SLICED-LINEAR ATTENTION: BEYOND THE FINITE-FEATURE BOTTLENECK OF LINEAR ATTENTION

Anonymous authors
Paper under double-blind review

ABSTRACT

To address the quadratic bottleneck of Softmax attention, many efficient attention mechanisms rely on finite-dimensional summaries of the context and, in the causal setting, on fixed-dimensional recurrent states. We introduce Sliced-Linear attention, which multiplies a standard Linear-attention kernel by a Laplace interaction between learned one-dimensional query and key coordinates. This removes the finite-rank constraint of finite-feature Linear attention while retaining exact quasi-linear computation through sorting and scans. We show that, for any prescribed finite key set, a single $R = 1$ Sliced-Linear head can approximate arbitrarily sharp associative retrieval, whereas finite-feature Linear attention requires $R = \Omega(n)$ to maintain fixed relative error as memory size grows. In the causal setting, exact retrieval of n arbitrary d_v-dimensional values by a continuous recurrent model requires state dimension $m \geq nd_v$. Controlled MQAR experiments show the corresponding qualitative capacity separation. On Augmented LRA and in 124M-parameter causal language modeling, Sliced-Linear consistently improves over matched Linear attention, with larger gains after continuation to 16k and 32k context. With $R = 32$, it outperforms Linear attention with $R = 128$ across all tested text settings. Measured forward-and-backward runtime scales near-linearly with sequence length.

1 INTRODUCTION

Self-attention is a central component of the Transformer architecture (Vaswani et al., 2017), but computing all pairwise token interactions requires quadratic time in the sequence length. Modern implementations such as FlashAttention substantially improve memory usage and wall-clock efficiency (Dao et al., 2022; Dao, 2023), while retaining this quadratic computational scaling.

Linear attention instead replaces the attention kernel by a finite-dimensional factorization, typically $K(q, k) = \phi(q)^\top \phi(k)$ with $\phi(q) \in \mathbb{R}^R$ (Katharopoulos et al., 2020; Choromanski et al., 2022). This yields linear complexity in sequence length, but constrains the resulting interaction matrix to rank at most R. In the causal setting, the same factorization summarizes the prefix through a fixed-dimensional recurrent state. More recent recurrent architectures replace these simple accumulations with learned input-dependent updates, while preserving a state dimension independent of context length (Yang et al., 2024; 2025; Kimi Team et al., 2025). These capacity constraints have motivated alternatives with richer or growing representations of the prefix (Arora et al., 2025; Guo et al., 2026; Pan et al., 2025).

This motivates efficient global interactions that do not rely on either a fixed finite-dimensional factorization or a fixed-size recurrent state. We introduce **Sliced-Linear attention** (Section 3), which augments a standard Linear-attention kernel with a Laplace interaction between learned scalar query and key coordinates:

$$K_{\mathrm{SL}}(q, k) = e^{-\lambda |p_q(q) - p_k(k)|}\, \phi(q)^\top \phi(k).$$

This simple modification removes the finite-rank constraint: even with a single linear feature, Sliced-Linear attention can realize full-rank interaction matrices. At the same time, the one-dimensional Laplace factor admits exact evaluation through sorting and scans, yielding quasi-linear algorithms for both bidirectional and causal attention without materializing the full $n \times n$ interaction matrix.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Section 4 studies the resulting capacity difference through associative retrieval. In the bidirectional setting, for any prescribed finite key set, a single $R = 1$ Sliced-Linear head can approximate arbitrarily sharp associative retrieval, whereas a finite-feature Linear-attention head requires $R = \Omega(n)$ to achieve a fixed relative approximation error for an n-item retrieval problem. In the causal setting, the same $R = 1$ Sliced-Linear construction provides arbitrarily sharp retrieval from any finite memory prefix. By contrast, we show that a continuous model that compresses n arbitrary d_v-dimensional values into a recurrent state of dimension m and retrieves them exactly must satisfy $m \geq nd_v$. Complementary finite contextual interpolation and universality results are given in the appendices.

We evaluate Sliced-Linear in Section 5. Controlled MQAR experiments show a strong associative-retrieval advantage over matched Linear attention and Gated Linear Attention (GLA) (Yang et al., 2024). On Augmented Long Range Arena and 124M-parameter causal language modeling, Sliced-Linear with $R = 32$ outperforms Linear attention with $R = 128$ across all tested text settings while using four times fewer random features. The gap increases after continuation to 16k and 32k context, and paired evaluations show that Sliced-Linear benefits more from additional history. Softmax remains stronger in absolute modeling quality, while Sliced-Linear provides a substantially better quality–capacity trade-off than the matched finite-feature Linear baseline.

2 Background and Related Work

We consider normalized kernel attention with queries $q_i \in \mathbb{R}^{d_h}$, keys $k_j \in \mathbb{R}^{d_h}$, and values $v_j \in \mathbb{R}^{d_v}$:

$$y_i = \frac{\sum_{j \in \mathcal{C}_i} K(q_i, k_j) v_j}{\sum_{j \in \mathcal{C}_i} K(q_i, k_j)}, \tag{2.1}$$

where $\mathcal{C}_i = \{1, \ldots, n\}$ for bidirectional attention and $\mathcal{C}_i = \{1, \ldots, i\}$ in the causal setting.

Softmax attention. Standard scaled dot-product attention (Vaswani et al., 2017) uses

$$K_{\mathrm{sm}}(q_i, k_j) = \exp\left(\frac{q_i^\top k_j}{\sqrt{d_h}}\right). \tag{2.2}$$

Without further structure or approximation, evaluating all query–key interactions requires quadratic time in the sequence length. Theoretical properties of self-attention and related normalizations have also been studied through dynamical-systems, measure-theoretic, and regularity viewpoints (Geshkovski et al., 2025; Sander et al., 2022; Castin et al., 2024).

Finite-rank Linear attention. Kernel-based Linear attention instead uses a finite-dimensional factorization

$$K_{\mathrm{lin}}(q_i, k_j) = \phi(q_i)^\top \phi(k_j), \qquad \phi : \mathbb{R}^{d_h} \to \mathbb{R}^R. \tag{2.3}$$

This includes Linear Transformers (Katharopoulos et al., 2020) and random-feature methods such as FAVOR+ (Choromanski et al., 2022). If $\Phi_Q, \Phi_K \in \mathbb{R}^{n \times R}$ collect the query and key features, then

$$A_{\mathrm{lin}} = \Phi_Q \Phi_K^\top, \qquad \mathrm{rank}(A_{\mathrm{lin}}) \leq R. \tag{2.4}$$

The same factorization permits linear-time evaluation by first computing $S = \sum_{j=1}^n \phi(k_j) v_j^\top$, $z = \sum_{j=1}^n \phi(k_j)$, and then $y_i = \phi(q_i)^\top S / (\phi(q_i)^\top z)$. In the causal setting, these statistics satisfy

$$S_i = S_{i-1} + \phi(k_i) v_i^\top, \qquad z_i = z_{i-1} + \phi(k_i). \tag{2.5}$$

Thus the complete prefix is represented by the fixed-dimensional state (S_i, z_i), which also admits a fast-weight associative-memory interpretation (Schlag et al., 2021).

Learned recurrent extensions. In the causal setting, recent linear-time architectures replace these fixed kernel statistics by learned input-dependent recurrent states. Examples include Gated Linear Attention (Yang et al., 2024), DeltaNet and its gated variants (Yang et al., 2025), and Kimi Delta Attention (Kimi Team et al., 2025). Abstractly,

$$h_i = F_\theta(h_{i-1}, x_i), \qquad y_i = G_\theta(h_i, x_i), \tag{2.6}$$

where h_i has dimension independent of the sequence length. Such models do not correspond to a fixed finite-rank attention matrix, but they still represent the complete prefix by a fixed-dimensional state. Gated Slot Attention (Zhang et al., 2024) specifically targets the recall limitations of linear recurrent models by combining gated bounded-memory updates with a Softmax readout.

Growing-state alternatives. Other approaches allow the retained state to grow with context length. Log-Linear Attention (Guo et al., 2026) maintains a logarithmically growing collection of states, while Sparse State Expansion (Pan et al., 2025) increases capacity through sparse partitions.

Sorting-based one-dimensional attention. Sliced ReLU attention (Boufadene & Vialard, 2026) also uses learned one-dimensional query and key projections to obtain exact quasi-linear attention through sorting. Its pairwise interaction, based on a ReLU kernel, depends only on these projected scalar coordinates. Sliced-Linear instead uses the sortable one-dimensional Laplace interaction to modulate an arbitrary finite-feature Linear-attention kernel, thereby retaining the higher-dimensional interaction $\phi(q)^\top \phi(k)$ between queries and keys while preserving exact quasi-linear computation.

3 Sliced-Linear Attention

Sliced-Linear augments a finite-feature Linear-attention kernel with a learned one-dimensional Laplace interaction. We first describe the kernel and its rank properties, then derive exact quasi-linear algorithms for bidirectional and causal attention.

3.1 Sliced-Linear kernel

Let $p_q, p_k : \mathbb{R}^{d_h} \to \mathbb{R}$, where p_q and p_k are learned scalar projections and $s_i = p_q(q_i), t_j = p_k(k_j) \in \mathbb{R}$. We define

$$K_{\mathrm{SL}}(q_i, k_j) = e^{-\lambda|s_i - t_j|}\phi(q_i)^\top \phi(k_j) = e^{-\lambda|s_i - t_j|} K_{\mathrm{lin}}(q_i, k_j), \qquad \lambda \geq 0. \tag{3.1}$$

At $\lambda = 0$, Sliced-Linear reduces to the underlying Linear attention kernel. For $\lambda > 0$, the scale λ can be absorbed into the scalar projections. We keep it explicit to include Linear attention as the $\lambda = 0$ boundary case.

At the level of unnormalized interaction matrices, $A_{\mathrm{SL}} = L \odot A_{\mathrm{lin}}$, where $L_{ij} = e^{-\lambda|s_i - t_j|}$. Unlike A_{lin}, A_{SL} is not constrained to rank at most R. For $R = 1$ and nonzero feature coefficients, $A_{\mathrm{SL}} = \mathrm{diag}(u) L \,\mathrm{diag}(v)$, with $u_i = \phi(q_i)$ and $v_j = \phi(k_j)$. For $\lambda > 0$, the Laplace kernel is strictly positive definite. Hence, if $s_i = t_i$ are pairwise distinct, L has rank n, and therefore $\mathrm{rank}(A_{\mathrm{SL}}) = n$. Row normalization preserves this rank whenever it is well defined. Thus Sliced-Linear can realize full-rank attention interactions even with $R = 1$.

If the scalar coordinates are token positions, the Laplace factor reduces to exponential temporal decay, as in RetNet (Sun et al., 2023). Here the coordinates are instead learned from content.

The scalar projections add two d_h-dimensional vectors per head, or $2Hd_h = 2d_{\mathrm{model}}$ parameters per layer. Experimental parameterization details are given in Appendix G.

