LINEAR ATTENTION LACKS INCENTIVE FOR RECALL, NOT CAPACITY

Anonymous authors
Paper under double-blind review

ABSTRACT

Linear-attention models exhibit disproportionate deficits in prefix-matching associative recall, particularly at long retrieval distances, even when their overall language-modeling loss matches or improves on softmax attention. Motivated by the performance gap, we revisit both real and synthetic recall tasks with different linear attention architectures, narrowly defining the recall-gap at an academic LLM scale. Consistent with prior work, we observe that this architecture-dependent gap is concentrated on precise predictions requiring prefix-matching associative recall. To investigate this gap, we use MQAR and find a shared empirical capacity frontier across the tested linear-attention architectures. Extrapolating this scaling, we find that a model matching a single layer of a trained LLM, can learn to retrieve thousands of synthetic associations across long distances. These results suggest that capacity alone does not explain the natural-language recall gap at this scale. We instead hypothesize that standard pretraining provides insufficient supervision for robust associative retrieval. A simple in-context repetition scheme improves recall-intensive benchmark performance and narrows the long-distance prefix-matching gap without changing the architecture, token budget, or number of optimization steps, at a slight cost to general language modeling. Targeted changes to the training distribution can thus elicit substantial but underutilized retrieval capacity in linear-attention models.

1 INTRODUCTION

Linear attention offers an efficient alternative to softmax attention by compressing the growing key-value cache into a fixed-size recurrent state (Katharopoulos et al., 2020). This allows inference costs per token to remain constant with sequence length, making these architectures attractive for processing long contexts. However, their ability to use information from earlier in the sequence remains a persistent concern. Even when overall language modeling performance is comparable, linear attention models often underperform on associative recall (AR), where previously encountered information must be retrieved from a partial cue (Jelassi et al., 2024; Waleffe et al., 2024; Arora et al., 2023). Understanding this gap is therefore important for translating efficient sequence processing into effective use of long contexts.

A fixed-size state necessarily limits how much information a model can retain. However, the existence of a capacity limit does not establish that this limit explains the recall deficits observed at practical model sizes and context lengths. Poor recall can also arise when training does not reliably elicit a retrieval operation that the architecture is capable of representing. This distinction matters for how the problem is addressed: increasing state capacity may help when storage is the limiting factor, while changing the training distribution may be sufficient when the available capacity is underutilized.

In this work, we investigate this distinction through natural-language evaluation, synthetic recall experiments, and a controlled intervention on the pretraining data. We compare Gated Linear Attention, DeltaNet, Gated DeltaNet, and Kimi Delta Attention with Softmax Attention within a shared Transformer-style block, using language models of approximately 340M parameters trained on 15B tokens of FineWeb-Edu. We first examine where the recall gap appears. Following Arora et al. (2023), we measure predictions that complete rare bigrams previously observed in the same context. The deficit is particularly pronounced for these prefix-matching predictions at long retrieval

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.

distances, while differences are much smaller when a repeated rare token appears after a different prefix. Similarly, while recall QA tasks show no gap in performance, modifying question-answering benchmarks to provide the exact prefix preceding the answer exposes a larger advantage for softmax attention. These results narrow the observed gap to a specific retrieval operation, rather than establishing a uniform deficit across all forms of recall.

We then use synthetic multi-query associative recall (MQAR)Arora et al. (2023) to examine whether state capacity is sufficient to support this operation. Across the evaluated linear attention variants, we find similar empirical capacity limits, with differences appearing more strongly in convergence speed. Scaling experiments suggest that the number of correctly retrieved associations grows approximately linearly with recurrent state size. Using a model matching a single layer of our language models, linear attention and DeltaNet retrieve thousands of synthetic associations from a 16k-token context. These results suggest that insufficient state capacity alone is unlikely to explain the natural-language recall gap at the scale considered here.

Instead, we hypothesize that standard pretraining provides insufficient incentive to learn robust prefix-matching recall. Qualifying AR-hits account for only about 1.7% of tokens in our held-out FineWeb-Edu sample, and their contribution to the training objective competes with other language modeling demands. To strengthen this signal, we introduce a simple in-context repetition scheme during the final stage of pretraining. Repeated subsequences replace part of the training text, leaving the architecture, recurrent state size, total token budget, and number of optimization steps unchanged. This intervention improves prefix-matching recall across the evaluated architectures and reduces the long-distance AR-hit gap, at the cost of a slight decrease in general language modeling performance.

