it reports high throughput and a small memory footprint at long context relative to comparable Transformers [19]. On the theoretical side, Mamba-2 established a structured state-space duality (SSD) showing that a broad class of SSMs and a form of masked linear attention are two views of the same computation, which lets a selective SSM be implemented with matrix-multiplication primitives and inherit system optimizations developed for Transformers, improving training speed [20]. This duality reframes the SSM-versus-attention divide as a spectrum rather than a dichotomy.

VI. APPLICATIONS ACROSS MODALITIES

The appeal of linear-time sequence models is strongest where sequences are long, and SSMs have been applied across several such domains.

A. Language

Language modeling has been the primary proving ground. Mamba and RWKV both target autoregressive language modeling at scale, and hybrids such as Jamba are explicitly built as long-context language models [13][14][19]. The recurring motivation is constant-memory decoding and stable throughput as context grows, which matters for long documents, retrieval-augmented generation, and extended dialogues where a Transformer's key-value cache becomes a bottleneck.

B. Audio

Raw audio is a natural fit because waveforms contain tens of thousands of samples per second, producing very long sequences. State-space models were shown to be strong autoregressive generators of raw audio, with SaShiMi using stacked S4 layers to model waveforms and capture long-range structure more effectively than comparable baselines [21]. The long-memory property of HiPPO-initialized SSMs directly addresses the challenge that meaningful audio structure spans many timescales.

C. Genomics

Genomic sequences can extend to hundreds of thousands of base pairs, far beyond typical Transformer context windows. Long-convolution and SSM-style models such as HyenaDNA operate at single-nucleotide resolution over very long contexts, using sub-quadratic scaling to reach sequence lengths that would be prohibitive for full attention [22]. This illustrates a domain where the asymptotic advantage of linear-cost models translates into qualitatively new capability rather than a marginal speedup.

D. Vision

Images and videos become long token sequences once flattened, so vision has adopted SSMs as backbones. Vision Mamba (Vim) applies bidirectional Mamba blocks with position embeddings to image patches and reports competitive accuracy on classification, detection, and segmentation while improving computation and memory efficiency relative to established vision Transformers on higher-resolution inputs [23]. Bidirectionality matters here because images have no causal ordering, which distinguishes vision SSMs from their autoregressive language counterparts.

VII. EMPIRICAL TRADE-OFFS AND HONEST COMPARISON

A balanced assessment must separate the clear structural advantages of SSMs from tasks where attention retains an edge. The advantages are consistent and well founded. Linear-time training and constant-memory, cache-free autoregressive inference give SSMs superior throughput and memory behavior as sequence length grows, and this scaling is the primary reason to prefer them for very long contexts [13][20]. Figure 2 shows a representative pattern in which

recurrent SSM decoding throughput stays roughly flat as context lengthens while attention throughput falls off as its cache and per-step cost grow.

A. The Recall Gap

The most important qualification concerns associative recall, the ability to look up a specific earlier token given a later cue, which underlies in-context learning and copying. Because a recurrent SSM compresses the past into a fixed-size state, it can lose exact details that full attention retains by keeping every key and value. Controlled studies of recall in efficient language models found that pure sub-quadratic recurrences trade recall for throughput, and that this gap is a systematic property of fixed-state models rather than an artifact of scale [24]. Selection mitigates but does not fully eliminate the effect.

Independent evaluations of Mamba-based language models corroborated this nuanced view: at matched scale, pure SSM models can match Transformers on many tasks yet lag on those that require precise long-range copying or recall, whereas hybrid models that reintroduce a small number of attention layers recover much of the deficit while retaining most of the efficiency benefit [25]. This is a central reason the field has converged on hybrids such as Jamba [19] and on the duality perspective of Mamba-2 [20], which let designers place attention exactly where exact recall is needed and use SSM layers everywhere else.