3.2 Efficient computation

Bidirectional attention. The one-dimensional Laplace factor admits the exact decomposition (Hofmeyr, 2021)

$$\sum_{j=1}^{n} e^{-\lambda|s_i - t_j|} a_j = e^{-\lambda s_i} \sum_{t_j \leq s_i} e^{\lambda t_j} a_j + e^{\lambda s_i} \sum_{t_j > s_i} e^{-\lambda t_j} a_j. \tag{3.2}$$

After sorting the scalar key and query coordinates, the two sums are computed by one forward and one reverse scan. Using $a_j = \phi(k_j) v_j^\top$ gives the numerator statistics, while $a_j = \phi(k_j)$ gives the normalization statistics. The resulting complexity is

$$O(n \log n + nRd_v)\,, \tag{3.3}$$

without forming the $n \times n$ interaction matrix. Stable scan recurrences are described in Appendix G.

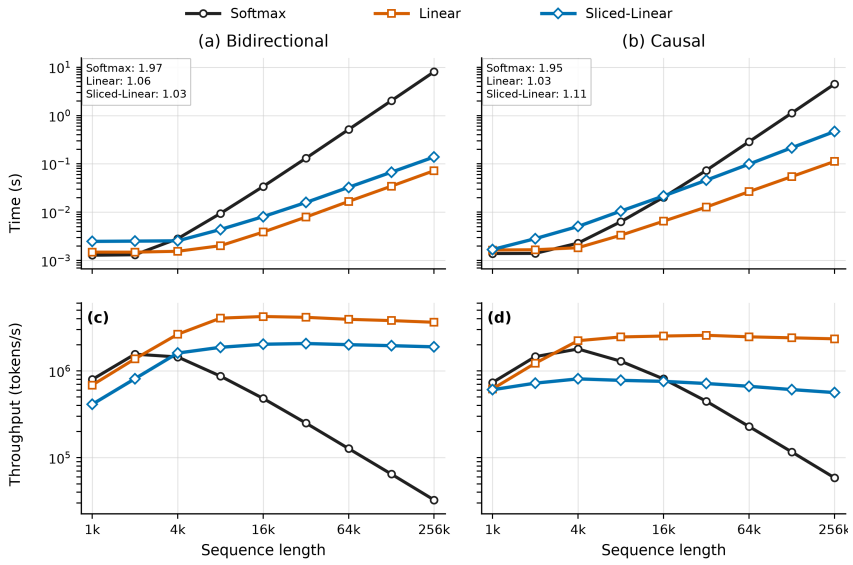

Figure 1: **Runtime scaling of Sliced-Linear attention.** Forward-and-backward wall-clock time (top) and throughput (bottom) for bidirectional (left) and causal (right) attention. Fitted Sliced-Linear scaling exponents are $\alpha = 1.03$ and $\alpha = 1.11$, respectively. Benchmark details are given in Appendix G.

Causal attention. The temporal constraint $j \leq i$ prevents a direct global sort by the scalar coordinates. Instead, we apply a divide-and-conquer decomposition along the temporal axis. If a contiguous interval $I = I_L \sqcup I_R$ is split into consecutive halves, then

$$\mathcal{C}(I) := \{(i,j) \in I^2 : j \leq i\} = \mathcal{C}(I_L) \sqcup \mathcal{C}(I_R) \sqcup (I_R \times I_L). \tag{3.4}$$

The diagonal terms are treated recursively. In the cross term $I_R \times I_L$, every key precedes every query, so causality is automatic and the interactions can be evaluated by the bidirectional sorted scans described above. The recursion terminates at blocks of size B, which are evaluated densely.

For leaf size B, the resulting full-sequence complexity is

$$O\left(nB(R + d_v) + nRd_v \log \frac{n}{B} + n \log^2 \frac{n}{B}\right). \tag{3.5}$$

The three terms correspond respectively to dense leaf blocks, cross-range feature contractions, and sorting. Hence, for fixed B, R, and d_v, the cost is quasi-linear in n. Dense pairwise computation is confined to the leaf blocks, while cross-range interactions are evaluated through sorted scans. Sorting, memory, and implementation details are given in Appendix G.

Implementation. We use numerically rescaled FP32 statistics and a chunked scan formulation for the cross-range interactions. The causal implementation considered here targets full-sequence training and evaluation. Incremental autoregressive decoding is not considered in this work. Further implementation details are given in Appendix G. Code is provided in the supplementary material.

4 Associative Retrieval: Finite-Rank and Fixed-State Bottlenecks

We study the capacity of Sliced-Linear attention through associative retrieval. In the bidirectional setting, this yields a quantitative separation from finite-feature Linear attention: for any prescribed finite key set, a single $R = 1$ Sliced-Linear head supports arbitrarily sharp associative retrieval, whereas Linear attention requires its feature dimension to grow linearly with memory size. In the causal setting, the corresponding $R = 1$ construction provides arbitrarily sharp retrieval from a finite memory prefix, while the same task exposes a more general obstruction for models that compress

the complete prefix into a fixed-dimensional recurrent state. Complementary finite interpolation and contextual universality results are given in Appendices B and C. These are adequacy results rather than separations from Linear or Softmax attention.

4.1 Associative retrieval and finite-rank separation

Associative retrieval provides a classical view of attention as content-addressable memory (Ramsauer et al., 2021). It has also become a central diagnostic for the memory capacity of efficient sequence models, notably through the MQAR task (Arora et al., 2023; 2025; Zhang et al., 2024). Here, we consider a simple version that isolates the attention operator itself. Let n key–value pairs $(x_j, v_j) \in \mathbb{R}^d \times \mathbb{R}^{d_v}$, $j = 1, \dots, n$, with pairwise distinct keys $x_1, \dots, x_n$. Given the same keys in an arbitrary order, the task is to retrieve their associated values, $x_{\sigma(i)} \mapsto v_{\sigma(i)}$ for $\sigma \in S_n$.

Taking $q_i = x_{\sigma(i)}$, $k_j = x_j$, and letting $V \in \mathbb{R}^{n \times d_v}$ contain the values row-wise, exact retrieval for arbitrary V is equivalent to the normalized attention matrix being the permutation matrix

$$A(Q, K) = \Pi_\sigma, \qquad (\Pi_\sigma)_{ij} = \mathbf{1}_{\{j=\sigma(i)\}}. \tag{4.1}$$

Thus the task isolates the ability of the attention operator to select independent memory locations.

Proposition 4.1 (Associative-retrieval separation)**.** *Let $p : \mathbb{R}^d \to \mathbb{R}$ be a fixed nonzero linear form, and let $x_1, \dots, x_n \in \mathbb{R}^d$ have pairwise distinct projected coordinates $p(x_i)$. Set*

$$\delta := \min_{i \neq j} |p(x_i) - p(x_j)| > 0.$$

· ***Sliced-Linear.*** *For every $\sigma \in S_n$ and $\tau > 0$, the normalized $R = 1$ Sliced-Linear head with $\phi \equiv 1$*

$$(A^\sigma_\tau)_{ij} = \frac{e^{-\tau |p(x_{\sigma(i)}) - p(x_j)|}}{\sum_{k=1}^n e^{-\tau |p(x_{\sigma(i)}) - p(x_k)|}} \tag{4.2}$$

satisfies

$$\frac{\|A^\sigma_\tau - \Pi_\sigma\|_F}{\|\Pi_\sigma\|_F} \leq \frac{4e^{-\tau\delta}}{1 + e^{-\tau\delta}}. \tag{4.3}$$

In particular, $A^\sigma_\tau \to \Pi_\sigma$ exponentially fast as $\tau \to \infty$.

· ***Linear attention.*** *Every well-defined normalized Linear-attention matrix with feature dimension R has rank at most R, and therefore*

$$\frac{\|A_{\mathrm{lin}} - \Pi_\sigma\|_F^2}{\|\Pi_\sigma\|_F^2} \geq 1 - \frac{R}{n}. \tag{4.4}$$

Consequently,

$$\frac{\|A_{\mathrm{lin}} - \Pi_\sigma\|_F}{\|\Pi_\sigma\|_F} \leq \varepsilon \qquad \Longrightarrow \qquad R \geq (1 - \varepsilon^2)n. \tag{4.5}$$

In particular, exact retrieval for arbitrary values requires $R \geq n$.

The non-collision condition is generic: for any fixed nonzero p, the n-tuples with $p(x_i) = p(x_j)$ for some $i \neq j$ form a finite union of codimension-one hyperplanes and hence have Lebesgue measure zero. The projection can also be fixed uniformly over any prescribed finite key set $\mathcal{V} \subset \mathbb{R}^d$. A generic p is injective on $\mathcal{V}$, so

$$\delta_{\mathcal{V}} := \min_{x \neq x' \in \mathcal{V}} |p(x) - p(x')| > 0.$$

Consequently, for any target error $\varepsilon > 0$, one sufficiently large τ gives a single $R = 1$ head satisfying (4.3) for every memory of distinct keys from $\mathcal{V}$ and every retrieval permutation. Thus a single scalar head can provide arbitrarily sharp routing without increasing the feature dimension with memory size, whereas Linear attention requires $R = \Omega(n)$ for fixed relative retrieval error. The comparison concerns the attention operator itself. A residual term depending only on the query cannot supply the arbitrary associated value and therefore does not change the lower bound. The proof and comparison with Softmax attention are given in Appendix A.1.

4.2 Causal associative retrieval and fixed-state bottlenecks

The rank argument above applies directly to the bidirectional attention operator. In the causal setting, associative retrieval exposes a more general bottleneck whenever the prefix is compressed into a fixed-dimensional recurrent state. This covers causal Linear attention and the learned recurrent mechanisms discussed in Section 2, all of which fit (2.6).

Fix distinct retrieval keys $x_1, \dots, x_n$, and let the associated values $V = (v_1, \dots, v_n) \in \mathbb{R}^{n \times d_v}$ be arbitrary. For a recurrent model of the form (2.6), let

$$H_n(V) \in \mathbb{R}^m$$

denote the state retained after processing a memory prefix encoding these associations, with the retrieval keys held fixed, and assume that H_n is continuous in V. For each i, let $G_i : \mathbb{R}^m \to \mathbb{R}^{d_v}$ denote the readout obtained when the model is queried with x_i.

Proposition 4.2 (Fixed-state retrieval bottleneck)**.** *Suppose that, for every $V \in \mathbb{R}^{n \times d_v}$,*

$$G_i(H_n(V)) = v_i, \qquad i = 1, \dots, n. \tag{4.6}$$

Then

$$m \geq n d_v. \tag{4.7}$$

Consequently, a continuous recurrent model with state dimension independent of context length cannot exactly retrieve an unbounded number of arbitrary real-valued key–value associations.

Indeed, exact retrieval implies that H_n is injective: two memories producing the same retained state would give the same answer to every query. Since $H_n : \mathbb{R}^{nd_v} \to \mathbb{R}^m$ is continuous, invariance of domain then gives $m \geq nd_v$. A complete proof is given in Appendix A.2. This is complementary to the finite-precision lower bound of Arora et al. (2025), who show that solving associative recall requires $\Omega(n)$ bits of recurrent state. Related fixed-state limitations have also been established for copying tasks (Jelassi et al., 2024).

Proposition 4.2 applies to continuous fixed-state mechanisms of the form discussed in Section 2, including Gated Linear Attention, Gated Slot Attention, gated DeltaNet, and Kimi Delta Attention. The bound concerns exact retrieval of arbitrary real-valued memories.