Taken together, our results show that linear attention models possess substantial retrieval capacity that standard language-model pretraining does not fully elicit. Capacity remains a constraint, but the observed recall gap also depends on how strongly the training distribution encourages models to learn and use associative retrieval.

2 Linear Attention

Attention (Vaswani et al., 2023) refers to the operation of selectively retrieving information in a sequence by constructing query, key and value tensors from the input. The most commonly used variant is softmax attention, which uses the softmax function as a similarity kernel to compute the output at time-step t as a normalized sum over all preceding positions. Equivalently in simplified[1] parallel and sequential forms:

$$O = \mathrm{softmax}(QK^\top)V \quad \equiv \quad o_t = \sum_{i=0}^{t} \frac{\exp\left(q_t^\top k_i\right)}{\sum_{j=0}^{t} \exp\left(q_t^\top k_j\right)} v_i \tag{1}$$

$$Q, K, V, O \in \mathbb{R}^{T,d} \qquad q_t, k_t, v_t, o_t \in \mathbb{R}^d \tag{2}$$

In parallel this implies a quadratic scaling to compute interactions between all inputs. During inference, the entire past key value cache (Pope et al., 2022) has to be retained and attended to by the new query, leading to linear scaling in both compute and memory usage. In an effort to improve scaling to process longer sequences, Linear Attention (Katharopoulos et al., 2020) removes the softmax function and uses associativity to reformulate the operation[2] as:

$$O = (QK^\top)V = Q(K^\top V) \tag{3}$$

[1] Leaving out the causal mask and temperature

[2] Simplified to remove normalization, feature maps, and the causal mask

This removes the pairwise interactions and allows the sequential form to be expressed as a recurrence:

$$o_t = \sum_{i=0}^{t} (q_t^\top k_i) v_i = q_t \sum_{i=0}^{t} k_i v_i^\top \tag{4}$$

$$o_t = q_t S_t, \qquad S_t = \sum_{i=0}^{t} k_i v_i^\top = S_{t-1} + k_t v_t^\top \tag{5}$$

$$q_t, k_t, v_t, o_t \in \mathbb{R}^d, S_t \in \mathbb{R}^{d,d} \tag{6}$$

This form compresses the KV-cache to a single fixed state size with a constant scaling in the inference step. In the recurrent form, linear attention closely resembles classical recurrent neural networks (Elman, 1990), however the associativity of the update means it retains the efficient parallelization of softmax attention. Building on this core by drawing inspiration from RNNs, subsequent architectures have added components which modify the state between steps while preserving the efficient parallelism.

Two prominent directions are the Forget Gate (Sun et al., 2023)(Yang et al., 2024a) and the Delta Rule (Schlag et al., 2021)(Yang et al., 2024b).

$$S_t = \alpha_t S_{t-1} + k_t v_t^\top \quad \text{(Forget Gate)} \qquad S_t = (I - k_t k_t^\top) S_{t-1} + k_t v_t^\top \quad \text{(Delta Rule)}$$

Both operators allow the model to erase information from past context, introducing a recency bias and preventing a linear growth of magnitude across sequence length. The forget gate introduces exponential decay. The decay is controlled by the α_t parameter which depends on the input (Yang et al., 2024a) allowing the model to control how much of the past to forget based on its current input. The delta rule instead updates the prediction associated with the current key, selectively modifying overlapping stored information. Related state-space and recurrent architectures (Gu & Dao, 2024; Dao & Gu, 2024; Lahoti et al., 2026; Peng et al., 2023; 2025) use similar state updates but different surrounding blocks. We focus on sequence mixers evaluated within a matched Transformer-style block; Appendix Appendix A summarizes related update rules and details of layer structure. We survey sequence mixers that use one or both of these two update mechanisms, in particular: Gated Linear Attention (Yang et al., 2024a), Delta Net (Yang et al., 2024b), Gated DeltaNet (Yang et al., 2025) and Kimi Delta Attention (Team et al., 2025).

3 Associative Recall

Associative recall (AR) (Hopfield, 1982) is the ability to retrieve information encountered earlier in context using a related partial cue. In language, this supports connecting references to previously introduced entities and recalling facts needed to interpret or continue a passage. For example, after reading 'Maya left her notebook in the kitchen before heading to work," correctly answering 'Where did Maya leave her notebook?" with "In the kitchen" constitutes a successfully completed associative recall query.

