Beyond Transformers: Next Generation Architectures for Efficient Large-Scale Learning

Abstract

The emergence of the Transformer architecture has fundamentally reshaped the landscape of artificial intelligence, yielding unprecedented performance across tasks in natural language processing, computer vision, and multimodal learning. However, the core of the Transformer—the self-attention mechanism—imposes a computational and memory complexity that scales quadratically (\mathcal{O}(N^2)) with the input sequence length N. This non-linear scaling creates a severe bottleneck, preventing the efficient processing of ultra-long contexts and imposing massive economic and environmental costs on resource-intensive large-scale training. This paper provides an exhaustive, expert-level analysis of alternative architectural paradigms that achieve linear (\mathcal{O}(N)) scaling, offering a necessary path for sustainable, truly large-scale machine learning. We systematically survey three distinct next-generation approaches: Structured State Space Models (SSMs) like Mamba, kernel-based Linear Attention methods such as the Performer, and sparse computation frameworks like Mixture-of-Experts (MoE). The analysis rigorously compares these architectures based on theoretical complexity, empirical throughput, memory efficiency, and representational fidelity, particularly concerning the modeling of long-range dependencies and complex in-context learning. We conclude that future success resides in modular, hybrid systems (e.g., Jamba) and specialized hardware co-design, which collectively address the architectural and system-level challenges inherent in transitioning beyond the Transformer era.

Keywords and Classification

Sequence Modeling, Computational Efficiency, State Space Models, Mamba, Linear Attention, Mixture-of-Experts, Quadratic Complexity, Hardware Acceleration, Large Language Models.

Table of Contents

1. Introduction

1.1 Background and Motivation

The Transformer architecture (Vaswani et al., 2017) catalyzed a revolution in deep learning, transitioning sequence modeling from traditional Recurrent Neural Networks (RNNs) and Convolutional Neural Networks (CNNs) to models optimized for massive parallelism. Its foundational success spans numerous domains. In Natural Language Processing (NLP), models like BERT and GPT demonstrate superior context modeling and scalability, forming the backbone of modern Large Language Models (LLMs). The architecture's influence extends into computer vision (Vision Transformers, ViT) and multimodality, where it enables tasks such as visual-question answering and generative modeling by requiring minimal inductive biases compared to convolutional designs. The ability of self-attention to model direct, long-range dependencies across sequence elements, coupled with efficient sequence-parallel processing, defined its rapid ascendancy.

Despite these capabilities, the standard Transformer architecture faces critical, intrinsic limitations rooted in its key mechanism: self-attention. The core computation involves calculating the attention score matrix via the dot product of Query (Q) and Key (K) matrices, Q K^{\top}. If the sequence length is denoted by N, and the hidden dimension by D, this operation necessitates the calculation and storage of an N \times N matrix. This operation results in a computational complexity that scales quadratically with the sequence length, denoted as \mathcal{O}(N^2 D).

This quadratic complexity poses a significant scaling crisis. As context lengths increase—a necessary feature for truly intelligent systems requiring expansive memory—the computational cost and memory consumption grow prohibitively fast. For training extremely large models or deploying them for inference on sequences exceeding N=10,000 tokens, the required latency and compute costs become unsustainable. This constraint not only limits the technical scope of problems solvable by current Transformer models but also imposes a substantial economic and environmental burden, as doubling the context length necessitates quadrupling the required computational resources, concentrating large-scale model development among organizations with access to exceptional compute capacity. The current paradigm is approaching its technical and financial asymptote, necessitating a fundamental architectural shift.

1.2 Problem Statement

The continued pursuit of scaling machine learning models—particularly in the context of extremely long documents, high-resolution imagery, or complex genomic sequences—is fundamentally constrained by the \mathcal{O}(N^2) complexity of the Transformer. To enable the next generation of truly scalable, resource-aware models, new architectures must be developed and adopted that maintain the powerful sequence modeling capacity of attention while strictly achieving linear (\mathcal{O}(N))

complexity in both computational time and memory consumption. This report addresses the viability and trade-offs of the most promising non-Transformer alternatives.