For standard causal Linear attention, the obstruction is explicit. With fixed keys, the value-dependent state is $S_n(V) = \Phi_K^\top V \in \mathbb{R}^{R \times d_v}$, while the normalization state depends only on the keys. Corollary A.3 in Appendix A shows that this map has a kernel of dimension at least $(n-R)d_v$. In particular, exact retrieval of arbitrary values again requires $R \geq n$.

By contrast, causal Sliced-Linear does not compress the complete prefix into a fixed-dimensional state. Appendix A.1 gives a fixed $R = 1$ causal construction whose retrieval weights converge to the correct memory selector as the Laplace scale increases. Hence the fixed-dimensional state obstruction of Proposition 4.2 does not apply.

5 Experiments

We evaluate Sliced-Linear attention in three settings. MQAR probes associative-retrieval capacity in a controlled synthetic task. Augmented Long Range Arena and causal language modeling test the method on sequence and text modeling, including continuation to 16k and 32k context. Linear attention is the primary matched baseline. Softmax provides an absolute-quality reference, and Gated Linear Attention (GLA) (Yang et al., 2024) provides a learned recurrent baseline for MQAR and causal language modeling.

Experimental control. Linear and Sliced-Linear use the same FAVOR+-style positive orthogonal random features (Choromanski et al., 2022), with the same antithetic construction and fixed random projections. They differ only through the learned scalar query/key projections and Laplace interaction of Sliced-Linear. For these matched comparisons, all remaining architecture and training choices are identical within each setting. Softmax uses its standard attention parameterization. In causal language modeling, GLA uses its native recurrent parameterization while sharing the same non-attention backbone and training protocol. MQAR is treated separately as a capacity experiment, with its search and model-selection protocol described in Appendix D. Further implementation details are given in Appendix G.

5.1 Multi-Query Associative Recall

We use the MQAR task of Arora et al. (2023) to test associative-retrieval capacity, following their width-scaling experiment. The vocabulary contains 8,192 tokens. For sequence length $N \in \{64, 128, 256\}$, each example contains $D = N/4$ stored key–value associations and corresponding retrieval queries.

All models use two causal layers with one head per layer. We vary the model width d, which is therefore also the head width. We compare Softmax, GLA, Linear, and Sliced-Linear attention, with $R \in \{32, 128\}$ for Linear and Sliced-Linear. For each model and setting, we use the same hyperparameter search budget and select the run with highest validation accuracy. Full search and model-selection details are given in Appendix D.

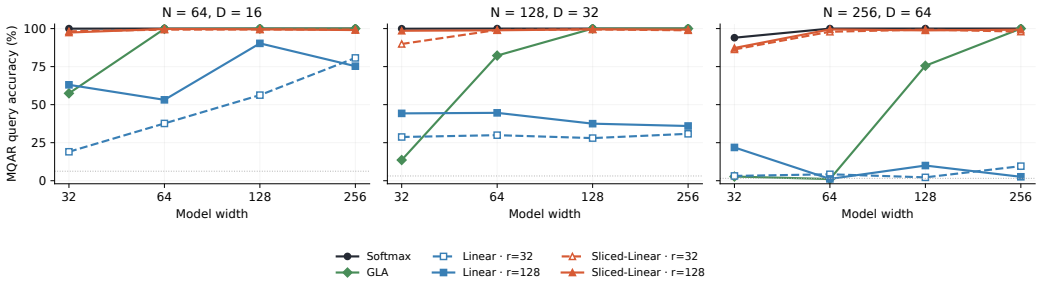

Figure 2: **Associative recall on MQAR.** Query accuracy as a function of model width for sequence length N and $D = N/4$ stored associations. Linear and Sliced-Linear use $R \in \{32, 128\}$. Each point reports independent-test accuracy for the run selected by validation accuracy. Dotted lines mark the $1/D$ baseline corresponding to uniform choice among the D stored values.

Figure 2 compares width scaling as the number of stored associations increases. Softmax reaches near-perfect recall once the width is sufficient. Sliced-Linear reaches high accuracy at fixed feature dimension: with $R = 32$ and width 64, accuracy is at least 97.9% in all three settings. In the hardest setting, $N = 256$, Linear attention remains low throughout the width range. GLA improves strongly with width and approaches perfect recall at width 256.

These results are consistent with the retrieval analysis of Section 4. Sliced-Linear sustains high recall as the number of stored associations increases without increasing the feature dimension R.

5.2 Augmented Long Range Arena

We evaluate Softmax, Linear, and Sliced-Linear attention on five Long Range Arena tasks (Tay et al., 2020), with $R \in \{32, 128\}$ for Linear and Sliced-Linear. We do not include GLA because these experiments use bidirectional encoders, whereas the GLA baseline considered here is causal. We follow the data-efficiency interventions of Miralles-González et al. (2025) and refer to this setting as *Augmented LRA*. These results are therefore not directly comparable to standard LRA results. The two byte-level text tasks, IMDb and AAN Retrieval, additionally use a masked-byte denoising objective during training. Within each task, architecture and training choices are matched across attention mechanisms. Complete protocols are given in Appendix E.

Results. The gains vary substantially across tasks (Table 1). On Pathfinder, Linear and Sliced-Linear perform similarly. The difference is larger on CIFAR-10 and largest on the two byte-level text tasks. At $R = 128$, Sliced-Linear improves accuracy over Linear by 17.14 points on IMDb and 7.88 points on Retrieval. Moreover, Sliced-Linear with $R = 32$ outperforms Linear with $R = 128$ on both text tasks, despite using four times fewer random features.

The same separation appears on the auxiliary masked-byte objective (Table 2). Sliced-Linear achieves lower denoising NLL and higher masked-byte accuracy at both feature dimensions. With $R = 32$, it again outperforms Linear with $R = 128$ on both datasets. Thus the two text tasks show the same ordering for both classification and token-level prediction.

Table 1: **Augmented LRA classification accuracy (%).** Mean accuracy over three random seeds, with standard deviation. Bold denotes the best overall result and underlining the best efficient-attention result in each column. Experimental protocols are given in Appendix E.

Model	ListOps	CIFAR-10	Pathfinder	IMDb	Retrieval	Avg.
Softmax	**62.95** $\pm$ 0.4	**84.98** $\pm$ 0.6	**92.25** $\pm$ 2.7	**93.16** $\pm$ 0.6	**89.71** $\pm$ 0.5	**84.61**
Linear ($R = 32$)	42.80 $\pm$ 0.7	59.29 $\pm$ 1.3	83.69 $\pm$ 0.8	65.04 $\pm$ 0.2	79.92 $\pm$ 0.7	66.15
Linear ($R = 128$)	42.55 $\pm$ 0.8	62.63 $\pm$ 0.8	83.40 $\pm$ 1.0	73.14 $\pm$ 3.1	77.17 $\pm$ 2.3	67.78
Sliced-Linear ($R = 32$)	43.20 $\pm$ 0.4	61.99 $\pm$ 0.8	<u>83.91</u> $\pm$ 0.9	86.42 $\pm$ 1.3	84.07 $\pm$ 2.5	71.92
Sliced-Linear ($R = 128$)	<u>44.30</u> $\pm$ 1.1	<u>67.04</u> $\pm$ 0.9	83.43 $\pm$ 0.8	<u>90.28</u> $\pm$ 1.2	<u>85.05</u> $\pm$ 1.7	<u>74.02</u>

Table 2: **Denoising performance on the LRA text tasks.** Metrics are evaluated on masked byte positions at the final training epoch. Lower NLL and higher accuracy are better.

	IMDb		Retrieval	
Model	MLM NLL $\downarrow$	MLM acc. (%) $\uparrow$	MLM NLL $\downarrow$	MLM acc. (%) $\uparrow$
Softmax	**0.795** $\pm$ 0.02	**76.80** $\pm$ 0.6	**0.536** $\pm$ 0.01	**84.76** $\pm$ 0.3
Linear ($R = 32$)	2.992 $\pm$ 0.07	19.22 $\pm$ 1.5	3.074 $\pm$ 0.02	16.58 $\pm$ 0.1
Linear ($R = 128$)	2.007 $\pm$ 0.12	43.06 $\pm$ 3.0	2.584 $\pm$ 0.05	27.35 $\pm$ 1.3
Sliced-Linear ($R = 32$)	1.454 $\pm$ 0.13	57.81 $\pm$ 0.3	1.257 $\pm$ 0.11	64.35 $\pm$ 0.1
Sliced-Linear ($R = 128$)	<u>1.194</u> $\pm$ 0.04	<u>65.25</u> $\pm$ 1.2	<u>1.095</u> $\pm$ 0.03	<u>68.87</u> $\pm$ 0.6

5.3 Causal Language Modeling and Long Context

We next test whether the text-modeling advantage transfers to causal language modeling and whether it grows when longer histories are available. We pretrain approximately 124M-parameter decoder-only models for 10B FineWeb tokens at context length 1k, then continue the corresponding checkpoints separately on PG-19 at 16k and 32k context for 3.9B additional tokens. All models use the same 12-layer, $d_{\text{model}} = 768$ backbone and $d_{\text{ff}} = 2048$ feed-forward blocks. Softmax, Linear, and Sliced-Linear use 12 heads, while GLA uses the authors' default 4-head parameterization (Yang et al., 2024). The tokenizer, optimization schedule, training data, and token budgets are shared across models. Softmax, Linear, and Sliced-Linear use RoPE, extended to the longer contexts by position interpolation (Chen et al., 2023). GLA does not use RoPE and retains its native recurrent parameterization. Full details are given in Appendix F.

Table 3: **Causal language modeling performance.** Perplexity is reported with NLL in parentheses. FineWeb models are pretrained for 10B tokens at 1k context; PG-19 models are continued from the corresponding checkpoints at 16k or 32k context. Lower is better.

Model	FineWeb 1k $\downarrow$	PG-19 16k $\downarrow$	PG-19 32k $\downarrow$
Softmax	**19.76** (2.984)	**17.62** (2.869)	**17.65** (2.871)
GLA	21.25 (3.056)	18.47 (2.916)	19.34 (2.962)
Linear ($R = 32$)	22.97 (3.134)	21.82 (3.083)	25.13 (3.224)
Linear ($R = 128$)	22.58 (3.117)	20.64 (3.027)	22.87 (3.130)
Sliced-Linear ($R = 32$)	22.22 (3.101)	19.67 (2.979)	21.30 (3.059)
Sliced-Linear ($R = 128$)	22.04 (3.093)	18.97 (2.943)	19.96 (2.994)

Sliced-Linear improves over matched Linear attention already at 1k context, and the gap widens after long-context continuation (Table 3). At $R = 128$, PG-19 perplexity decreases from 20.64 to 18.97 at 16k and from 22.87 to 19.96 at 32k. GLA achieves the lowest loss among the efficient models, with NLL 2.916 at 16k and 2.962 at 32k. The corresponding differences to Sliced-Linear are 0.027 and 0.032, compared with 0.037 at the FineWeb checkpoint.

