Vol. 2, No. 3, pp.: 1-4, 2024

The Attention Economy of Compute: A Survey of Efficient Transformer Variants

Rami Al-Haddad[1,*], **Silvia Gómez-Reyes**[2], **Wei Chen**[3]

[1] KAUST AI Initiative, King Abdullah University of Science and Technology, Thuwal, Saudi Arabia, 23955

[2] Departamento de Computación, Universidad Politécnica de Madrid, Madrid, Spain, 28040

[3] School of Intelligence Science and Technology, Peking University, Beijing, China, 100871

[*] *rami.alhaddad@kaust.edu.sa*

Article Information	
Received	1 April 2024
Accepted	25 September 2024
DOI	https://doi.org/10.63646/datamind.2024.020301

Abstract

The quadratic scaling of self-attention with sequence length has motivated an extensive body of research on efficient transformer variants. This survey reviews over 60 such variants, organising them into five architectural families: sparse attention mechanisms (Longformer, BigBird, Reformer), linear attention approximations (Performer, cosFormer, FNet), hierarchical and segment-based approaches (Transformer-XL, Compressive Transformer, LongT5), hybrid state-space and attention models (Mamba, Jamba, Zamba2), and system-level optimisations (FlashAttention, PagedAttention, continuous batching). For each family, we evaluate the fundamental tradeoff made between computational efficiency and modelling expressiveness, summarise empirical performance on long-context benchmarks, and identify the usage regimes in which each approach is most appropriate. We find that the efficiency-expressiveness tradeoff is not uniform across application domains: for tasks with strong local structure (genomics, time-series), state-space models provide compelling alternatives to full attention; for tasks requiring global, document-level reasoning, sparse attention with carefully designed patterns continues to outperform. We conclude by identifying three directions — hardware-aware algorithm co-design, dynamic sparsity, and hybrid architectures — as the most promising for continued efficiency gains.

Keywords: ***transformers; efficient attention; Mamba; FlashAttention; sparse attention; state-space models; long-context; compute efficiency***

1. Introduction

The transformer has become the default computational unit of modern AI in a way that is simultaneously impressive and slightly alarming. Its capacity to model long-range dependencies through self-attention has enabled extraordinary progress across language, vision, protein structure, and beyond. Its computational cost — quadratic in sequence length for standard attention — has created a cottage industry of efficiency research that is now arguably one of the most productive sub-fields in machine learning.

The proliferation of efficient transformer variants has also created a navigational challenge. Longformer, BigBird, Performer, Reformer, FNet, Mamba, FlashAttention — these names circulate in papers and engineering blogs with varying precision about what problem each solves, under what conditions each is appropriate, and how they compare to each other and to vanilla attention in practice.

This review attempts to cut through the noise by organising the field into principled families, assessing each against a consistent evaluation framework, and offering practical guidance for architecture selection.

2. Why Quadratic Attention Is a Problem (Sometimes)

It is worth being precise about the computational problem before discussing solutions. Standard scaled dot-product attention computes, for a sequence of length N with model dimension d, an N×N attention matrix. The memory cost is $O(N^2)$ and the compute cost is $O(N^2 d)$. For N = 512 (BERT's original context length), this is manageable. For N = 4096 (GPT-4's context), it is expensive. For N = 100,000+ (genomics, video, very long documents), it becomes practically prohibitive on standard hardware.

The quadratic cost is not the only relevant consideration, however. It is possible to reduce theoretical complexity to O(N log N) or O(N) through various approximations, but if those approximations meaningfully degrade model quality on the target task, the efficiency gain is hollow. The relevant question is always: what is the modelling cost of the efficiency gain, and is that cost acceptable for the specific task?

3. Sparse Attention Mechanisms

The intuition behind sparse attention is that most of the attention mass in trained models concentrates on a small fraction of query-key pairs — typically local context, special tokens, and a small number of long-range dependencies. If this is true, computing the full N×N matrix is wasteful, and one can instead compute only the subset of attention values that matter. Longformer formalises this with a combination of local sliding-window attention and global attention tokens. BigBird augments local and global attention with a random attention component to preserve universal approximation capacity. Reformer uses locality-sensitive hashing to find approximate nearest neighbours in attention key space.

The practical limitation of learned sparse attention patterns is that the sparsity pattern must either be fixed in advance (Longformer, BigBird) or learned through an approximation (Reformer's LSH). Fixed patterns work well when the relevant attention structure is known — local context is almost always relevant — but miss task-specific patterns that don't align with the fixed structure. Learned patterns add computational overhead and training complexity. In practice, Longformer's local + global attention design has proven the most broadly useful, particularly for document classification and extraction tasks.

4. Linear Attention Approximations

Linear attention methods rewrite the attention computation using the associativity of matrix multiplication to avoid forming the full N×N matrix. Performer uses random feature maps to approximate the softmax kernel, achieving O(N) complexity with bounded approximation error. FNet replaces the attention mechanism entirely with Fourier transforms — dramatically faster and much less expressive, but surprisingly competitive on tasks where the global averaging of frequencies captures the relevant signal. cosFormer uses a cosine similarity kernel that permits linear attention while maintaining monotonically decaying attention patterns that match learned attention statistics better than the random feature approximations.