1.3 Contributions of the Paper

This paper contributes a tripartite analysis designed to guide future architectural design choices:

1. **Systematic Classification and Deep Mechanistic Analysis:** A thorough review of modern sequence architectures, including Structured State Space Models (SSMs), Linear Attention methods, and Mixture-of-Experts (MoE), detailing the underlying mathematical mechanisms that enable their linear scaling.
2. **Rigorous Comparative Analysis:** A comparison focused not only on theoretical complexity but on practical deployment metrics such as GPU throughput (tokens/second), memory footprint, and observed trade-offs in long-context understanding and in-context learning performance.
3. **Identification of Systemic Challenges:** An examination of critical hardware and distributed systems bottlenecks that govern the real-world efficiency of these new paradigms, emphasizing the need for co-design and optimization beyond the algorithmic level.

1.4 Paper Structure

The remainder of this paper is structured as follows: Section 2 reviews the theoretical and empirical bottlenecks of the standard Transformer. Section 3 details the architecture and mechanisms of three families of next-generation, linear-scaling models. Section 4 presents a comparative analysis of their performance metrics and functional trade-offs. Section 5 discusses future research directions and open challenges, and Section 6 concludes the study.

2. Related Work: The Transformer Paradigm

2.1 The Standard Transformer

The original Transformer proposed by Vaswani et al. (2017) relies solely on attention mechanisms, replacing the recurrence and convolution found in prior sequence models. The core of the architecture is the Self-Attention mechanism, where an input sequence is projected into three representations: Query (Q), Key (K), and Value (V). The output is computed by weighting the values based on the compatibility scores derived from Q and K.

The standard architecture employs Multi-Head Attention (MHSA), where the Q, K, V matrices are split into multiple "heads," allowing the model to jointly attend to information from different representation subspaces and at different positions. Critically, because the self-attention calculation is permutation-invariant, positional encoding is necessary to inject sequential order information into the input representations. The remainder of the standard Transformer block consists of residual connections, layer normalization, and a position-wise Feed-Forward Network (FFN). This successful foundational design has led to predictable performance scaling laws for Large Language Models (LLMs).

2.2 Efficiency Bottlenecks in Attention

The success of the Transformer is tempered by the profound inefficiency of its attention mechanism when dealing with long sequences.

Computational Complexity

The bottleneck stems directly from the calculation of the N \times N attention score matrix Q K^{\top}. For sequences of length N and hidden dimension D, the matrix multiplication of Q (N \times D) and K^{\top} (D \times N) requires \mathcal{O}(N^2 D) operations. As modern LLMs push sequence lengths to thousands of tokens, the N^2 term rapidly dominates the total floating-point operations (FLOPs), increasing latency and computational costs exponentially.

Memory Overhead

The memory consumption during training and inference is also significantly affected by the quadratic term. The storage of the full N \times N attention matrix consumes memory scaling as \mathcal{O}(N^2). Furthermore, during autoregressive decoding—a common mode for LLM deployment—the Key and Value states from previous timesteps (the KV cache) must be stored. While the KV cache itself scales linearly with the sequence length N and hidden dimension D (i.e., \mathcal{O}(ND)), the accumulation of cache states across multiple layers leads to high peak GPU memory usage, severely restricting the maximum context length that can be maintained during deployment. The combination of quadratic computation and substantial memory footprint makes the pure Transformer impractical for sequence lengths where N

extends beyond typical LLM limits.

2.3 Existing Approaches to Transformer Efficiency

A parallel line of research has focused on improving the efficiency of the standard Transformer structure without outright replacing the attention mechanism.

Sub-Quadratic Attention Variants

Established techniques aim to reduce the \mathcal{O}(N^2) complexity to sub-quadratic scaling (e.g., \mathcal{O}(N \log N) or \mathcal{O}(N \sqrt{N})) through modifications to the attention matrix calculation.

- **Sparse Attention:** These methods impose structural constraints on the attention graph, restricting each token's attention to a limited subset of other tokens. This restriction lowers the computational cost but introduces a critical design challenge: unlike dense attention, which captures global context in a single layer, sparse attention models must rely on deeper stacks of restricted attention layers to achieve comparable long-range feature propagation. The expressive capacity of sparse attention, while promising in practice, is still theoretically less understood than that of the dense Transformer.

Parameter Efficiency Techniques

Other methods focus on reducing the total parameter count or the compute required for fine-tuning.

- **Parameter Sharing:** This involves reusing the same set of weights across different positions or layers within the model. For example, parameter-efficient multi-task fine-tuning utilizes shared hypernetworks to generate task-specific adapter parameters, achieving knowledge sharing across tasks while adding only a small percentage (e.g., 0.29%) of new parameters per task, significantly reducing the memory footprint required for task adaptation.