Empirically, fixed-state architectures are known to underperform on AR compared to softmax attention (Jelassi et al., 2024; Waleffe et al., 2024; Arora et al., 2023), particularly in long context correlations. The performance gap therefore prevents models from effectively utilizing their beneficial sequence length scaling properties which has made the performance gap between fixed-state models and softmax attention of particular importance in order to unlock this efficient scaling. To investigate the issue and validate prior results using current state-of-the-art sequence mixers, we train a series of LLMs at a standard academic scale of $\sim$340M parameters on 15B tokens of Fineweb-Edu(Penedo et al., 2024), using the Llama-2 tokenizer (Touvron et al., 2023) on a context length of 4096 tokens. The appendix Appendix B provides further details on training and detailed model setup.

We use associative recall broadly to mean retrieving previously encountered information from a cue. Our primary operational measure is narrower: predicting a token after a previously observed matching prefix. We refer to qualifying repeated-bigram predictions as AR-hits. The target token for an AR-hit is the second token in a bigram which has already appeared earlier in the same context

and where the bigram's estimated corpus frequency is at most 1,250 occurrences per 10B tokens (Arora et al., 2023).

Motivating this narrow formulation is that the solution to retrieving an AR-hit closely resembles the operation performed by induction head circuits (Elhage et al., 2021; Olsson et al., 2022) found in transformers which are known to be strongly associated with ICL operations. We sample 16M tokens from a held-out sample of the training dataset and find that 1.7 percent of tokens are AR-hits according to this definition. This number is notably lower than reported by Zoology, which reports 6.4 % AR-hits. The main reason for this difference stems from the dataset itself as Fineweb-Edu contains mainly filtered natural language in comparison to the Pile (Gao et al., 2020) which used a larger share of code and other forms of unstructured text which contains higher frequency of bigrams, see Appendix C for a detailed description of this difference.

			Softmax loss / change vs. softmax (%)				
Group	Token Count	% of total	Softmax	DN	GLA	GDN	KDA
AR-hits	273,070	1.708	0.886	+20.2	+25.1	+16.3	+13.2
Other tokens	15,712,918	98.292	2.391	+1.5	+1.0	+0.3	−1.0
AR-hit share of net gap			–	8.0%	14.4%	29.4%	–
Rare tokens, 1st mention							
	35,323	0.221	5.791	+1.1	+0.7	0.0	−1.1
Rare tokens, 2nd mention							
New Prefix							
1–511 tokens back	16,158	0.101	4.027	+3.4	+1.8	+1.0	−2.6
512+ tokens back	3,541	0.022	5.355	+2.8	+3.7	+2.5	+0.4
Same Prefix							
1–511 tokens back	12,472	0.078	1.794	+12.2	+13.1	+9.8	+7.1
512+ tokens back	3,158	0.020	2.409	+27.3	+35.3	+26.1	+21.6

Table 1: Softmax loss and percentage change in loss for pretrained models at 15B. Model changes are relative to the softmax loss for the corresponding row. The AR-hit share row gives the token-weighted AR-hits contribution to the net loss gap across AR-hits and other tokens. Computed at a maximum sequence length of 4096 tokens

The results show that while the architectures achieve comparable language modeling performance, there's a noticeable gap at the AR-hits, particularly at longer contexts. The differences between linear attention models appear to be mostly tied to their overall loss, with KDA and GDN consistently being slightly better at all subtasks compared to DN and GLA.

To test whether the deficit extends beyond repeated prefixes, we compare rare-token [3] predictions at their first and second occurrences, separating second occurrences by whether the preceding prefix matches. The diagnostic does not isolate semantic recall, but measures whether the architectural difference persists without an exact prefix match.

On the first mention, loss for these tokens is uniformly high, with relatively small differences between different sequence mixers. However, on the 2nd mention, we see the predictions splitting into two cases. If the token is preceded by the same tokens as when it was initially mentioned, loss sharply drops for all architectures but in particular for softmax attention. If it is not, then loss remains relatively high though noticeably lower than at the first mention of the word.

3.1 Recall Benchmark Suite

To benchmark the above gap we use a standard suite of six benchmarks (Arora et al., 2024). SWDE (Lockard et al., 2019; Arora et al., 2025) and FDA (Wu et al., 2021) measure attribute extraction from documents. SQuAD (Rajpurkar et al., 2016), TriviaQA (Joshi et al., 2017), Natural Questions (Kwiatkowski et al., 2019), and DROP (Dua et al., 2019) use cloze-formatted prompts with supporting documents in context.

[3] An individual token matching the bigram rarity condition.