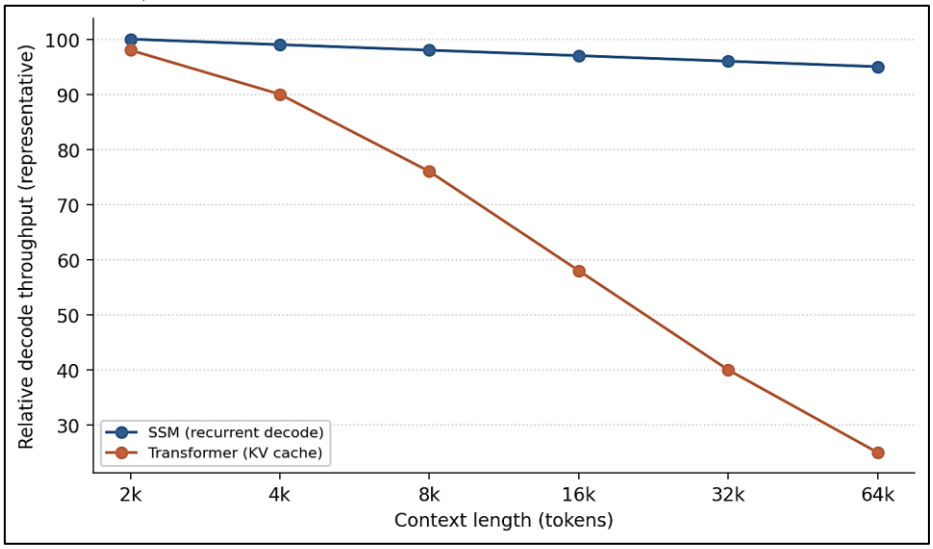
B. Comparative Summary

Table 2 summarizes salient properties of the main efficient architectures discussed here. The entries are qualitative and representative; precise rankings depend on scale, task, implementation, and hardware, and should be verified against primary sources for any specific deployment.

Table 2. Qualitative Comparison of Representative Efficient Sequence Architectures (Illustrative).

Model	Core mechanism	Length scaling	Input-dependent	Parallel training	Recall vs attention
Transformer	Softmax self-attention	O(n squared)	Yes (via attention)	Yes	Reference (strong)
S4 / S4D / S5	Time-invariant structured SSM	O(n log n) / O(n)	No	Yes (convolution/scan)	Weaker on lookup
Mamba	Selective SSM + scan	O(n)	Yes	Yes (parallel scan)	Improved, still a gap
RWKV	Linear recurrence + gating	O(n)	Yes	Yes	Weaker on lookup
RetNet	Retention (multi-mode)	O(n)	Partial	Yes	Weaker on lookup
Jamba (hybrid)	Attention + Mamba + MoE	Near-linear	Yes	Yes	Near attention

Fig. 2: Representative autoregressive decoding throughput versus context length: a recurrent SSM sustains roughly constant per-step throughput while attention throughput declines as its key-value cache and per-step cost grow. Values are schematic and normalized, not measured benchmarks.



It also helps to be clear about why the recall gap arises rather than treating it as a mysterious deficiency. Exact associative recall over a long context requires, in the worst case, the ability to retrieve any one of many earlier items with full fidelity, which is information that a fixed-size state cannot store without loss once the number of distinguishable items exceeds its capacity. Attention sidesteps this by never compressing: it retains every key and value and pays the quadratic

price. A recurrent model can enlarge its state or use selection to prioritize likely-needed items, but it cannot escape the basic capacity argument. This is why the gap narrows with larger states and better selection yet does not vanish, and why interleaving even a few attention layers is so effective, since those layers restore exact lookup exactly where it is required [25].

A further practical consideration is maturity. Attention benefits from years of tuned kernels, quantization support, and tooling, and FlashAttention already narrows its memory disadvantage without approximation [18]. SSM kernels are improving rapidly but are less universally optimized, so measured wall-clock gains vary by implementation and hardware. The honest summary is that SSMs win decisively on asymptotic cost and long-context memory, are competitive on aggregate language quality, and remain weaker than attention on tasks demanding exact recall, which is precisely why hybrids are currently the most robust choice.

VIII. OPEN CHALLENGES AND FUTURE DIRECTIONS

Several questions remain open as the field matures. Closing the recall gap without giving up linear-time efficiency is the most pressing. Approaches include larger or better-structured states, learned selection that better preserves salient tokens, and principled hybridization that decides where attention is truly necessary [24][25]. The state-space duality result suggests a productive middle ground in which a single framework can be tuned along the recurrence-to-attention spectrum per layer [20].

A second direction is theoretical understanding. The expressive power of selective SSMs, their inductive biases, and the precise class of functions they can and cannot represent are only partially characterized. Clarifying what selection buys over time-invariant SSMs, and how it relates to gating in RNNs and to masked linear attention, would guide architecture design rather than trial and error [11][20].