These existing efficiency improvements, while valuable, primarily serve as evolutionary steps within the quadratic attention paradigm. Although they mitigate some limitations, they do not fundamentally resolve the core asymptotic problem, which remains the barrier to entry for genuinely massive sequence modeling. The models discussed in Section 3 represent a true architectural paradigm shift, abandoning the Q K^{\top} mechanism entirely.

3. Next Generation Architectures: The Shift

The pursuit of architectures exhibiting strict \mathcal{O}(N) complexity has led to the development of novel approaches that fundamentally diverge from the attention mechanism. These models replace token-to-token pairwise comparisons with recurrent state propagation or kernel approximations, or employ conditional computation to ensure sparse resource utilization.

3.1 Linear Complexity Attention Mechanisms

State-Space Models (SSMs): Mamba and the S-Family

Structured State Space Models (SSMs) have emerged as a highly efficient alternative to both RNNs and Transformers, specifically addressing the conflicting needs of long-range dependency modeling and computational efficiency.

Evolution from S4 to S5

The foundational model, S4 (Structured State Space Sequence Model), introduced an efficient parameterization of continuous-time dynamics using the HiPPO algorithm and frequency-domain computation. This approach successfully enabled long-range reasoning with linear time complexity. Successors, such as S5 (Simplified Structured State Space Sequence Model), simplified the architecture by reformulating S4's independent SISO (single-input single-output) systems into a unified MIMO (multiple-input multiple-output) structure. S5 maintains \mathcal{O}(N) runtime complexity and linear memory usage, while also allowing the handling of irregularly spaced data.

Mamba: The Selective State Space Model

Mamba builds upon the S4/S5 framework by introducing a crucial innovation: the Selective State Space (S6) mechanism. Traditional SSMs are time-invariant, meaning their internal state matrices are static, limiting their capacity for context-dependent reasoning. Mamba overcomes this by making the state transition matrices—specifically A, B, and C—functions of the input sequence x. This *selection mechanism* allows the model to dynamically determine which input tokens are relevant and should be compressed into the hidden state, effectively focusing and ignoring information based on the sequence context.

This selective ability yields two massive efficiency gains:

1. **Linear Time Complexity:** Mamba achieves \mathcal{O}(N D) time complexity for both training and inference.
2. **Constant Inference Memory:** Because the model selectively compresses information into a bounded hidden state representation, its memory usage during autoregressive inference is constant (\mathcal{O}(D)), regardless of the input sequence length N. This feature provides theoretically unlimited context

length, constrained only by hardware capacity.

Hardware-Aware Implementation

Mamba's practicality relies on specialized, hardware-aware algorithms. Although the SSM recurrence relation is sequential, Mamba enables efficient parallel computation during training via the **parallel scan (prefix sum) operation**. This technique maps the recurrence efficiently onto parallel hardware, such as GPUs, which are typically optimized for matrix multiplication. The result is a highly competitive model that demonstrates fast inference, often achieving throughput up to 5\times higher than comparable Transformers, and strong performance scaling up to sequences of million tokens. Mamba-3B models have even demonstrated performance matching Transformer models twice their size in pretraining and downstream evaluation.

Low-Rank and Kernel Methods: The Performer

An alternative approach to achieving linear complexity involves approximating the original quadratic attention matrix using kernel methods. The Performer architecture, utilizing the Fast Attention Via positive Orthogonal Random features (FAVOR+) method, exemplifies this strategy.

The FAVOR+ Mechanism

Standard softmax attention can be interpreted through the lens of kernel methods, where the attention weights represent an inner product in a high-dimensional feature space. The challenge lies in explicitly computing these features efficiently. FAVOR+ provides a technique using positive orthogonal random features to construct a provably unbiased estimator of the softmax attention kernel.
By applying this approximation, the core computational sequence is rearranged. Instead of calculating Q K^{\top} and then multiplying by V, the operation is transformed into a sequence of linear matrix multiplications: (\phi(Q)^{\top} \phi(K)) V, where \phi represents the random feature map. This change allows the matrix multiplication order to be switched from (N \times N) \times (N \times D) to (N \times k) \times (k \times D), where k is the random feature dimension, typically set much smaller than the sequence length N.

Complexity and Accuracy

The Performer achieves linear space and time complexity, scaling as \mathcal{O}(N D k). Crucially, it achieves this linear scaling without relying on heuristic priors such as sparsity or low-rankness, providing a high-fidelity approximation of the original full-rank softmax attention. This architectural compatibility allows the Performer to be directly integrated into existing Transformer frameworks with strong theoretical guarantees, showcasing competitive results across diverse tasks from text to protein sequencing.