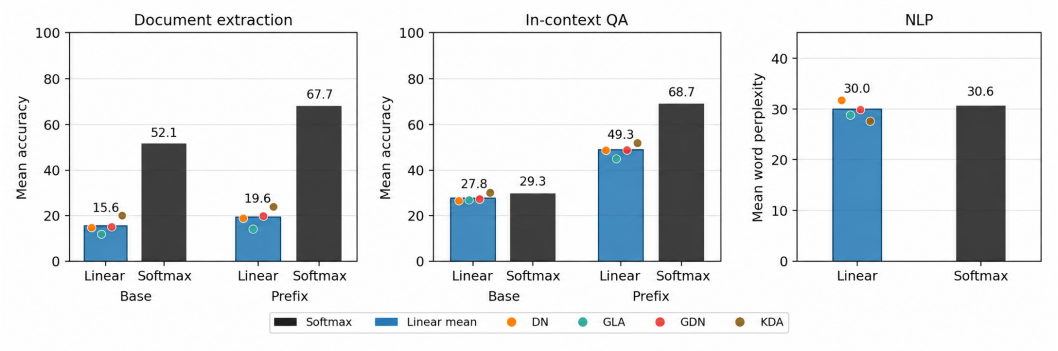


Figure 1: Mean accuracy on the recall suite, divided by category. Document Extraction consists of SWDE and FDA. QA encompasses SQuAD, TQA, NQ and DROP. The linear attention bar represents the mean across linear-attention architectures; dots represent individual sequence mixers. NLP performance on LAMBADA and WikiText provides a reference for overall language modeling performance

For these tasks we can see that softmax attention is clearly better for document extraction, while the gap on QA is small. Different linear attention variants mostly perform relatively similarly and all have a mean accuracy within a few percentage points of each other. Once again, KDA is the strongest performer, but its performance appears to be mainly correlated with a generally better language understanding rather than specifically for recall.

Reviewing the details behind the benchmarks reveals why the gap concentrates on document extraction tasks. The primary difference between them is how the generation prompt is formatted: for SWDE and FDA, the prompt cues closely resemble how the information is initially presented while the QA tasks use a cloze (Taylor, 1953; Brown et al., 2020) format which phrases the generation prompt as a reply to the question with the final answer left out (Arora et al., 2024). Therefore for these queries, there is no immediate preceding context to latch on to and the model has to rely on more general context rather than precise prefix retrieval formulated by induction heads. Changing the format of the prompts to more closely resemble the prefix-matching retrieval, we instead construct the prompt from the four words immediately preceding the selected answer's first occurrence in context. Matching the results regarding the recall of rare tokens, the prefix prompts substantially improve accuracy, especially on the QA datasets, and expose a larger advantage for softmax attention in comparison to the linear architectures. Complete results by task can be found in the appendix Appendix E. Therefore, it appears that at this scale, the gap between the architecture types is particularly concentrated in this specific operation, rather than reflecting strictly better performance at recall in general.

4 MQAR

The observed issue of precise, associative recall based on a prefix forms the basis of a synthetic task frequently used to test linear sequence mixers, multi-query associative recall (MQAR) (Arora et al., 2023). The MQAR task forms a synthetic version of AR-hits, where a model has to learn to bind key-value pairs in context to later retrieve the value when the key next appears, forming the core of an induction head circuit. The keys and the values are sampled independently which means the model cannot memorize the answer in its weights and in order to retrieve the correct value, the model has to store the key–value association and emit the value when prompted by the key. See Appendix F for details about how the task is constructed.

4.1 Experiments

In this synthetic environment, we compare six different architectures, the five described above, as well as simple linear attention. Compared to softmax attention, which can perfectly solve the task at any scale, the difference between linear architectures is relatively small with no architecture appear-

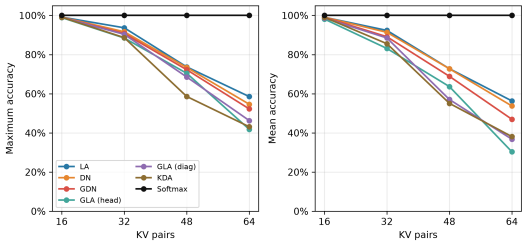


Figure 2: Maximum and Mean Accuracy on MQAR on four different KV pair settings. Vocabulary size is 8193, and sequence length is 512. All models use dimension 64 split into 4 heads. Mean accuracy is averaged over three random seeds at the best learning rate.