Third, systems and hardware co-design will shape practical adoption. Mamba's gains depend on carefully engineered scan kernels, and broad deployment needs mature support for quantization, distributed training, variable-length batching, and inference serving comparable to what Transformers enjoy [13][19]. The wider trajectory of large language models, whose capabilities and ethical challenges have been surveyed elsewhere, adds urgency to this efficiency agenda, since serving ever-longer contexts at scale is costly [26]. The same efficiency argument motivates small language models for on-device and private intelligence, where constant-memory, cache-free recurrent inference is especially attractive under tight compute and memory budgets [27]. Fourth, scaling behavior at the largest sizes is still being mapped: whether pure SSMs, hybrids, or duality-based models offer the best quality per unit of compute at frontier scale remains an active empirical question [25]. Finally, extending SSMs to structured, multimodal, and non-causal settings, as bidirectional vision SSMs began to do [23], and understanding their behavior under long-context retrieval and reasoning, are fertile areas where the linear-cost advantage could unlock capabilities that quadratic attention cannot reach economically.

IX. CONCLUSION

The quadratic cost of self-attention created a durable incentive to find sequence models that scale linearly with length, and state-space models have emerged as the most theoretically grounded response. From the HiPPO theory of continuous memory through the structured and diagonal S4 family and the parallel-scan S5, the deep SSM line established that a linear recurrence can match attention on long-range tasks at far lower asymptotic cost. Mamba added input-dependent selection and a hardware-aware scan to recover content-based behavior while keeping linear-time training and constant-memory inference, and Mamba-2 unified the recurrent and attention views through state-space duality.

Set alongside linear attention, RWKV, RetNet, long-convolution models, and the exact but memory-efficient FlashAttention kernel, SSMs are best understood as one region of a shared design space rather than a wholesale replacement for attention. The empirical record is encouraging but not triumphalist: SSMs deliver clear efficiency and long-context memory advantages and competitive quality, yet retain a measurable recall gap on tasks that demand exact lookup, which is why hybrid architectures such as Jamba currently offer the most balanced trade-off. As kernels mature, theory sharpens, and hybrids are refined, efficient long-sequence architectures are poised to make very long context routine. The figures and comparisons in this survey are representative rather than definitive, and specific claims should be confirmed against primary sources for any given application.

REFERENCES

[1] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is All You Need," in Advances in Neural Information Processing Systems (NeurIPS), 2017, pp. 5998–6008.

[2] Y. Tay, M. Dehghani, D. Bahri, and D. Metzler, "Efficient Transformers: A Survey," ACM Computing Surveys, vol. 55, no. 6, pp. 1–28, 2022.

[3] A. Gu, K. Goel, and C. Re, "Efficiently Modeling Long Sequences with Structured State Spaces," in International Conference on Learning Representations (ICLR), 2022.

[4] I. Beltagy, M. E. Peters, and A. Cohan, "Longformer: The Long-Document Transformer," arXiv preprint arXiv:2004.05150, 2020.

[5] S. Wang, B. Z. Li, M. Khabsa, H. Fang, and H. Ma, "Linformer: Self-Attention with Linear Complexity," arXiv preprint arXiv:2006.04768, 2020.

[6] K. Choromanski, V. Likhosherstov, D. Dohan, X. Song, A. Gane, T. Sarlos, P. Hawkins, J. Davis, A. Mohiuddin, L. Kaiser, D. Belanger, L. Colwell, and A. Weller, "Rethinking Attention with Performers," in International Conference on Learning Representations (ICLR), 2021.

[7] A. Katharopoulos, A. Vyas, N. Pappas, and F. Fleuret, "Transformers Are RNNs: Fast Autoregressive Transformers with Linear Attention," in International Conference on Machine Learning (ICML), 2020, pp. 5156–5165.

[8] A. Gu, T. Dao, S. Ermon, A. Rudra, and C. Re, "HiPPO: Recurrent Memory with Optimal Polynomial Projections," in Advances in Neural Information Processing Systems (NeurIPS), 2020, pp. 1474–1487.