Use of additional context. Absolute PG-19 loss partly reflects differences already present in the 1k checkpoints. To isolate the benefit of longer history, we perform two paired evaluations on identical held-out PG-19 windows. First, we keep the final 1,024 target tokens fixed while progressively increasing the visible context. Second, we measure the Sliced-Linear–Linear NLL difference as a function of target position within the full evaluation window.

At every context length above 1k, Sliced-Linear obtains a larger context-induced NLL reduction than matched Linear attention (Figure 3, left). Its advantage is also generally larger at later target positions (right). Thus the long-context gap is not solely inherited from the 1k checkpoints. It increases as additional history is exposed. End-to-end training times and the corresponding quality–runtime comparison are reported in Appendix F.

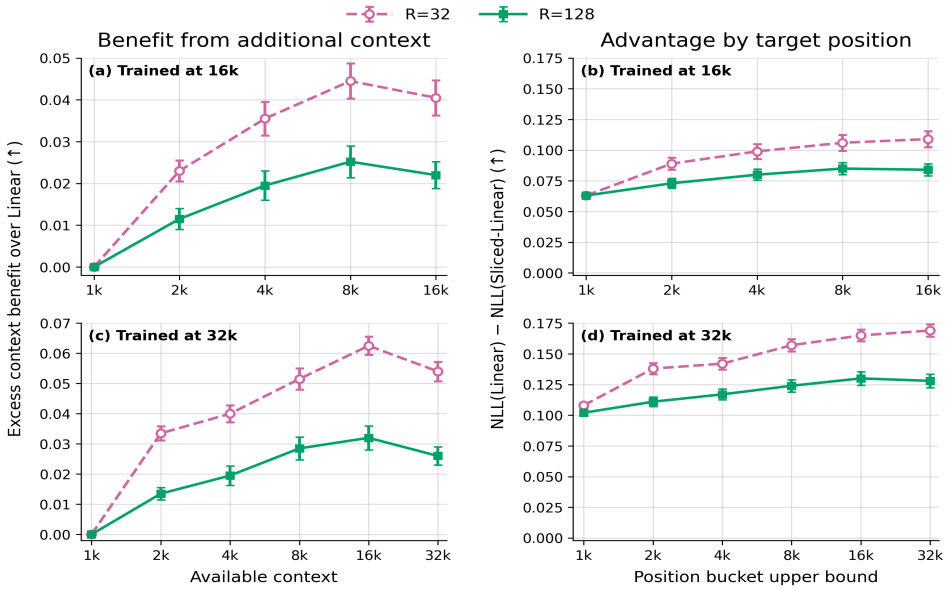

Figure 3: **Effect of additional context on Sliced-Linear and Linear attention. Left:** excess NLL reduction obtained by increasing the visible context from 1k to c, $\big[\mathcal{L}_{\mathrm{SL}}(1\mathrm{k}) - \mathcal{L}_{\mathrm{SL}}(c)\big] - \big[\mathcal{L}_{\mathrm{Lin}}(1\mathrm{k}) - \mathcal{L}_{\mathrm{Lin}}(c)\big]$, evaluated on the same final 1,024 target tokens. **Right:** direct NLL advantage $\mathcal{L}_{\mathrm{Lin}} - \mathcal{L}_{\mathrm{SL}}$ as a function of target position. Rows correspond to models continued at 16k and 32k context; curves show $R \in \{32, 128\}$. Positive values favor Sliced-Linear. Error bars denote 95% book-clustered bootstrap confidence intervals.

6 Conclusion

We introduced Sliced-Linear attention, which augments finite-feature Linear attention with a learned one-dimensional Laplace interaction. This modification removes the finite-rank constraint of Linear attention while preserving exact quasi-linear computation through sorting and scans. For any prescribed finite key set, a single $R = 1$ Sliced-Linear head supports arbitrarily sharp retrieval, whereas finite-feature Linear attention requires $R = \Omega(n)$ to maintain fixed relative error as memory size grows. We also established the related bottleneck for fixed-dimensional causal recurrent states, together with complementary contextual interpolation and universality results.

Across controlled bidirectional and causal experiments, the largest gains over matched Linear attention occur on text modeling. Sliced-Linear with $R = 32$ outperforms Linear attention with $R = 128$ across all tested text settings while using four times fewer random features, and the gap widens after long-context continuation to 16k and 32k tokens. Paired evaluations further show a larger NLL reduction for Sliced-Linear as additional history is exposed. Together, these results show that structured pairwise interactions can increase the capacity of finite-feature attention without returning to quadratic attention. Softmax remains stronger in absolute modeling quality, while scaling Sliced-Linear and improving the efficiency of its causal implementation remain important directions for future work.

AI USE STATEMENT

Generative AI tools, including LLM-based coding assistants, were used to assist with code development and implementation, mathematical derivations and proof checking, experimental design and analysis, literature search, and drafting and editing parts of the manuscript. All AI-assisted mathematical arguments, experimental claims, code, citations, and text were reviewed and verified by the authors. The authors take full responsibility for the final content of this work.

REPRODUCIBILITY STATEMENT

Complete proofs and additional theoretical details are provided in Appendices A–C. Experimental protocols, hyperparameters, evaluation procedures, and implementation details are given in Appendices D–G, and a reference implementation of Sliced-Linear attention and the Transformer architecture is provided in the supplementary material.

REFERENCES

Simran Arora, Sabri Eyuboglu, Aman Timalsina, Isys Johnson, Michael Poli, James Zou, Atri Rudra, and Christopher Ré. Zoology: Measuring and improving recall in efficient language models, 2023. URL https://arxiv.org/abs/2312.04927.

Simran Arora, Sabri Eyuboglu, Michael Zhang, Aman Timalsina, Silas Alberti, Dylan Zinsley, James Zou, Atri Rudra, and Christopher Ré. Simple linear attention language models balance the recall-throughput tradeoff, 2025.

Siwan Boufadene and François-Xavier Vialard. Sliced relu attention: Quasi-linear contextual expressivity via sorting, 2026. URL https://arxiv.org/abs/2512.11411.

Valérie Castin, Pierre Ablin, and Gabriel Peyré. How smooth is attention?, 2024.

Shouyuan Chen, Sherman Wong, Liangjian Chen, and Yuandong Tian. Extending context window of large language models via positional interpolation, 2023. URL https://arxiv.org/abs/2306.15595.

Krzysztof Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Lukasz Kaiser, David Belanger, Lucy Colwell, and Adrian Weller. Rethinking attention with performers, 2022.

Ekin D. Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V. Le. Autoaugment: Learning augmentation policies from data, 2019. URL https://arxiv.org/abs/1805.09501.

Tri Dao. Flashattention-2: Faster attention with better parallelism and work partitioning, 2023.

Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness, 2022. URL https://arxiv.org/abs/2205.14135.

Peter M. Fenwick. A new data structure for cumulative frequency tables. *Software: Practice and Experience*, 24(3):327–336, 1994. doi: 10.1002/spe.4380240306.

Takashi Furuya, Maarten V. de Hoop, and Gabriel Peyré. Transformers are universal in-context learners, 2024.

Borjan Geshkovski, Cyril Letrouit, Yury Polyanskiy, and Philippe Rigollet. A mathematical perspective on transformers, 2025. URL https://arxiv.org/abs/2312.10794.

Borjan Geshkovski, Philippe Rigollet, and Domènec Ruiz-Balet. Measure-to-measure interpolation using transformers. *Foundations of Computational Mathematics*, 2026. doi: 10.1007/s10208-026-09754-w.

Han Guo, Songlin Yang, Tarushii Goel, Eric P. Xing, Tri Dao, and Yoon Kim. Log-linear attention, 2026. URL https://arxiv.org/abs/2506.04761.

David P. Hofmeyr. Fast exact evaluation of univariate kernel sums. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(2):447–458, 2021. doi: 10.1109/TPAMI.2019.2930501.

Samy Jelassi, David Brandfonbrener, Sham M. Kakade, and Eran Malach. Repeat after me: Transformers are better than state space models at copying. In *Proceedings of the 41st International Conference on Machine Learning*, 2024.

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are rnns: Fast autoregressive transformers with linear attention, 2020.

Kimi Team, Yu Zhang, Zongyu Lin, Xingcheng Yao, Jiaxi Hu, Fanqing Meng, Chengyin Liu, Xin Men, Songlin Yang, Zhiyuan Li, Wentao Li, Enzhe Lu, Weizhou Liu, Yanru Chen, Weixin Xu, Longhui Yu, Yejie Wang, Yu Fan, Longguang Zhong, Enming Yuan, Dehao Zhang, Yizhi Zhang, T. Y. Liu, Haiming Wang, Shengjun Fang, Weiran He, Shaowei Liu, Yiwei Li, Jianlin Su, Jiezhong Qiu, Bo Pang, Junjie Yan, Zhejun Jiang, Weixiao Huang, Bohong Yin, Jiacheng You, Chu Wei,

Zhengtao Wang, Chao Hong, Yutian Chen, Guanduo Chen, Yucheng Wang, Huabin Zheng, Feng Wang, Yibo Liu, Mengnan Dong, Zheng Zhang, Siyuan Pan, Wenhao Wu, Yuhao Wu, Longyu Guan, Jiawen Tao, Guohong Fu, Xinran Xu, Yuzhi Wang, Guokun Lai, Yuxin Wu, Xinyu Zhou, Zhilin Yang, and Yulun Du. Kimi linear: An expressive, efficient attention architecture, 2025. URL https://arxiv.org/abs/2510.26692.

Pablo Miralles-González, Javier Huertas-Tato, Alejandro Martín, and David Camacho. On the locality bias and results in the long range arena, 2025.

Yuqi Pan, Yongqi An, Zheng Li, Yuhong Chou, Ruijie Zhu, Xiaohui Wang, Mingxuan Wang, Jinqiao Wang, and Guoqi Li. Scaling linear attention with sparse state expansion, 2025. URL https://arxiv.org/abs/2507.16577.

Guilherme Penedo, Hynek Kydlíček, Loubna Ben allal, Anton Lozhkov, Margaret Mitchell, Colin Raffel, Leandro Von Werra, and Thomas Wolf. The fineweb datasets: Decanting the web for the finest text data at scale, 2024. URL https://arxiv.org/abs/2406.17557.

Jack W Rae, Anna Potapenko, Siddhant M Jayakumar, Chloe Hillier, and Timothy P Lillicrap. Compressive transformers for long-range sequence modelling. *arXiv preprint*, 2019. URL https://arxiv.org/abs/1911.05507.

Hubert Ramsauer, Bernhard Schäfl, Johannes Lehner, Philipp Seidl, Michael Widrich, Thomas Adler, Lukas Gruber, Markus Holzleitner, Milena Pavlović, Geir Kjetil Sandve, Victor Greiff, David Kreil, Michael Kopp, Günter Klambauer, Johannes Brandstetter, and Sepp Hochreiter. Hopfield networks is all you need, 2021. URL https://arxiv.org/abs/2008.02217.

Michael E. Sander, Pierre Ablin, Mathieu Blondel, and Gabriel Peyré. Sinkformers: Transformers with doubly stochastic attention, 2022.

Imanol Schlag, Kazuki Irie, and Jürgen Schmidhuber. Linear transformers are secretly fast weight programmers, 2021. URL https://arxiv.org/abs/2102.11174.