3.2 Convolutional and Hybrid Architectures

Modern Convolutional Networks (ConvNeXt, FNet)

While often overshadowed by the Transformer, Convolutional Neural Networks (CNNs) have experienced a revival by incorporating architectural principles derived from the success of Transformers. Models like ConvNeXt adopt techniques such as layer normalization and depth-wise/point-wise convolutions, drawing inspiration from Vision Transformers (ViT) and Swin Transformers.
Modern CNNs benefit from an inherent efficiency advantage: they leverage locality and utilize highly optimized, mature GPU kernels. The effective computational efficiency of these models, particularly ConvFirst, demonstrates a narrower gap between ideal arithmetic complexity (MACs) and actual inference latency on specialized hardware compared to older designs (e.g., EfficientNet). ConvFirst, using block-fusion kernels, achieved computational efficiencies ranging from 47% to 55%, significantly minimizing the performance difference between theoretical and realized speed.

Architectures Replacing Self-Attention with MLP/Pooling (MLP-Mixer)

The MLP-Mixer architecture represents a maximalist approach to simplicity, removing both self-attention and convolutions in favor of stacking Multi-Layer Perceptrons (MLPs). The token mixing layer handles spatial interaction by applying an MLP across the sequence dimension, while the channel-mixing layer applies an MLP across the feature dimension.
The key design distinction lies in the token mixing: it is **input-agnostic**, meaning the mixing weights are identical for all spatial positions, contrasting sharply with the context-dependent nature of self-attention. This structural simplicity guarantees linear complexity and high efficiency. Furthermore, parameter sharing is applied across spatial or feature dimensions within each MLP block, which aids memory efficiency and prevents parameter explosion, demonstrating that highly competitive results in vision tasks (e.g., ImageNet) can be achieved using only foundational MLP components, provided they are supported by large-scale training and modern optimization.
3.3 Modular and Conditional Computation

Mixture-of-Experts (MoE) Models

Mixture-of-Experts (MoE) models offer a paradigm shift by addressing the need for massive parameter scaling while keeping the training and inference compute costs low. This is achieved through **conditional computation** and sparse activation.

Mechanism and Scaling

In MoE layers, tokens are routed by a learnable gating network to a small, predefined subset of experts, typically Feed-Forward Networks (FFNs). The most common

configuration involves routing each token to the top-2 experts, as introduced in foundational works like GShard.

The primary advantage of MoE is the decoupling of model capacity from computational cost. The total number of parameters (P_{total}) can be scaled into the trillions by adding more experts, thereby increasing capacity and potentially model quality. Yet, because only a constant number of experts (e.g., two) are activated per token, the FLOPs required for training and inference scale linearly (\mathcal{O}(D)) relative to a standard dense Transformer of the same hidden dimension D. This sparse activation enables significantly more compute-efficient pre-training compared to dense models of comparable capacity.

Systemic Challenges and Bottlenecks

While theoretically efficient, MoE models face critical system-level challenges in deployment and training:

1. **Communication Overhead:** In distributed training environments, tokens must be sent to their selected expert on potentially distant GPUs, and the resulting computation must be gathered. This "All-to-All" communication operation is highly intensive and often dominates the training time. Empirical evidence shows that this communication overhead can account for up to 75% of the total iteration time.
2. **Load Imbalance and Capacity:** The dynamic routing mechanism frequently fails to achieve perfect load balancing. This leads to imbalanced expert utilization, where a large portion of tokens (e.g., 70%) may be handled by only a few experts. This requires setting an "expert capacity" threshold, defining the maximum number of tokens an expert can process. When capacity is exceeded, tokens are dropped or rerouted, introducing inefficiency. Techniques like EfficientMoE address this through dynamic scheduling, load prediction, and setting different capacities for replicas of highly utilized experts to reduce padding overhead, resulting in substantial training time reductions (e.g., 30% improvement).
3. **Memory Constraints:** The sparse computation benefit of MoE only applies to FLOPs. All parameters, even those not actively used for a given token, must be loaded into GPU memory (VRAM). This leads to extremely high total memory requirements for MoE models compared to dense models, restricting their deployment to systems with massive VRAM capacity.