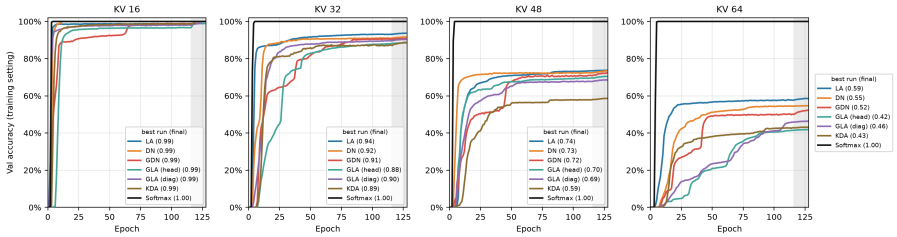


Figure 3: Accuracy over training epochs for different sequence mixers. The grey region shows the learning-rate cooldown phase

ing to close the gap in either capacity or convergence speed. Simple linear attention is the overall strongest performer. A clear trend is that gated architectures perform worse as they take longer to converge which aligns with previous studies (Trockman et al., 2024; Okpekpe & Orvieto, 2025).

4.2 Capacity Scaling

From the training curves, all architectures saturate at roughly 30 correctly retrieved KV-pairs. We hypothesize that this reflects a capacity bound set by the state size, leading to an approximate limit of $\approx S/30$ for a state of $S = h\,d_k d_v = 1024$. Prior work bounds this capacity under different assumptions. Interference-free readout requires mutually orthogonal keys, limiting capacity to d_k pairs retrieved per head (Schlag et al., 2021), which for a fixed number of square heads grows only as $\sqrt{S}$. Conversely, idealized constructions that tolerate interference need a state that grows only logarithmically in the number of pairs it can retrieve (Huang et al., 2025). Empirically, our implied scaling falls somewhere in between these ratios. To see whether this scaling holds across other scales we extend the analysis in both directions with larger models of model dimension 128, with a state size of 4096, and smaller models of model dimension 32 and a total state size of 256. Matching the scaling of the state size by a factor of four, we increase the number of KV-pairs by the same ratio.

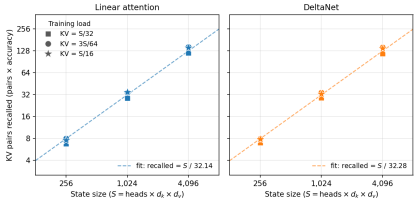


Figure 4: Capacity scaling for Linear Attention and Delta Net, measured at three different scales of state size.

A second difficulty factor is the vocabulary size, and as such we fit similar curves for LA on vocabulary size 1024 and 32768 which yields the ratios $S/25$ and $S/36$ compared to the $S/32$ ratio for a vocabulary size of 8192, only shifting performance slightly. This agrees with concurrent work on Mamba (Koren et al., 2026), which likewise finds recall capacity growing linearly with state size and only logarithmically with vocabulary size. For reliable recall at $|\mathcal{V}| = 8192$, their law implies roughly 20–50 state entries per pair, consistent with our $S/32$, and it predicts the shift we observe across vocabulary sizes. Our results thus suggest that the same scaling holds for linear attention models.

4.3 MQAR AT LLM SCALE

Linear scaling in relation to state size implies that at the scale of LLMs a single layer could feasibly store several thousands bound associations. Consider the previously evaluated models at the ∼340M-parameter scale. A single layer in this model has a state size of $S = 131072$ meaning that if the scaling holds, it has the capacity to perform associative recall on approximately ∼3,500 KV pairs at the vocabulary size of 32000 used for language modeling.

To evaluate recall at this scale, we train with a ramping schedule: the model starts with fewer KV pairs, where MQAR accuracy improves faster, and the number of pairs increases over training. This lets us reach large KV-pair counts sooner while testing whether the model can recall associations near its estimated capacity. The model correctly retrieves 3,442 of 4,096 stored KV pairs, matching the scaling.

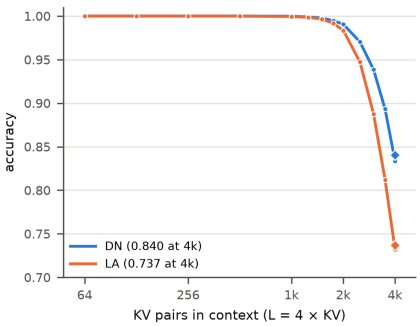


Figure 5: Large Scale MQAR (1 layer, d_{model}=1024, 8×128 heads, 131k state floats, vocab 32k). DN retrieves 84.0% (3,442) and LA 73.7% (3,018) of 4,096 pairs from a 16k-token context.