Yutao Sun, Li Dong, Shaohan Huang, Shuming Ma, Yuqing Xia, Jilong Xue, Jianyong Wang, and Furu Wei. Retentive network: A successor to transformer for large language models, 2023. URL https://arxiv.org/abs/2307.08621.

Yi Tay, Mostafa Dehghani, Samira Abnar, Yikang Shen, Dara Bahri, Philip Pham, Jinfeng Rao, Liu Yang, Sebastian Ruder, and Donald Metzler. Long range arena: A benchmark for efficient transformers, 2020.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Songlin Yang, Bailin Wang, Yikang Shen, Rameswar Panda, and Yoon Kim. Gated linear attention transformers with hardware-efficient training, 2024. URL https://arxiv.org/abs/2312.06635.

Songlin Yang, Jan Kautz, and Ali Hatamizadeh. Gated delta networks: Improving mamba2 with delta rule, 2025. URL https://arxiv.org/abs/2412.06464.

Chulhee Yun, Srinadh Bhojanapalli, Ankit Singh Rawat, Sashank J. Reddi, and Sanjiv Kumar. Are transformers universal approximators of sequence-to-sequence functions?, 2020.

Yu Zhang, Songlin Yang, Ruijie Zhu, Yue Zhang, Leyang Cui, Yiqiao Wang, Bolun Wang, Freda Shi, Bailin Wang, Wei Bi, Peng Zhou, and Guohong Fu. Gated slot attention for efficient linear-time sequence modeling, 2024. URL https://arxiv.org/abs/2409.07146.

A Associative Retrieval: Proofs and Additional Results

A.1 Proof of Proposition 4.1

Proof. Let

$$\rho := e^{-\tau\delta}.$$

Fix $\sigma \in S_n$ and a row i. Since the projected key coordinates $p(x_1), \dots, p(x_n)$ are pairwise separated by at least δ, the k-th nearest projected key to $p(x_{\sigma(i)})$ on either side is at distance at least $k\delta$. Hence, writing

$$r_i := \sum_{j \neq \sigma(i)} e^{-\tau |p(x_{\sigma(i)}) - p(x_j)|},$$

we have

$$r_i \leq 2 \sum_{k \geq 1} \rho^k = \frac{2\rho}{1-\rho}.$$

The correct entry of the normalized row is $(1 + r_i)^{-1}$, while the remaining entries have total mass $r_i/(1 + r_i)$. Therefore

$$\|(A^\sigma_\tau)_{i,:} - (\Pi_\sigma)_{i,:}\|_1 = \frac{2r_i}{1 + r_i} \leq \frac{4\rho}{1 + \rho}.$$

Using $\|\cdot\|_2 \leq \|\cdot\|_1$ row-wise gives

$$\frac{\|A^\sigma_\tau - \Pi_\sigma\|_F}{\|\Pi_\sigma\|_F} \leq \frac{4\rho}{1 + \rho},$$

which proves (4.3).

For normalized Linear attention,

$$A_{\text{lin}} = D^{-1}\Phi_Q\Phi_K^\top,$$

so $\operatorname{rank}(A_{\text{lin}}) \leq R$ whenever the normalization is well defined. Exact retrieval for arbitrary values requires $A_{\text{lin}}V = \Pi_\sigma V$ for every V, and hence $A_{\text{lin}} = \Pi_\sigma$. More generally, for $R < n$, since every singular value of Π_σ equals 1, the Eckart–Young theorem gives

$$\inf_{\operatorname{rank}(A) \leq R} \|A - \Pi_\sigma\|_F^2 = n - R.$$

Since $\|\Pi_\sigma\|_F^2 = n$, this yields (4.4), and (4.5) follows immediately. □

For any fixed nonzero p, projected collisions occur only on a finite union of codimension-one hyperplanes in $(\mathbb{R}^d)^n$, and hence on a Lebesgue-null set of key tuples.

Remark A.1 (Uniformity over finite key sets). Let $\mathcal{V} \subset \mathbb{R}^d$ be any finite set of distinct possible keys. A generic linear form p is injective on $\mathcal{V}$, since it suffices to avoid the finite union

$$\bigcup_{\substack{x,x' \in \mathcal{V} \\ x \neq x'}} (x - x')^\perp.$$

Hence

$$\delta_{\mathcal{V}} := \min_{\substack{x,x' \in \mathcal{V} \\ x \neq x'}} |p(x) - p(x')| > 0.$$

After fixing such a p, Proposition 4.1 holds with $\delta \geq \delta_{\mathcal{V}}$ for every memory containing distinct keys from $\mathcal{V}$ and every retrieval permutation. Thus the scalar projection can be fixed uniformly over the entire finite key set rather than chosen separately for each memory.

Causal retrieval. The construction extends directly to the causal setting provided memory and query tokens are distinguishable in their representation. For a finite key set $\mathcal{V}$, fix p as in Remark A.1 and let

$$\Gamma \geq \max_{x\in\mathcal{V}} p(x) - \min_{x\in\mathcal{V}} p(x) + \delta_{\mathcal{V}}.$$

Choose the scalar projections so that memory keys have coordinate $p(x)$, queries have query coordinate $p(x)$, and query tokens, when acting as keys, have coordinate $p(x) + \Gamma$. This can be realized by an affine projection when the token representation contains a memory/query role coordinate.

The key coordinates contributed by query tokens then lie outside the memory range and remain separated from it by at least $\delta_{\mathcal{V}}$. Consequently, a query for x uniquely matches the corresponding memory key, and the same argument as above gives arbitrarily sharp causal retrieval with a fixed $R = 1$ head.

Remark A.2 (Relation to Softmax attention). Softmax attention has the same sharp-retrieval property. Let $x_1, \ldots, x_n$ be pairwise distinct unit vectors, take $q_i = x_{\sigma(i)}$, and define

$$\gamma := 1 - \max_{i\neq j} \langle x_i, x_j \rangle > 0.$$

With inverse temperature τ, every incorrect logit is separated from the correct one by at least $\tau\gamma$. The same row-wise argument therefore gives

$$\frac{\|A^{\sigma}_{\mathrm{sm},\tau} - \Pi_\sigma\|_F}{\|\Pi_\sigma\|_F} \leq 2(n-1)e^{-\tau\gamma}.$$

Thus Softmax and Sliced-Linear both support arbitrarily sharp content-dependent retrieval. Proposition 4.1 separates Sliced-Linear from finite-feature Linear attention, while Proposition 4.2 concerns the distinct bottleneck created by fixed-dimensional causal state compression.

A.2 Proof of Proposition 4.2

Proof. Suppose that

$$H_n(V) = H_n(V')$$

for two memories $V, V' \in \mathbb{R}^{n\times d_v}$. Then, for every $i = 1, \ldots, n$,

$$v_i = G_i(H_n(V)) = G_i(H_n(V')) = v'_i.$$

Hence $V = V'$, so

$$H_n : \mathbb{R}^{nd_v} \to \mathbb{R}^m$$

is continuous and injective.

If $m < nd_v$, compose H_n with the canonical embedding $\iota : \mathbb{R}^m \hookrightarrow \mathbb{R}^{nd_v}$. Then $\iota \circ H_n$ is a continuous injection from $\mathbb{R}^{nd_v}$ into itself whose image is contained in a proper linear subspace. By invariance of domain, its image must be open, a contradiction. □

A.3 Causal Linear attention

For standard causal Linear attention, fixed keys make the loss of information especially explicit. The value-dependent component of the prefix state is

$$S_n(V) = \sum_{j=1}^{n} \phi(x_j) v_j^\top = \Phi_K^\top V,$$

while the normalization state $z_n = \sum_{j=1}^{n} \phi(x_j)$ depends only on the keys.

Corollary A.3 (Linear-attention state)**.** *If $R < n$, the map $V \mapsto S_n(V)$ is not injective. More precisely,*

$$\dim \ker\big(V \mapsto \Phi_K^\top V\big) \geq (n-R)d_v. \tag{A.1}$$

Hence distinct value memories can produce the same recurrent state, and exact retrieval of arbitrary values requires $R \geq n$.

Proof. For fixed keys, $\Phi_K^\top \in \mathbb{R}^{R\times n}$, and therefore

$$\dim\ker\big(V \mapsto \Phi_K^\top V\big) = d_v\big(n - \operatorname{rank}(\Phi_K)\big) \geq (n-R)d_v.$$

If $R < n$, choose $0 \neq c \in \ker(\Phi_K^\top)$ and $0 \neq w \in \mathbb{R}^{d_v}$. For $V' = V + cw^\top$,

$$\Phi_K^\top V' = \Phi_K^\top V.$$

The normalization state z_n is unchanged because the keys are fixed. Since $c \neq 0$, at least one row of V' differs from the corresponding row of V. Thus the two memories produce the same recurrent state while requiring different answers to at least one retrieval query. □

B FINITE CONTEXTUAL INTERPOLATION

We complement the retrieval results of Section 4.1 with a finite contextual interpolation result in the spirit of Yun et al. (2020). Related measure-to-measure interpolation results were developed by Geshkovski et al. (2026). Throughout this section, we restrict to the normalized pure-Laplace subfamily of Sliced-Linear attention,

$$k_\lambda(s,t) = e^{-\lambda|s-t|}, \qquad \lambda > 0,$$

corresponding to $R = 1$ and $\phi \equiv 1$, and allow affine query, key, and value maps.

Proposition B.1 (Finite contextual interpolation)**.** *Let $X^a = (x_1^a, \dots, x_n^a) \in (\mathbb{R}^d)^n$, $a = 1, \dots, p$. Assume that the tokens within each X^a are pairwise distinct and that the empirical measures*

$$\mu_a = \frac{1}{n}\sum_{j=1}^n \delta_{x_j^a}$$

are pairwise distinct. Then, for arbitrary targets $Y^a = (y_1^a, \dots, y_n^a) \in (\mathbb{R}^d)^n$, there exist a normalized single-head Sliced-Linear layer Γ, with residual connection, and a pointwise two-layer ReLU network F of width at most pn such that

$$F\big(\Gamma(X^a)_j\big) = y_j^a, \qquad a = 1, \dots, p, \quad j = 1, \dots, n. \tag{B.1}$$

In particular, one attention head and an MLP width independent of d suffice.

The proof first constructs a single Laplace head that separates the empirical measures through exponentially tilted barycenters. The residual connection then separates all pn contextualized tokens under a scalar projection, after which a one-dimensional ReLU network interpolates the prescribed targets.

For the remainder of this section, set

$$N := pn, \qquad \mu_a := \frac{1}{n}\sum_{j=1}^n \delta_{x_j^a}.$$

B.1 SINGLE-HEAD CONTEXTUAL SEPARATION

We first show that a normalized Laplace head can distinguish any finite family of distinct empirical measures.