The true practical bottleneck for MoE models is systemic—network latency and memory capacity—rather than an algorithmic one. Realizing the full computational efficiency of MoE requires breakthroughs in distributed systems, networking (minimizing All-to-All latency), and highly intelligent router design to manage load distribution effectively.

4. Comparative Analysis and Evaluation

The transition to linear-scaling architectures introduces significant trade-offs, making a direct comparison essential to understanding their practical viability. Evaluation must move beyond simple accuracy metrics to focus on true resource efficiency under load.

4.1 Experimental Setup

To accurately gauge the effectiveness of next-generation architectures, evaluation must prioritize benchmarks designed to expose the scalability limits of the Transformer. Standard NLP tasks (e.g., GLUE) are often insufficient as they use truncated context lengths. Key evaluation environments include the Long Range Arena (LRA) and the RULER benchmark, which explicitly tests long-context understanding and reasoning. Baselines should include not only the standard dense Transformer but also established sub-quadratic variants like Sparse Attention to provide a comprehensive context for performance improvement. Models under test should include high-performing representatives from each category, such as Mamba, Performer, and MoE variants like Mixtral or Jamba.

4.2 Performance Metrics

Evaluation requires a dual focus on quality and efficiency.

Quality Metrics

Standard domain-specific metrics are used, such as perplexity (a measure of a model's predictive capability, where lower scores indicate better performance) , F1 scores, and accuracy (e.g., ImageNet classification accuracy). SSMs like Mamba have achieved perplexity levels comparable to Transformers on large corpora like WikiText103.

Efficiency Metrics

The critical efficiency metrics for comparing linear-scaling models are:

- **Computational Complexity:** Theoretical scaling relative to sequence length N (\mathcal{O}(N^2) vs. \mathcal{O}(N)).
- **Throughput (Tokens/Second):** The empirical measure of processing speed, particularly crucial at high N. Mamba, for instance, has achieved up to 5\times higher throughput than Transformers due to its optimized structure.
- **Memory Usage:** The peak GPU memory consumption. This must distinguish between the linear KV cache growth in Transformers versus the constant inference state memory (\mathcal{O}(D)) in Mamba. The high total parameter memory (\mathcal{O}(P_{total})) required for MoE models is also a key consideration.

- **Parameter Efficiency:** Quality achieved relative to the number of active parameters. MoE excels here, using fewer FLOPs than a dense model of equivalent total parameter count.

4.3 Results and Discussion

The architectural shift provides clear benefits in terms of asymptotic complexity, but this often comes with functional trade-offs, especially regarding specific cognitive tasks.

Table 1: Comparative Theoretical and Empirical Scaling of Sequence Architectures

Architecture Type	Example Model	Time Complexity (T)	Inference Memory (M)	Key Efficiency Claim	Primary Limitation
Standard Attention	Transformer	\mathcal{O}(N^2 D)	\mathcal{O}(ND) (KV Cache)	Standard scaling (1x throughput)	Quadratic scaling at high N, high latency
Recurrent/State Space	Mamba/S5	\mathcal{O}(N D)	\mathcal{O}(D) (Constant State)	Up to 5\times higher throughput than Transformers	Deficit in In-Context Learning (ICL) and copying
Linear Attention	Performer (FAVOR+)	\mathcal{O}(N D k)	\ma[span_41](start_span)[span_41](end_span)thcal{O}(ND)	Linear scaling approximation	Approximation error and required feature dimension k
Conditional Compute	MoE	\mathcal{O}(N D) (Per Token)	\mathcal{O}(P_{total}) (Total VRAM)	Faster pre-training	Systemic communication overhead and load imbalance

Trade-offs: Efficiency vs. Representational Fidelity

While SSMs deliver superior efficiency, research indicates they often underperform Transformers on tasks that require highly precise information retention and manipulation, such as few-shot in-context learning (ICL) or copying long, repetitive sequences. This functional gap suggests that the continuous-time state propagation mechanism of SSMs, while excellent for aggregating large amounts of context, sacrifices the ability to perfectly capture the discrete, symbolic relationships inherently represented by the pairwise interaction matrix of the attention mechanism.

Furthermore, a specific failure mode has been identified in Mamba's long-context performance when input lengths significantly exceed the sequence length used during training. Analysis shows that the internal "global channels"—those responsible for maintaining long-context memory—are the primary bottleneck, exhibiting limitations in adaptively extending their receptive fields. This breakdown suggests that the continuous nature of the SSM, while solving recurrence, causes hidden state decay over extremely long timescales, which must be addressed through architectural

modifications. Training-free techniques like LongMamba, which mitigate this decay by identifying and filtering non-critical tokens in the global channels, are necessary to extend the model's operational range.