5 REPEATING IN CONTEXT

Our MQAR experiments show that linear-attention models can retrieve thousands synthetic associations at recurrent state sizes comparable to those used in our language models. Therefore, fixed state size does not prevent these architectures from learning the associative retrieval operation underlying AR-hits at this scale. This finding challenges the view that retrieval gaps arise primarily from insufficient state capacity (Arora et al., 2025; Liu et al., 2025). The results instead point to a difference in learning dynamics. Although the evaluated linear-attention models can represent effective associative-recall solutions, they learn them more slowly than softmax attention under comparable MQAR training conditions. This suggests a weaker inductive bias toward retrieval, rather than an inability to represent it. Consistent with this interpretation, toy-model studies report that copying structure emerges more gradually under linear attention (Lavie et al., 2026). We hypothesize that this learning disadvantage also matters during language-model pretraining, where direct supervision for prefix-matching recall is sparse: explicit AR-hits occur alongside many other predictive demands. Prior work on softmax models finds that induction-head emergence depends on the data distribution, particularly on the repetition of information in context (Chan et al., 2022; Zucchet et al., 2025; Aoyama et al., 2026).

Therefore, to encourage recall, we use a simple in-context repetition scheme. To repeat sentences while attempting to preserve the natural language structure, we target sampling subsequences at the start of sentences, detected as positions where whitespace follows a period. From this point we splice sequences of varying lengths with a mean sequence length of 64 tokens, chosen to roughly match the length of a sentence. The subsequences are then inserted into the context after the original copying location at a different sentence break chosen at random or at the end of the sequence.

We repeat tokens targeting a ratio of 10 % of the training tokens which increases the proportion of AR-hits by approximately 1.15 percentage points to 2.86 %, an increase of about 68 %, with the remaining repetitions being considered above the rarity threshold. This repetition is introduced at the start of the learning-rate cooldown, covering approximately the final 20% of training. The total token budget and number of optimization steps remain unchanged as repeated tokens replace tokens that would otherwise introduce new text. Unlike Birdie (Blouir et al., 2025) and Just Read Twice (Arora et al., 2024), our approach changes only the training data; it preserves the original architecture and causal inference.

Introducing repetition rapidly improves recall performance while slightly worsening general language-modeling performance. After this initial phase, the gains in recall while the NLP loss curve largely tracks the matched runs. All evaluated architectures improve on the prefix-prompt benchmarks. The relative benefit varies across tasks, and the long-distance AR-hit loss gap to softmax attention narrows by 28% to the baseline softmax model, see Appendix G.

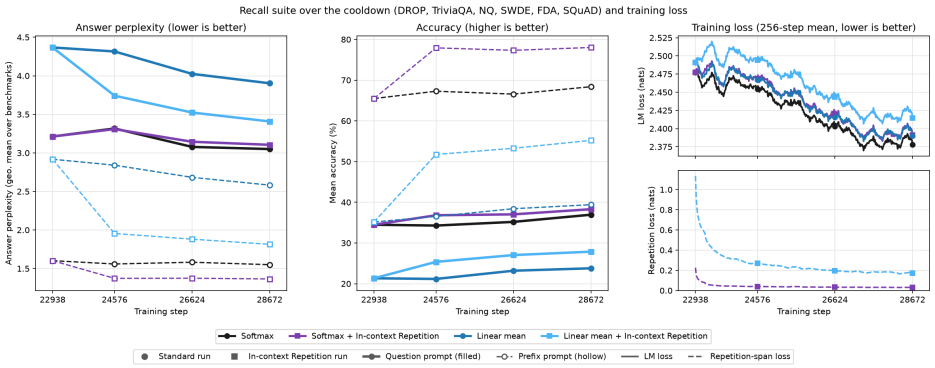


Figure 6: Answer perplexity and accuracy of the recall suite of benchmarks, evaluated at four intermediate checkpoints during the LR-cooldown phase. Loss is split into loss on regular LM tokens and loss on the repeated sequence. Linear attention aggregates the mean of each objective across the four variants.