Lemma B.2 (Laplace-transform separation)**.** *Let $\mu_1, \dots, \mu_p$ be pairwise distinct finitely supported probability measures on $\mathbb{R}^d$, and define*

$$M_a(u) := \int e^{\lambda\langle u,x\rangle}\, d\mu_a(x), \qquad b_a(u) := \frac{\int x e^{\lambda\langle u,x\rangle}\, d\mu_a(x)}{\int e^{\lambda\langle u,x\rangle}\, d\mu_a(x)}.$$

Then:

1. *there exists $u \in \mathbb{R}^d$ such that $M_1(u), \dots, M_p(u)$ are pairwise distinct;*
2. *there exist $u, v \in \mathbb{R}^d$ such that $\langle v, b_1(u)\rangle, \dots, \langle v, b_p(u)\rangle$ are pairwise distinct.*

Proof. For $a \neq b$, uniqueness of the Laplace transform implies that the real-analytic function $M_a - M_b$ is not identically zero. Hence its zero set has empty interior, and a generic $u \in \mathbb{R}^d$ satisfies $M_a(u) \neq M_b(u)$ for all $a \neq b$, proving the first statement.

For the second, note that

$$b_a(u) = \frac{1}{\lambda} \nabla \log M_a(u).$$

If $b_a \equiv b_b$, then $\log M_a - \log M_b$ is constant. Since $M_a(0) = M_b(0) = 1$, this gives $M_a \equiv M_b$, so that $\mu_a = \mu_b$. Thus $b_a - b_b$ is not identically zero for $a \neq b$. By analyticity, we can choose u such that $b_1(u), \ldots, b_p(u)$ are pairwise distinct. Finally, choosing v outside the finite union of hyperplanes

$$\left\{ v \in \mathbb{R}^d : \langle v, b_a(u) - b_b(u) \rangle = 0 \right\}, \qquad a \neq b$$

proves the result. □

We next show how the distinct sequence-level scalars can be used to separate all tokens across the finite collection of sequences.

Lemma B.3 (Separation by scalar shifts)**.** *Suppose that, for each $a = 1, \ldots, p$, the tokens $x_1^a, \ldots, x_n^a$ are pairwise distinct, and let $\gamma_1, \ldots, \gamma_p \in \mathbb{R}$ be pairwise distinct. Then there exist a linear form $\ell \in (\mathbb{R}^d)^*$, a vector $e \in \mathbb{R}^d$, and $\beta \in \mathbb{R}$, with $\ell(e) = 1$, such that the pn scalars*

$$s_j^a := \ell\big(x_j^a + \beta \gamma_a e\big)$$

are pairwise distinct.

Proof. Choose a nonzero linear form ℓ such that $\ell(x_j^a - x_k^a) \neq 0$ for every a and $j \neq k$. Such a choice exists by avoiding finitely many hyperplanes. Hence, within each sequence, the scalars $\ell(x_j^a)$ are pairwise distinct. Choose e such that $\ell(e) = 1$. Then, for $a \neq b$,

$$\ell(x_j^a) + \beta \gamma_a = \ell(x_k^b) + \beta \gamma_b$$

can hold only if

$$\beta = \frac{\ell(x_k^b) - \ell(x_j^a)}{\gamma_a - \gamma_b}.$$

Choosing β outside this finite set ends the proof. □

Lemma B.4 (Finite ReLU interpolation)**.** *Let $s_1, \ldots, s_N \in \mathbb{R}$ be pairwise distinct and let $r_1, \ldots, r_N \in \mathbb{R}^d$. There exists a two-layer ReLU network $g : \mathbb{R} \to \mathbb{R}^d$ of width at most N such that*

$$g(s_i) = r_i, \qquad i = 1, \ldots, N.$$

Proof. The case $N = 1$ is immediate, so assume $N \geq 2$. After relabeling, assume $s_1 < \cdots < s_N$. A two-layer ReLU network has the form

$$g(s) = W_2 \, \mathrm{ReLU}(W_1 s + b_1) + b_2.$$

Equivalently, with width N it can represent functions of the form

$$g(s) = a + bs + \sum_{i=2}^{N-1} c_i \, \mathrm{ReLU}(s - s_i),$$

since $s = \mathrm{ReLU}(s) - \mathrm{ReLU}(-s)$. Choosing the coefficients to match the successive affine pieces interpolating (s_i, r_i) gives $g(s_i) = r_i$ for all i. □

B.2 Proof of Proposition B.1

Proof. Apply Lemma B.2 and choose u, v such that

$$\gamma_a = \langle v, b_a(u)\rangle, \qquad a = 1, \dots, p,$$

are pairwise distinct. Let ℓ, e, β be given by Lemma B.3.

Choose $c > \max_{a,j}\langle u, x_j^a\rangle$, and consider a normalized Laplace head with constant query score c, key score $\langle u, x\rangle$, and value map

$$V(x) = \beta\langle v, x\rangle e.$$

Since $c > \langle u, x_j^a\rangle$ for all a, j, its output at every position of X^a is

$$\frac{\sum_{j=1}^n e^{-\lambda|c-\langle u,x_j^a\rangle|}\beta\langle v, x_j^a\rangle e}{\sum_{j=1}^n e^{-\lambda|c-\langle u,x_j^a\rangle|}} = \beta\langle v, b_a(u)\rangle e = \beta\gamma_a e.$$

With the residual connection, the resulting attention layer Γ therefore satisfies

$$\Gamma(X^a)_j = x_j^a + \beta\gamma_a e\,.$$

Hence, by Lemma B.3, the pn scalars $s_j^a := \ell\big(\Gamma(X^a)_j\big)$ are pairwise distinct.

By Lemma B.4, there exists a two-layer ReLU network $g : \mathbb{R} \to \mathbb{R}^d$ of width at most $N = pn$ such that

$$g(s_j^a) = y_j^a \qquad \text{for all } a, j.$$

Defining $F(z) := g(\ell(z))$ gives a two-layer ReLU network of the same width, and $F\big(\Gamma(X^a)_j\big) = y_j^a$ for all a, j, proving the result. □

Parameter count. For fixed inputs $X^1, \dots, X^p$, arbitrary targets range over $\mathbb{R}^{pnd}$. If an architecture with W real parameters depends locally Lipschitz-continuously on its parameters, its evaluation map into $\mathbb{R}^{pnd}$ has image of Hausdorff dimension at most W. Consequently, representing every possible target assignment requires $W \geq pnd$. Thus the $O(pnd)$ parameter count of the interpolation construction is optimal up to constants for arbitrary exact targets.

B.3 Attention-only realizations

The pointwise MLP in Proposition B.1 can also be replaced by unnormalized Sliced-Linear attention, with a trade-off between the number of heads per layer and depth.

Proposition B.5 (Attention-only width–depth trade-off)**.** *Under the assumptions of Proposition B.1, augment the token representations with one auxiliary scalar coordinate. For every $1 \leq H \leq N = pn$, the prescribed mappings $X^a \mapsto Y^a$ can be realized by an unnormalized Laplace-kernel Sliced-Linear network consisting of one context layer followed by*

$$L = \left\lceil \frac{N}{H} \right\rceil$$

residual attention layers, each with at most H heads. The construction uses $N + 1$ heads in total across all layers.

Proof. By the first part of Lemma B.2, choose $u \in \mathbb{R}^d$ such that

$$\rho_a := M_a(u) = \frac{1}{n}\sum_{j=1}^n e^{\lambda\langle u, x_j^a\rangle}, \qquad a = 1, \dots, p,$$

are pairwise distinct. Applying Lemma B.3 to $\rho_1, \dots, \rho_p$, choose ℓ and β such that

$$s_j^a := \ell(x_j^a) + \beta\rho_a, \qquad a = 1, \dots, p, \quad j = 1, \dots, n,$$

are pairwise distinct.

Using the auxiliary scalar coordinate, represent each token as

$$\widehat{x}_j^a := \big(x_j^a, \ell(x_j^a)\big) \in \mathbb{R}^{d+1}.$$

Choose $c > \max_{a,j} \langle u, x_j^a \rangle$, and consider an unnormalized Laplace head with constant query score c, key score $\langle u, x \rangle$, and constant value

$$\left(0_d, \frac{\beta e^{\lambda c}}{n}\right) \in \mathbb{R}^{d+1}.$$

At every position of X^a, its output is

$$\left(0_d, \frac{\beta}{n} \sum_{j=1}^{n} e^{\lambda \langle u, x_j^a \rangle}\right) = (0_d, \beta \rho_a)\,.$$

With the residual connection, the resulting tokens are therefore

$$\widetilde{x}_j^a = (x_j^a, s_j^a).$$

Enumerate these N tokens as (x_r, s_r), $r = 1, \ldots, N$, with corresponding targets y_r. Since the s_r are pairwise distinct, the Laplace Gram matrix $K_{rq} := e^{-\lambda |s_r - s_q|}$ is strictly positive definite and hence invertible. Therefore, there exist $\alpha_1, \ldots, \alpha_N \in \mathbb{R}^d$ such that

$$y_r - x_r = \sum_{q=1}^{N} e^{-\lambda |s_r - s_q|} \alpha_q, \qquad r = 1, \ldots, N.$$

Partition $\{1, \ldots, N\}$ into $L = \lceil N/H \rceil$ sets $I_1, \ldots, I_L$, each of cardinality at most H. In layer m, for each $q \in I_m$, use one unnormalized Laplace head with query score s, constant key score s_q, and constant value $(\alpha_q/n, 0)$. Such a head contributes $\left(e^{-\lambda |s - s_q|} \alpha_q, 0\right)$, so that the residual update of layer m is

$$(z, s) \longmapsto \left(z + \sum_{q \in I_m} e^{-\lambda |s - s_q|} \alpha_q,\ s\right).$$

After the L residual layers, the first d coordinates of token r are

$$x_r + \sum_{q=1}^{N} e^{-\lambda |s_r - s_q|} \alpha_q = y_r.$$

A final linear projection discards the auxiliary coordinate. The first layer uses one head, while the remaining layers use N heads in total, which proves the result. □

C CONTEXTUAL UNIVERSALITY

We next formulate contextual expressivity independently of the number of tokens, using the measure-theoretic in-context framework of Furuya et al. (2024). The result is an adequacy statement rather than a separation: the proof only requires uniform aggregation, which is also available to normalized Linear attention.

As in Appendix B, we allow affine query, key, and value maps.

Proposition C.1 (Contextual universality). *Let $\Omega \subset \mathbb{R}^d$ be compact, let $R \geq 1$, and let $\phi : \mathbb{R}^k \to \mathbb{R}^R$ be continuous with $\phi \not\equiv 0$. For every continuous*

$$\Lambda^\star : \mathcal{P}(\Omega) \times \Omega \to \mathbb{R}^{d'}$$

and every $\varepsilon > 0$, there exists a finite-depth Sliced-Linear Transformer $\mathcal{T}$ such that

$$\sup_{(\mu, x) \in \mathcal{P}(\Omega) \times \Omega} \|\mathcal{T}(\mu, x) - \Lambda^\star(\mu, x)\| \leq \varepsilon. \tag{C.1}$$

The construction can use scalar attention heads and at most d' heads per layer, independently of ε and of the context size. In particular, the result already holds for $R = k = 1$ and $\phi \equiv 1$.