The Rise of Hybrid Architectures

The observed trade-offs demonstrate that no single linear-scaling architecture is universally superior to the Transformer across all metrics. This realization has driven the development of **hybrid architectures** that strategically combine the strengths of different components.

A prime example is Jamba, which integrates Transformer, Mamba (SSMs), and MoE layers. This functional modularity allows Jamba to leverage:

1. **Mamba Layers:** For highly efficient processing of long-range dependencies and reduced Key-Value (KV) cache memory usage (by an order of magnitude compared to pure Transformer models).
2. **Transformer Layers:** Retained self-attention layers provide the high-fidelity, discrete token interaction necessary for specific tasks like in-context learning.
3. **MoE:** Provides massive capacity scaling during pre-training.

By specializing the computational layers, Jamba achieves state-of-the-art performance on long-context benchmarks like RULER, demonstrating that the optimal design trajectory is moving toward heterogeneous, modular models that leverage the specific efficiency advantages of each non-Transformer component.

5. Future Directions and Open Challenges

Realizing the full potential of these next-generation architectures requires overcoming significant challenges that span modalities, hardware design, and theoretical understanding.

5.1 Bridging the Modality Gap

The initial success of SSMs and linear attention has been heavily concentrated in 1D sequence modeling (language). Extending this efficiency to complex, higher-dimensional modalities presents a challenge. While models like Vision Mamba (Mamba-ND) have been proposed, they often require non-trivial modifications, such as alternating sequence ordering across layers or processing different axes independently.

However, the current architecture of selective SSMs has limitations when applied to multimodal learning (MLLMs). Because Mamba independently applies the selective SSM to each channel, it struggles to effectively model the interactions and dependencies between multiple channels, potentially overlooking crucial contextual information. Developing robust frameworks for cross-modal transfer learning and joint representations is essential to fully utilize the efficiency of SSMs in multimodal data streams (e.g., video, audio, and genomics).

5.2 Hardware Acceleration and Optimization

The empirical throughput of efficient architectures is deeply intertwined with their ability to map efficiently onto modern parallel hardware, specifically GPUs.

SSM Hardware Co-design

Mamba’s achieved efficiency relies on avoiding the materialization of expanded states in memory-intensive layers and employing kernel fusion and the parallel scan algorithm. However, the core operation of SSMs—solving differential equations through continuous integration—is structurally non-standard compared to the dense matrix multiplication (MM) operations that current GPUs (like those optimized with tensor cores) are overwhelmingly designed for.

To fully unlock the potential of SSMs, specialized hardware accelerators are necessary. For instance, the EpochCore accelerator, designed specifically for SSMs using systolic arrays, has demonstrated substantial gains, achieving up to 2000x improvement in performance on LRA datasets compared to general-purpose GPUs. This evidence strongly suggests that hardware specialization is a necessary condition for maximizing the realized speed gains of SSMs.

MoE System Optimization

For Mixture-of-Experts models, the central bottleneck is network communication. Overcoming the documented 75% iteration time cost attributed to All-to-All communication requires system-level solutions beyond algorithmic improvements. Continued research into efficient scheduling methods (like EfficientMoE) and hardware solutions that accelerate network throughput and reduce memory movement, such as dynamic quantization techniques for LLMs, is crucial for improving MoE's practical viability and energy efficiency.

5.3 Theoretical Understanding

While the empirical success of linear-scaling models is evident, a deeper theoretical foundation is needed. The representational power of SSMs and kernel methods requires formal analysis. Specifically, researchers must determine if these approaches are true universal sequence approximators in the same sense as the dense Transformer.

The empirical finding that Transformers maintain an edge in discrete, symbolic tasks like ICL and copying highlights a gap in the theoretical understanding of continuous-state models. Future work must aim to develop formal guarantees for how to embed high-fidelity discrete processing capacity into continuous state-space mechanisms like Mamba, ensuring the model retains specific contextual details rather than relying solely on global feature aggregation.

5.4 Democratising Large-Scale Learning

The massive resource requirements of contemporary LLMs are a significant barrier to entry, limiting access to large-scale AI research to a few well-funded organizations. The shift towards \mathcal{O}(N) architectures is inherently democratizing.