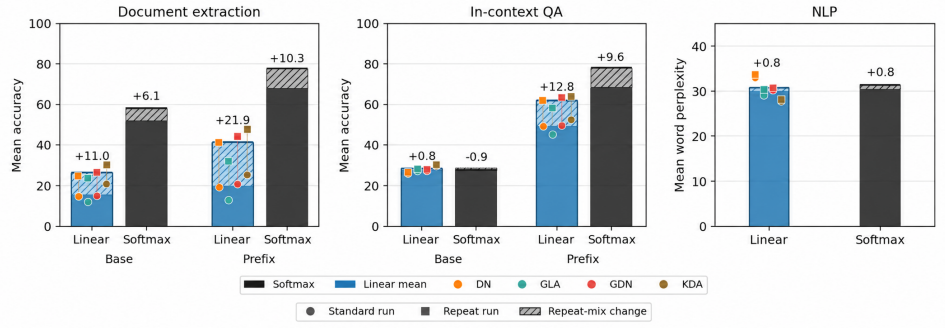


Figure 7: Mean accuracy on the recall suite, divided by category. Document Extraction consists of SWDE and FDA. QA encompasses, SQuAD, TQA, NQ and DROP. Linear attention bar represents mean of linear attention architectures, round dots represent individual sequence mixers trained with the standard methods, while square represent the with lines showing how performance changes between them. NLP performance on Lambada and Wikitext as reference for overall language modeling performance

Evaluation on trained models shows consistent improvement on the prefix matching recall tasks for all architectures at the cost of a slight decrease in language modeling performance. Linear attention models benefit relatively more, reducing the gap and suggesting that repetition may help the models form long-context recall circuits.

5.1 Needle In A Haystack

We evaluate long-context recall with RULER's Needle in a Haystack (NIAH) tasks (Hsieh et al., 2024). NIAH-1 places a seven-digit number in a repeated passage[4]; NIAH-2 places a number in natural text, and NIAH-3 places a 32-character UUID in natural text[5]. We add NIAH-4, which uses a two-word value sampled like the key[6], to test retrieval of words rather than numbers or UUIDs. Some models emit ": " before the answer. Adding those characters to the prompt raises DeltaNet's NIAH-2 accuracy from 24.4% to 71.6%, illustrating the sensitivity of exact-match scores to prompt wording. We also test a prefix-only prompt that removes the instruction and question. Because these prompt changes substantially affect exact-match accuracy, we report teacher-forced token accuracy as an additional measure. That measure can overstate word retrieval when later tokens are predictable

[4]"The sky is blue, the grass is green. There and back again."

[5]Paul Graham essays.

[6]Using Wonderwords (Rebguns, 2025)

from a word's beginning. Nevertheless, on artificial data, this also requires recall, though only one token at a time. Most models retrieve roughly 90% of tokens even for UUID values. Repetition improves retrieval with an exact prefix, particularly for word values. The differing results across value types suggest that retrieval content also affects NIAH performance. Complete results appear in Appendix H.

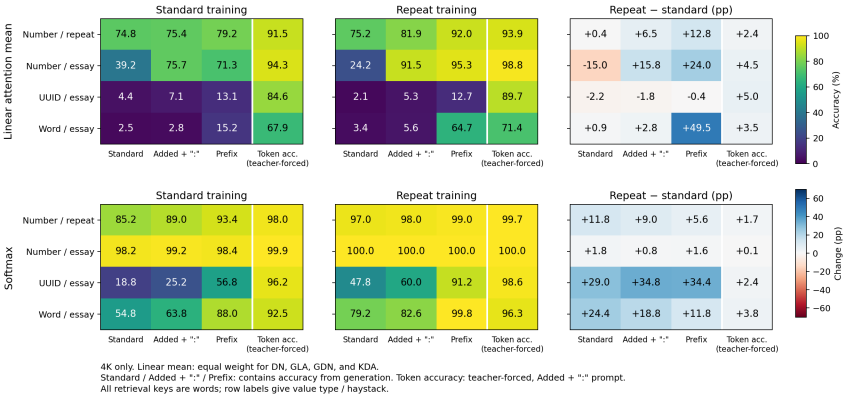


Figure 8: Accuracy on NIAH benchmarks. Number/Repeat corresponds to NIAH-1, Number/Essay corresponds to NIAH-2, UUID/Essay corresponds to NIAH-3 and Word/Essay corresponds to our NIAH-4. Teacher-forced token accuracy uses the added colon form of the generation prompt

6 Conclusions

Across the recall measures studied here, the clearest gap between linear and softmax attention occurs in prefix-matching retrieval. The linear-attention update rules differ relatively little in MQAR performance, and their performance on MQAR does not consistently predict their performance on the natural-language benchmarks. Although recurrent state size limits retrieval capacity, our synthetic experiments show that models at the state sizes studied can learn to retrieve thousands of associations. We therefore hypothesize that the observed prefix-retrieval gap reflects, at least in part, a weak training signal for this operation. Increasing in-context repetition during training narrows the gap without changing the architecture, though it slightly reduces general language-modeling performance. These conclusions are limited to the model sizes and benchmarks studied here. Performance differences between fixed-state and softmax models have also been observed at larger scales on other benchmarks (Waleffe et al., 2024), where additional factors may matter. Our language-model comparisons use a single training run per architecture; because training randomness can affect results (Allen-Zhu, 2026), small differences between individual models should be interpreted cautiously.