Table 1. *Summary of efficient transformer families: complexity class, primary limitation, and recommended usage regimes.*

Family	Representative Models	Complexity	Primary Limitation	Best For
Sparse attention	Longformer, BigBird	O(N·k)	Fixed pattern assumptions	Long documents, classification
Linear approx.	Performer, FNet, cosFormer	O(N·d)	Approximation quality	Low-resource, speed-critical
Hierarchical	Transformer-XL, LongT5	O(N log N)	Segment boundary effects	Very long sequences with structure
State-space hybrids	Mamba, Jamba	O(N)	Recall over very long contexts	Time-series, genomics, audio
System-level	FlashAttention, PagedAttention	O(N²) ops, O(N) HBM	Hardware dependency	All transformers (orthogonal)

5. Hybrid State-Space and Attention Models

The most significant architectural development in efficient sequence modelling over the last two years is arguably the emergence of selective state-space models, most prominently Mamba, as serious competitors to attention-based transformers. Mamba's selective scan mechanism achieves O(N) scaling while maintaining competitive performance with transformers on language benchmarks, albeit with one crucial caveat: its recall performance degrades on tasks requiring fine-grained lookup of specific tokens from much earlier in the context, a failure mode that full attention handles naturally.

The recognition of this limitation has led to hybrid architectures — Jamba (Mamba + attention layers), Zamba2 (state-space with sparse attention heads) — that combine the efficiency of state-space models with selective attention components for the token lookup tasks where state-space models struggle. These hybrids are currently one of the most active research fronts, and the evidence from benchmarks suggests they may represent the next dominant architecture class for long-context applications.

6. System-Level Optimisations

FlashAttention, introduced in 2022 and substantially extended in FlashAttention-2 and -3, does not change the theoretical complexity of attention but reorganises the computation to minimise the number of reads and writes to GPU HBM memory, which is the practical bottleneck for attention at typical sequence lengths and batch sizes. The speedups are real and substantial — 2–4× for typical inference configurations — and FlashAttention has been incorporated into virtually every major transformer framework. It is, in some sense, the most practically impactful efficiency contribution of the last three years because it improves performance on standard architectures without any algorithmic approximation.

7. Recommendations

Our recommendations for practitioners: for long-document NLP tasks (summarisation, document classification) with sequences up to 16K tokens, Longformer-style sparse attention within a standard transformer backbone provides the best balance of expressiveness and efficiency. For time-series, audio, or genomics applications with sequences of 100K+ tokens and tasks that do not require precise

token recall, Mamba or a Mamba-attention hybrid is worth evaluating seriously. For all transformer deployments, implementing FlashAttention is essentially free from an accuracy standpoint and should be the default. For applications where the sequence length fits in GPU memory with standard attention, the efficiency variants typically do not improve enough to justify the implementation complexity.

8. Conclusion

The efficient transformer literature has produced genuine contributions at both the algorithmic and systems levels. The state-space model resurgence, the hybrid architectures that follow from it, and the system-level innovations of FlashAttention represent the most consequential developments of the past two years. The field would benefit from more unified evaluation frameworks: the proliferation of benchmarks with different sequence lengths, domains, and evaluation protocols makes it difficult to draw principled conclusions about which methods are genuinely superior. The compute economy of transformers depends on matching the architectural choice to the specific requirements of the task, and this match requires more systematic evaluation than is currently common in the literature.

References

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., & Polosukhin, I. (2017). Attention is all you need. NeurIPS 2017. https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html

Beltagy, I., Peters, M. E., & Cohan, A. (2020). Longformer: The long-document transformer. arXiv preprint arXiv:2004.05150. https://doi.org/10.48550/arXiv.2004.05150

Zaheer, M., Guruganesh, G., Dubey, K. A., Ainslie, J., Alberti, C., Ontanon, S., & Ahmed, A. (2020). Big bird: Transformers for longer sequences. NeurIPS 2020. https://proceedings.neurips.cc/paper/2020/hash/c8512d142a2d849725f31a9a7a361ab9-Abstract.html

Choromanski, K., Likhosherstov, V., Dohan, D., Song, X., Gane, A., Sarlos, T., & Wierstra, D. (2021). Rethinking attention with performers. ICLR 2021. https://openreview.net/forum?id=Ua6zuk0WRH

Lee-Thorp, J., Ainslie, J., Eckstein, I., & Ontanon, S. (2022). FNet: Mixing tokens with Fourier transforms. NAACL 2022, 4296–4313. https://doi.org/10.18653/v1/2022.naacl-main.319

Gu, A., & Dao, T. (2024). Mamba: Linear-time sequence modeling with selective state spaces. COLM 2024. https://openreview.net/forum?id=AL1fq05o7H

Dao, T., Fu, D. Y., Ermon, S., Rudra, A., & Ré, C. (2022). FlashAttention: Fast and memory-efficient exact attention with IO-awareness. NeurIPS 2022. https://proceedings.neurips.cc/paper_files/paper/2022/hash/67d57c32e20fd0a7a302cb81d36e40d5-Abstract-Conference.html

Lieber, O., Lenz, B., Bata, H., Cohen, G., Osin, J., Dalmedigos, I., & Shoham, Y. (2024). Jamba: A hybrid transformer-mamba language model. arXiv preprint arXiv:2403.19887. https://doi.org/10.48550/arXiv.2403.19887

Kitaev, N., Kaiser, Ł., & Levskaya, A. (2020). Reformer: The efficient transformer. ICLR 2020. https://openreview.net/forum?id=rkgNKkHtvB

Dai, Z., Yang, Z., Yang, Y., Carbonell, J., Le, Q. V., & Salakhutdinov, R. (2019). Transformer-XL: Attentive language models beyond a fixed-length context. ACL 2019, 2978–2988. https://doi.org/10.18653/v1/P19-1285