Architectures like Mamba, with their constant inference memory footprint, allow researchers with limited compute resources to deploy and test models on contexts orders of magnitude longer than previously possible. Complementary innovations in training optimization, such as memory-saving inter-operator parallelism (MPress), further reduce the memory consumption of deep learning models by nearly 2\times. By prioritizing resource efficiency alongside performance, these next-generation designs make sophisticated, large-scale sequence modeling accessible to a broader scientific community, fostering wider innovation and research participation.

6. Conclusion

6.1 Summary of Findings

The dominance of the Transformer architecture, built upon the foundation of quadratic (\mathcal{O}(N^2)) self-attention, is fundamentally unsustainable in the face of ever-growing data and context length demands. This survey confirms that the architectural paradigm shift to linear-scaling models is both necessary and actively underway.

The most promising avenues for achieving \mathcal{O}(N) complexity fall into three categories:

1. **State Space Models (SSMs) like Mamba:** Offering linear time complexity and, critically, constant memory usage during inference through the Selective SSM mechanism, leading to high throughput gains. Mamba-type models excel at long-context aggregation but currently show empirical weaknesses in high-fidelity discrete tasks like ICL.
2. **Linear Attention Methods (e.g., Performer):** Providing theoretically sound, provably accurate approximations of the softmax attention kernel using techniques like FAVOR+, achieving linear scaling without abandoning the original attention mechanism's representational goals.
3. **Conditional Computation (MoE):** Enabling massive, capacity-rich models with sparse FLOP utilization per token, constrained primarily by distributed system bottlenecks, specifically network communication latency and global memory capacity.

The analysis of empirical trade-offs concludes that the future optimal architecture will be modular and heterogeneous. Hybrid models such as Jamba, which strategically combine the high-fidelity token interaction of attention layers with the long-context efficiency of Mamba layers and the massive capacity of MoE, represent the state-of-the-art in resource-aware performance.

6.2 Final Outlook

The trajectory of machine learning architecture design is irrevocably shifting towards models that prioritize scalability, resource efficiency, and deployability alongside raw performance. The full realization of the efficiency gains promised by SSMs and MoE models hinges not only on continued algorithmic innovation but also on critical breakthroughs in systems engineering—specifically, hardware co-design (e.g., specialized ASICs for SSMs) and the optimization of distributed communication for sparse models. By embracing \mathcal{O}(N) scaling, the research community can transition from building resource-hungry models accessible only to a few, to constructing the next generation of truly scalable, sustainable, and democratized artificial intelligence systems.

References

1. Transformer Foundations & Surveys

1. Transformers in Vision: A Survey - arXiv
   https://arxiv.org/pdf/2101.01169
2. Survey: Transformer-based Models in Data Modality Conversion - arXiv
   https://arxiv.org/html/2408.04723v1

---

2. Transformer Efficiency & Parameter Sharing

3. Maximizing Efficiency: Parameter Sharing | by Tharun Sivamani - Cubed
   https://blog.cubed.run/maximizing-efficiency-parameter-sharing-0c285c8602c7
4. Parameter-efficient Multi-task Fine-tuning for Transformers via Shared Hypernetworks
   https://aclanthology.org/2021.acl-long.47/
5. Scaling Graph Transformers: A Comparative Study of Sparse and Dense Attention - arXiv
   https://arxiv.org/html/2508.17175v1

---

3. Beyond Attention & Sub-Quadratic Architectures

6. Attention Mechanism Complexity Analysis | by Mridul Rao - Medium
   https://medium.com/@mridulrao674385/attention-mechanism-complexity-analysis-7314063459b1
7. Rethinking Attention with Performers - arXiv [2009.14794]
   https://arxiv.org/abs/2009.14794
8. Rethinking Attention with Performers (Paper Explained) - YouTube
   https://www.youtube.com/watch?v=xJrKIPwVwGM
9. Beyond Attention: Breaking the Limits of Transformer Context Length with Recurrent Memory
   https://ojs.aaai.org/index.php/AAAI/article/view/29722
10. The End of Transformers? On Challenging Attention and the Rise of Sub-Quadratic Architectures - arXiv [2510.05364]
    https://arxiv.org/html/2510.05364v1