We recall the in-context formalism needed for the proof. Let $\Omega \subset \mathbb{R}^d$ be compact and $\mu \in \mathcal{P}(\Omega)$. A normalized multi-head attention layer induces

$$\Gamma_\theta(\mu, x) = x + \sum_{h=1}^{H} W^h \frac{\int S^h(Q^h x, K^h y) V^h y \, d\mu(y)}{\int S^h(Q^h x, K^h z) \, d\mu(z)}. \tag{C.2}$$

For an empirical measure $\mu = \frac{1}{n}\sum_{i=1}^n \delta_{x_i}$, this recovers the corresponding token-based attention layer.

For Sliced-Linear attention,

$$S^{\mathrm{SL}}(q, k) = e^{-\lambda |p_q(q) - p_k(k)|} \phi(q)^\top \phi(k). \tag{C.3}$$

For in-context maps Γ_1, Γ_2, composition is defined by

$$(\Gamma_2 \diamond \Gamma_1)(\mu, x) = \Gamma_2((\Gamma_1(\mu))_{\#}\mu, \Gamma_1(\mu, x)), \tag{C.4}$$

where $\Gamma_1(\mu) : x \mapsto \Gamma_1(\mu, x)$. Pointwise maps are regarded as context-independent in-context maps.

We prove Proposition C.1 by combining a Stone-Weierstrass argument with a sequential construction following Furuya et al. (2024). The only attention-specific ingredient is that normalized Sliced-Linear attention can realize uniform aggregation.

Proof of Proposition C.1. Since $\phi \not\equiv 0$, choose $u_0 \in \mathbb{R}^k$ with $\phi(u_0) \neq 0$. Take the query and key maps constantly equal to u_0 and choose the scalar projections p_q, p_k equal. Then

$$S^{\mathrm{SL}}(u_0, u_0) = \|\phi(u_0)\|^2 > 0,$$

so normalization yields uniform aggregation. In particular, if one value coordinate contains $g(y)$, the head computes

$$M_g(\mu) \coloneqq \int_\Omega g(y) \, d\mu(y).$$

Consider the unital algebra

$$\mathcal{A} \coloneqq \mathrm{alg}\left\{(\mu, x) \mapsto f(x), \quad (\mu, x) \mapsto M_g(\mu) : \ f, g \in \mathcal{C}(\Omega)\right\}.$$

The space $\mathcal{P}(\Omega) \times \Omega$ is compact for the product weak-* $\times$ l_2 topology. The generators of $\mathcal{A}$ are continuous, and $\mathcal{A}$ separates points: continuous functions of x separate distinct tokens, while, if $\mu \neq \nu$, there exists $g \in \mathcal{C}(\Omega)$ such that $M_g(\mu) \neq M_g(\nu)$. Hence, by the Stone-Weierstrass theorem, $\mathcal{A}$ is dense in $C(\mathcal{P}(\Omega) \times \Omega)$ for the uniform norm.

It remains to realize elements of $\mathcal{A}$. Each such element is a finite sum of terms of the form

$$f_n(x) \prod_{t=1}^{T_n} M_{g_{n,t}}(\mu).$$

For a scalar output, represent each token by

$$(x, A, P, G) \in \mathbb{R}^{d+3},$$

where x is retained, A stores the accumulated sum, P the current product, and G is a working coordinate. A pointwise map initializes $P = f_n(x)$ and sets $G = g_{n,t}(x)$. Uniform attention, with its output written into the same coordinate, performs the update

$$G \longmapsto g_{n,t}(x) + M_{g_{n,t}}(\mu).$$

The following pointwise map recomputes $g_{n,t}(x)$ from the retained coordinates, recovers the moment by subtraction, updates $P \mapsto P\, M_{g_{n,t}}(\mu)$, and resets G. After all factors have been processed, it updates $A \mapsto A + P$ and proceeds to the next summand.

Although the full context measure is pushed forward between layers, its projection onto the retained x-coordinates remains μ. Consequently, whenever the working coordinate contains $g(y)$, uniform attention still returns $\int g \, d\mu$. Pointwise MLPs approximate all the continuous maps and multiplications used above uniformly on compact sets. Since the construction involves only finitely many bounded

intermediate operations, their approximation errors can be chosen so that the resulting Transformer approximates any prescribed element of $\mathcal{A}$ arbitrarily well.

For an $\mathbb{R}^{d'}$-valued target, apply the construction coordinatewise with $A, P, G \in \mathbb{R}^{d'}$. At each layer, at most one scalar aggregation head is needed per output coordinate. Thus

$$d_{\text{in}}(\theta_\ell) \le d + 3d', \qquad d_{\text{head}}(\theta_\ell) = 1, \qquad k(\theta_\ell) = k, \qquad H(\theta_\ell) \le d'.$$

Combining this realization with the density of $\mathcal{A}$ proves the result. The feature dimension R, the embedding dimension, and the number of heads are independent of the approximation accuracy and of the number of context tokens. □

D MQAR Experimental Details

Task and architecture. We use the synthetic multi-query associative recall task of Arora et al. (2023). The vocabulary contains 8,192 tokens. For a sequence of length N, we sample $D = N/4$ key–value associations and evaluate prediction accuracy only at the corresponding query positions. We consider $N \in \{64, 128, 256\}$. All models contain two causal sequence-mixing layers, one head per layer, learned absolute positional embeddings, and GELU feed-forward blocks of width $4d$, where d is the model width. Linear and Sliced-Linear are evaluated with feature dimensions $R \in \{32, 128\}$.

Training and model selection. Models are trained for 64,000 updates with batch size 64. For each model–width–sequence-length setting, all architectures receive the same number of candidate runs under a common hyperparameter-search budget. The search includes four learning rates logarithmically spaced between 10^{-4} and 10^{-2}. Checkpoints are evaluated every 4,000 updates on 512 validation sequences. We select the run with highest validation accuracy, using validation loss to break ties, and evaluate the selected run on an independent test set of 3,000 sequences.

E Augmented LRA experimental details

Tasks and preprocessing. We use five Long Range Arena tasks under the augmented protocol described in Section 5.2.

For **ListOps**, expressions are truncated or padded to 2,000 tokens. During training, we parse each expression into a tree and independently permute the children of every operator node at each epoch. Since the ListOps operators are commutative, these permutations preserve the expression value and length. Validation and test examples retain their canonical ordering.

For **CIFAR-10**, the original 32×32 RGB images are augmented before grayscale conversion and sequence flattening. We use random crops with four pixels of padding, horizontal flips with probability 0.5, and the standard CIFAR-10 AutoAugment policy (Cubuk et al., 2019). Images are then converted to grayscale and flattened in row-major order into sequences of length 1,024. No augmentation is applied at validation or test time.

For **PathFinder**, we combine equal numbers of examples from the easy, intermediate, and hard generators, yielding 480,000 training examples. Each binary image is flattened in row-major order into a sequence of length 1,024. Separate validation and test sets are retained for each difficulty. Checkpoint selection uses the hard validation split, and results are reported on the hard test split.

For **IMDb** and **AAN Retrieval**, we use the byte-level representation of the original LRA pipeline and truncate inputs to 4,000 bytes. Training includes an auxiliary denoising objective. At each epoch, 30% of non-padding bytes are selected independently. Two thirds are replaced by a mask token and one third by a uniformly sampled byte token. The model predicts the original byte at the selected positions. The training loss is the unweighted sum of the classification and denoising losses.

Architecture and optimization. All tasks use bidirectional pre-normalized Transformer encoders with GELU feed-forward blocks, masked mean pooling, and one-dimensional rotary position embeddings with base $\theta = 10{,}000$. We do not use QK normalization. Within each task, the encoder architecture and classification head are identical across attention mechanisms. Task-specific configurations are reported in Table 4. Models are optimized with AdamW and gradient clipping.

Table 4: Architecture and training configurations for Augmented LRA. "Length" denotes the maximum sequence length and "Batch" the effective optimization batch size.

	ListOps	CIFAR-10	PathFinder	IMDb	Retrieval
Length	2,000	1,024	1,024	4,000	4,000
Layers	6	10	6	8	8
d_{model}	256	160	128	256	256
Heads	4	4	4	4	4
d_{ff}	512	320	256	512	512
Batch	64	48	128	32	64
Epochs	120	200	67	80	36
Peak LR	10^{-4}	10^{-3}	10^{-3}	10^{-3}	8×10^{-4}
Dropout	0.0	0.0	0.0	0.1	0.1

Model selection and reporting. Each configuration is run with three random seeds. For ListOps and CIFAR-10, we select checkpoints by validation accuracy and report the corresponding test accuracy. For PathFinder, checkpoint selection uses the hard validation split and results are reported on the hard test split. For IMDb, we report accuracy on the fixed 5k validation split. For AAN Retrieval, checkpoints are selected on the clean validation set and evaluated on the test set. Denoising metrics are evaluated at the final training epoch.

F Causal language modeling details

Data and tokenization. We use the RoBERTa byte-level BPE tokenizer. Training sequences are sampled from FineWeb as contiguous 1,024-token windows within individual documents; we do not concatenate unrelated documents to fill a sequence. The same sampled training data and tokenization pipeline are used for all attention mechanisms.

FineWeb pretraining. We train approximately 124M-parameter decoder-only models on 10B FineWeb tokens (Penedo et al., 2024). All models use 12 layers, hidden dimension 768, a SwiGLU feed-forward dimension of 2,048, pre-RMSNorm, tied input/output embeddings, and no dropout. Softmax, Linear, and Sliced-Linear use 12 attention heads. GLA uses the default 4-head parameterization of Yang et al. (2024), implemented with the FLA backend, with key expansion 0.5, value expansion 1.0, gate rank 16, gate-logit normalizer 16, and normalization $\varepsilon = 10^{-5}$. Total parameter counts range from 123.6M to 123.8M across models. The remaining training protocol is shared across architectures. Sequences have length 1,024.

Training lasts 38,147 optimizer updates with a global batch of 262,144 tokens. We use AdamW with $(\beta_1, \beta_2) = (0.9, 0.95)$, weight decay 0.1, gradient clipping at 1, and peak learning rate 6×10^{-4}. The learning rate is warmed up for 2,000 updates and cosine-decayed to 10% of its peak value. Each run uses four NVIDIA H100 GPUs. The Softmax baseline uses PyTorch scaled dot-product attention with the automatically selected fused backend.

PG-19 continuation. Each long-context model is initialized from the corresponding FineWeb checkpoint and continued on PG-19 (Rae et al., 2019). We train separate models at context lengths 16,384 and 32,768. For Softmax, Linear, and Sliced-Linear, RoPE is extended by position interpolation (Chen et al., 2023). GLA retains its native recurrent parameterization without RoPE.

Both context lengths use 30,000 optimizer updates with 131,072 tokens per update, corresponding to 3.93B continuation tokens. We use AdamW with peak learning rate 8×10^{-5}, 1,000 warmup updates, cosine decay to 10% of the peak learning rate, weight decay 0.1, and no dropout. Training uses four NVIDIA H100 GPUs.

Long-context evaluation. We do not perform checkpoint selection during PG-19 continuation: all reported results use the final checkpoint after 30,000 updates. To reduce evaluation variance, we pool held-out windows from the official validation and test splits, sampling 2,048 token-uniform windows in total, comprising 683 validation and 1,365 test windows. No window from either split is used for

training or checkpoint selection. Evaluation is paired across architectures, with every model scored at identical book and token coordinates.