[9] Y. Tay, M. Dehghani, S. Abnar, Y. Shen, D. Bahri, P. Pham, J. Rao, L. Yang, S. Ruder, and D. Metzler, "Long Range Arena: A Benchmark for Efficient Transformers," in International Conference on Learning Representations (ICLR), 2021.

[10] A. Gupta, A. Gu, and J. Berant, "Diagonal State Spaces Are as Effective as Structured State Spaces," in Advances in Neural Information Processing Systems (NeurIPS), 2022, pp. 22982–22994.

[11] A. Gu, K. Goel, A. Gupta, and C. Re, "On the Parameterization and Initialization of Diagonal State Space Models," in Advances in Neural Information Processing Systems (NeurIPS), 2022, pp. 35971–35983.

[12] J. T. H. Smith, A. Warrington, and S. W. Linderman, "Simplified State Space Layers for Sequence Modeling," in International Conference on Learning Representations (ICLR), 2023.

[13] A. Gu and T. Dao, "Mamba: Linear-Time Sequence Modeling with Selective State Spaces," arXiv preprint arXiv:2312.00752, 2023.

[14] B. Peng, E. Alcaide, Q. Anthony, A. Albalak, S. Arcadinho, S. Biderman, et al., "RWKV: Reinventing RNNs for the Transformer Era," in Findings of the Association for Computational Linguistics: EMNLP, 2023, pp. 14048–14077.

[15] Y. Sun, L. Dong, S. Huang, S. Ma, Y. Xia, J. Xue, J. Wang, and F. Wei, "Retentive Network: A Successor to Transformer for Large Language Models," arXiv preprint arXiv:2307.08621, 2023.

[16] D. Y. Fu, T. Dao, K. K. Saab, A. W. Thomas, A. Rudra, and C. Re, "Hungry Hungry Hippos: Towards Language Modeling with State Space Models," in International Conference on Learning Representations (ICLR), 2023.

[17] M. Poli, S. Massaroli, E. Nguyen, D. Y. Fu, T. Dao, S. Baccus, Y. Bengio, S. Ermon, and C. Re, "Hyena Hierarchy: Towards Larger Convolutional Language Models," in International Conference on Machine Learning (ICML), 2023, pp. 28043–28078.

[18] T. Dao, D. Y. Fu, S. Ermon, A. Rudra, and C. Re, "FlashAttention: Fast and Memory-Efficient Exact Attention with IO-Awareness," in Advances in Neural Information Processing Systems (NeurIPS), 2022, pp. 16344–16359.

[19] O. Lieber, B. Lenz, H. Bata, G. Cohen, J. Osin, I. Dalmedigos, et al., "Jamba: A Hybrid Transformer-Mamba Language Model," arXiv preprint arXiv:2403.19887, 2024.

[20] T. Dao and A. Gu, "Transformers Are SSMs: Generalized Models and Efficient

Algorithms Through Structured State Space Duality," in International Conference on Machine Learning (ICML), 2024.

[21] K. Goel, A. Gu, C. Donahue, and C. Re, "It's Raw! Audio Generation with State-Space Models," in International Conference on Machine Learning (ICML), 2022, pp. 7616–7633.

[22] E. Nguyen, M. Poli, M. Faizi, A. Thomas, C. Birch-Sykes, M. Wornow, et al., "HyenaDNA: Long-Range Genomic Sequence Modeling at Single Nucleotide Resolution," in Advances in Neural Information Processing Systems (NeurIPS), 2023, pp. 43177–43201.

[23] 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 International Conference on Machine Learning (ICML), 2024.

[24] S. Arora, S. Eyuboglu, A. Timalsina, I. Johnson, M. Poli, J. Zou, A. Rudra, and C. Re, "Zoology: Measuring and Improving Recall in Efficient Language Models," arXiv preprint arXiv:2312.04927, 2023.

[25] R. Waleffe, W. Byeon, D. Riach, B. Norick, V. Korthikanti, T. Dao, A. Gu, et al., "An Empirical Study of Mamba-Based Language Models," arXiv preprint arXiv:2406.07887, 2024.

[26] "Large Language Models: Architectures, Capabilities, and Ethical Challenges," Peer-Reviewed Journal of Computer Science (PRJCS), E-ISSN 3139-5082, 2025.

[27] "Small Language Models for On-Device and Private Intelligence," Peer-Reviewed Journal of Computer Science (PRJCS), E-ISSN 3139-5082, 2025.