---

4. State Space Models (Mamba & SSMs)

11. S5: Simplified State Space Layers for Efficient Sequence Modeling - Medium
    https://medium.com/@kdk199604/s5-simplified-state-space-layers-for-efficient-sequence-modeling-61ef5f3386ac
12. Mamba: Linear-Time Sequence Modeling with Selective State Spaces - arXiv [2312.00752]

https://arxiv.org/html/2312.00752v2
13. Mamba-ND: Selective State Space Modeling for Multi-Dimensional Data - arXiv
https://arxiv.org/html/2402.05892v1
14. What Is A Mamba Model? | IBM
https://www.ibm.com/think/topics/mamba-model
15. Mamba Explained - The Gradient
https://thegradient.pub/mamba-explained/
16. Mamba (deep learning architecture) - Wikipedia
https://en.wikipedia.org/wiki/Mamba_(deep_learning_architecture)
17. [D] So, Mamba vs. Transformers... is the hype real? : r/MachineLearning - Reddit
https://www.reddit.com/r/MachineLearning/comments/190q1vb/d_so_mamba_vs_transformers_is_the_hype_real/
18. From S4 to Mamba: A Comprehensive Survey on Structured ... - arXiv [2503.18970]
https://arxiv.org/abs/2503.18970
19. LongMamba: Enhancing Mamba's Long Context Capabilities via Training-Free Receptive Field Enlargement - arXiv
https://arxiv.org/html/2504.16053v1
20. A Comprehensive Survey on Mamba: Architectures, Challenges, and Opportunities
https://www.computer.org/csdl/magazine/co/2025/08/11104194/28MaWDg8y0o
21. Characterizing the Behavior of Training Mamba-based State Space Models on GPUs - arXiv
https://arxiv.org/html/2508.17679v1

---

5. Alternative Architectures (MLPs & ConvNets)

22. MLP-Mixer: Breaking the Mold of Vision Architectures | by Dong-Keon Kim | Medium
https://medium.com/@kdk199604/mlp-mixer-breaking-the-mold-of-vision-architectures-bef20d46fce7
23. Rethinking Token-Mixing MLP for MLP-based Vision Backbone - BMVA Archive
https://www.bmva-archive.org.uk/bmvc/2021/assets/papers/0296.pdf
24. (PDF) Enhancing ConvNeXt for efficient small-size image classification - ResearchGate
https://www.researchgate.net/publication/397672676_Enhancing_ConvNeXt_for_efficient_small-size_image_classification
25. On the Efficiency of Convolutional Neural Networks - arXiv
https://arxiv.org/pdf/2404.03617

---

6. Mixture-of-Experts (MoE)

26. What is mixture of experts? | IBM
https://www.ibm.com/think/topics/mamba-model
27. Mixture of Experts Explained - Hugging Face
https://huggingface.co/blog/moe
28. Can someone explain what a Mixture-of-Experts model really is? : r/LocalLLaMA - Reddit
https://www.reddit.com/r/LocalLLaMA/comments/1oqttg0/can_someone_explain_what_a_mixtureofexperts_model/
29. EfficientMoE: Optimizing Mixture-of-Experts Model Training With Adaptive Load Balance
https://ieeexplore.ieee.org/document/10876795/
30. Jamba: Redefining Long-Context AI Performance | by Srujananjali - Medium (Covers an MoE/Mamba Hybrid)
https://medium.com/@srujananjali888/jamba-redefining-long-context-ai-performance-f477306c17e1

---

7. Hardware, Acceleration & Evaluation

31. Ultimate Guide to LLM Evaluation Metrics for AI Optimization - Lamatic.ai Labs
https://blog.lamatic.ai/guides/llm-evaluation-metrics/
32. LLM Evaluation Metrics: The Ultimate LLM Evaluation Guide - Confident AI
https://www.confident-ai.com/blog/llm-evaluation-metrics-everything-you-need-for-llm-evaluation
33. Cobra: Extending Mamba to Multi-Modal Large Language Model for Efficient Inference
https://ojs.aaai.org/index.php/AAAI/article/view/33131/35286
34. Systolic Array-based Accelerator for Structured State-Space Models - arXiv [2507.21394]
https://arxiv.org/abs/2507.21394
35. MPress: Democratizing Billion-Scale Model Training on Multi-GPU Servers via Memory-Saving Inter-Operator Parallelism - ResearchGate
https://www.researchgate.net/publication/369522440_MPress_Democratizing_Billion-Scale_Model_Training_on_Multi-GPU_Servers_via_Memory-Saving_Inter-Operator_Parallelism
36. Hardware Acceleration of LLMs: A comprehensive survey and comparison - arXiv
https://arxiv.org/html/2409.03384v1