For position-profile evaluation, every token in the complete 16k or 32k window is scored. For nested-context evaluation, the same final 1,024 target tokens are scored while exposing histories of length 1k, 2k, 4k, 8k, 16k, and, for the 32k models, 32k. Confidence intervals in Figure 3 are obtained by bootstrap resampling at the book level.

End-to-end training time. For each PG-19 continuation run, we compute training time per 10^6 tokens from the median logged throughput. All runs use four NVIDIA H100 GPUs. These measurements include the complete training step and are distinct from the standalone single-GPU attention benchmarks of Appendix G.

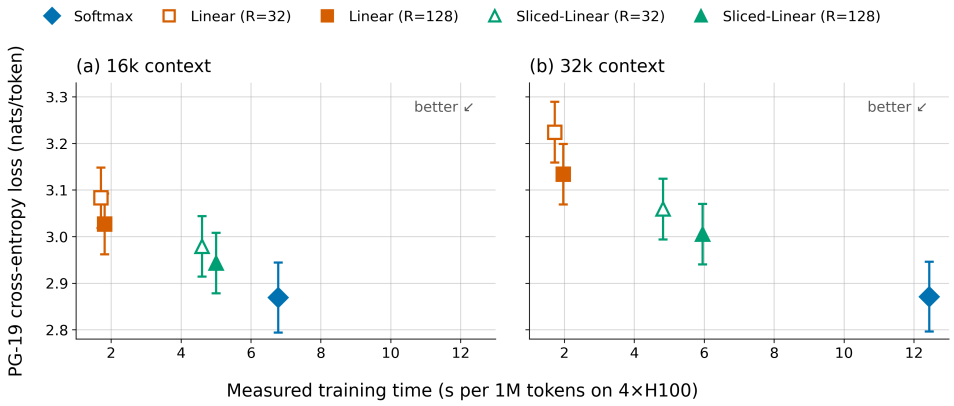

Figure 4: **End-to-end quality–runtime trade-off on PG-19.** Cross-entropy loss versus measured training time per 10^6 tokens on $4\times$H100 GPUs after continuation at 16k (**left**) and 32k (**right**) context. Vertical error bars show 95% book-clustered bootstrap confidence intervals for evaluation loss.

Analytical training compute. We also estimate matrix FLOPs analytically. A multiply–add counts as two FLOPs, and forward-plus-backward compute is approximated as three times the forward cost. For a model with L layers, model width C, intermediate width I, H heads of dimension D, feature dimension R, and vocabulary size V, the estimated cost per token is

$$F_{\text{train}} = 3\left\{L\left[8C^2 + 6CI + F_{\text{feat}} + F_{\text{scalar}} + F_{\text{attn}}\right] + 2CV\right\}. \tag{F.1}$$

For Linear and Sliced-Linear attention, $F_{\text{feat}} = 4HDR$. Sliced-Linear additionally has $F_{\text{scalar}} = 4HD$. The attention-specific terms are

$$F_{\text{attn}} = \begin{cases} 2(N+1)HD, & \text{causal Softmax,} \\ 4HR(D+1), & \text{causal Linear attention,} \\ H[(B+1)(R+D+1) + 8R(D+1)\log_2(N/B)], & \text{causal Sliced-Linear.} \end{cases} \tag{F.2}$$

For Sliced-Linear, B is chosen among powers of two to minimize this analytical cost, giving $B^\star = 256$ for $R = 32$ and $B^\star = 512$ for $R = 128$. We use $L = 12$, $C = 768$, $I = 2048$, $H = 12$, $D = 64$, and $V = 50{,}000$. The resulting estimates are reported in Table 5 and compared with PG-19 NLL in Figure 5.

G Additional Implementation Details

Standalone attention runtime benchmarks. All methods are benchmarked at identical batch size, head dimension, number of heads, and value dimension. Linear and Sliced-Linear attention use $R = 32$ features, and causal Sliced-Linear uses block size $B = 512$. Measurements are obtained on a single NVIDIA H100 GPU. For each sequence length, we perform 25 warm-up iterations followed by 40 seconds of timed execution. Reported wall-clock times and throughputs measure the combined forward and backward pass.

Table 5: **Analytical matrix FLOPs on PG-19.** Estimated training GFLOPs per token. Sliced-Linear uses $B = 256$ for $R = 32$ and $B = 512$ for $R = 128$.

Model	16k	32k
Linear ($R = 32$)	0.748	0.748
Linear ($R = 128$)	0.770	0.770
Sliced-Linear ($R = 32$)	0.799	0.806
Sliced-Linear ($R = 128$)	0.942	0.971
Softmax	1.647	2.553

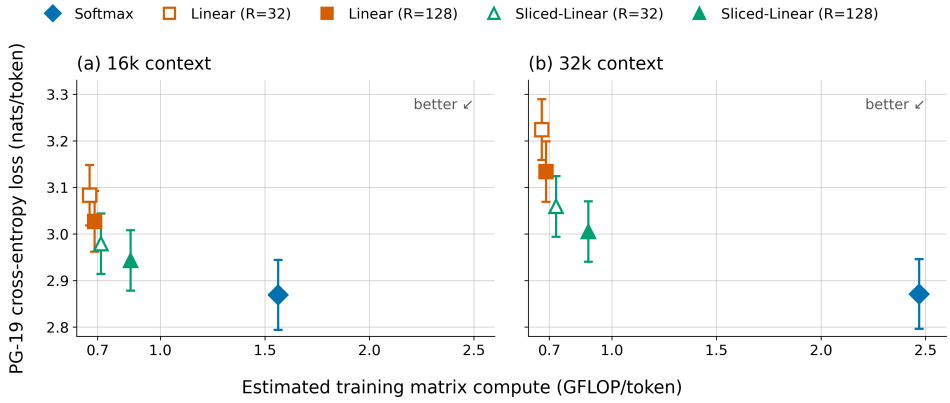

Figure 5: **Analytical quality–compute trade-off on PG-19.** PG-19 cross-entropy loss versus estimated training matrix FLOPs per token after continuation at 16k (**left**) and 32k (**right**) context. The estimate excludes sorting and search comparisons, exponentials, normalization, elementwise operations, memory traffic, communication, optimizer updates, and checkpoint recomputation. Vertical error bars show 95% book-clustered bootstrap confidence intervals for evaluation loss.

Random features. Linear and Sliced-Linear use the same FAVOR+-style positive orthogonal random features (Choromanski et al., 2022). Let $R = 2m$ denote the total feature dimension, counting both signs of each antithetic pair. For a query or key x after RoPE, we use

$$\phi(x) = \frac{1}{\sqrt{R}} \left(\left[e^{\omega_r^\top \tilde{x} - \|\tilde{x}\|^2/2} \right]_{r=1}^{m}, \left[e^{-\omega_r^\top \tilde{x} - \|\tilde{x}\|^2/2} \right]_{r=1}^{m} \right), \qquad \tilde{x} = \frac{x}{d_h^{1/4}}. \tag{G.1}$$

The directions ω_r are generated blockwise by QR decomposition of Gaussian matrices, normalized to $\|\omega_r\| = \sqrt{d_h}$, shared between queries and keys within each head, and fixed throughout training. The same random-feature realization is used by matched Linear and Sliced-Linear models within each experiment.

Features are evaluated in FP32, and the exponential argument is clipped to $[-20, 20]$ before exponentiation as a numerical safeguard.

Scalar projections and Laplace scale. Each Transformer layer ℓ and attention head h uses separate learned scalar query and key projections

$$p^Q_{\ell,h}, p^K_{\ell,h} : \mathbb{R}^{d_h} \to \mathbb{R}.$$

The implementation supports positive trainable Laplace scales through $\lambda_h = \exp(\rho_h)$. In all experiments reported in this paper, however, the scales are fixed to $\lambda_h = 0.4$ for every head. Since any fixed $\lambda_h > 0$ can be absorbed into the scalar projections, this choice does not restrict the represented kernel family; $\lambda = 0$ corresponds to the Linear-attention boundary.

Normalization. The FAVOR+ feature maps used in all experiments are strictly positive. Hence both the Linear and Sliced-Linear kernels are nonnegative, and normalized attention uses

$$y_i = \frac{\sum_{j \in \mathcal{C}_i} K_{\mathrm{SL}}(q_i, k_j) v_j}{\sum_{j \in \mathcal{C}_i} K_{\mathrm{SL}}(q_i, k_j)}.$$

The denominator is clamped below by $\varepsilon = 10^{-6}$ purely as a numerical safeguard.

Stable Laplace scans. Equation (3.2) is evaluated in rescaled form rather than by forming $e^{\lambda t_j}$ and $e^{-\lambda t_j}$ separately. Let $z_1 \le \cdots \le z_M$ denote the merged sorted query and key coordinates. For the ascending contribution, we maintain

$$U_r = e^{-\lambda(z_r - z_{r-1})} U_{r-1} + \mathbf{1}_{\{r \text{ is a key}\}}\, a_r, \qquad U_0 = 0. \tag{G.2}$$

At a query coordinate $z_r = s_i$,

$$U_r = \sum_{t_j \le s_i} e^{-\lambda(s_i - t_j)} a_j. \tag{G.3}$$

A reverse recurrence gives the contribution from $t_j > s_i$. All exponential arguments in these recurrences are non-positive.

For causal cross-range interactions, projected keys and queries are sorted within each range pair. Keys are ordered before queries when their scalar coordinates coincide, ensuring consistency with the decomposition in (3.2). The scans use $a_j = \phi(k_j) v_j^\top$ for the numerator and $a_j = \phi(k_j)$ for the normalization statistics.

Causal hierarchy and sorting metadata. The causal implementation recursively partitions the temporal axis down to blocks of size B. The n/B dense leaf blocks require $O(nB(R + d_v))$ work. At each of the $\log_2(n/B)$ higher levels, the cross-range scans process $O(n)$ tokens in total, giving

$$O\Big(nRd_v \log \frac{n}{B}\Big)$$

feature-contraction work.

Projected keys and queries are sorted independently within each cross-range problem. Across the hierarchy this adds $O(n \log^2(n/B))$ work. Sorting orders are constructed separately at each level and cached for the backward pass, requiring $O(n \log(n/B))$ integer metadata. The same ordering metadata are shared across feature channels.

Cross-range scans. Within each cross-range problem, temporal causality is automatic. Query and key scalar coordinates are sorted separately and merged into an event ordering, with keys preceding queries at equal coordinates. The two Laplace branches are then evaluated by forward and reverse scan recurrences. In the GPU implementation, the event stream is processed in fixed-size chunks: matrix products form chunk summaries, prefix accumulations propagate contributions between chunks, and small triangular products handle interactions within each chunk. Numerator and normalization statistics are accumulated together.

A sequential implementation using Fenwick trees after coordinate compression is also possible (Fenwick, 1994), with logarithmic tree operations per token. We use the level-wise formulation because it is better suited to parallel GPU execution.

Backward pass. The backward pass differentiates through the same hierarchical cross-range computations and dense leaf blocks without materializing a global $n \times n$ attention matrix. The current implementation retains per-level event data for the backward pass, giving $O(n \log(n/B))$ memory scaling for fixed B, R, and d_v. Recomputation can trade this memory for additional sorting and scan work.