6.1 Future Work

Future work could test whether training linear-attention models on longer, contextually similar sequences improves long-context retrieval, as reported for softmax attention (Shi et al., 2024; Staniszewski et al., 2025). Targeted repetition could provide additional retrieval examples during training. Our repetition scheme did not close the gap to softmax attention and slightly reduced language-modeling performance. Developing training interventions that improve recall without this trade-off remains an open problem.

AI USE STATEMENT

In this work, we used generative AI tools to assist with implementing research code, suggesting experimental parameters, refining hypotheses, providing feedback on research methods and experiments, and interpreting and organizing results. We did not use these tools to generate synthetic datasets, clean or reformat datasets, or develop theoretical models or conceptual frameworks. Formal proofs, translation, and qualitative or thematic analysis were not part of this work.

We also used generative AI tools to create or revise figures, identify and summarize related literature, and provide feedback on the manuscript's structure, language, and reference formatting. The authors wrote the initial complete draft and incorporated AI-assisted edits during revision. The authors reviewed AI-assisted code, approved experiments before they were run, and verified and interpreted the experimental results. The research question and overall direction originated with the authors.

The authors take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

REFERENCES

Zeyuan Allen-Zhu. Physics of language models: Part 4.1, architecture design and the magic of canon layers, 2026. URL https://arxiv.org/abs/2512.17351.

Tatsuya Aoyama, Ethan Gotlieb Wilcox, and Nathan Schneider. Predicting the emergence of induction heads in language model pretraining, 2026. URL https://arxiv.org/abs/2511.16893.

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, Aman Timalsina, Aaryan Singhal, Benjamin Spector, Sabri Eyuboglu, Xinyi Zhao, Ashish Rao, Atri Rudra, and Christopher Ré. Just read twice: closing the recall gap for recurrent language models, 2024. URL https://arxiv.org/abs/2407.05483.

Simran Arora, Brandon Yang, Sabri Eyuboglu, Avanika Narayan, Andrew Hojel, Immanuel Trummer, and Christopher Ré. Language models enable simple systems for generating structured views of heterogeneous data lakes, 2025. URL https://arxiv.org/abs/2304.09433.

Sam Blouir, Jimmy T. H. Smith, Antonios Anastasopoulos, and Amarda Shehu. Birdie: Advancing state space models with reward-driven objectives and curricula, 2025. URL https://arxiv.org/abs/2411.01030.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D. Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33:1877–1901, 2020. URL https://arxiv.org/abs/2005.14165.

Stephanie C. Y. Chan, Adam Santoro, Andrew K. Lampinen, Jane X. Wang, Aaditya Singh, Pierre H. Richemond, Jay McClelland, and Felix Hill. Data distributional properties drive emergent in-context learning in transformers, 2022. URL https://arxiv.org/abs/2205.05055.

Tri Dao and Albert Gu. Transformers are ssms: Generalized models and efficient algorithms through structured state space duality, 2024. URL https://arxiv.org/abs/2405.21060.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. Drop: A reading comprehension benchmark requiring discrete reasoning over paragraphs, 2019. URL https://arxiv.org/abs/1903.00161.

Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Nova DasSarma, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. A mathematical framework for transformer circuits. *Transformer Circuits Thread*, 2021. https://transformer-circuits.pub/2021/framework/index.html.

Jeffrey L. Elman. Finding structure in time. *Cognitive Science*, 14(2):179–211, 1990.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, Shawn Presser, and Connor Leahy. The pile: An 800gb dataset of diverse text for language modeling, 2020. URL https://arxiv.org/abs/2101.00027.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. The language model evaluation harness, 07 2024. URL https://zenodo.org/records/12608602.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. In *First Conference on Language Modeling*, 2024. URL https://arxiv.org/abs/2312.00752.

Ali Hatamizadeh, Yejin Choi, and Jan Kautz. Gated deltanet-2: Decoupling erase and write in linear attention, 2026. URL https://arxiv.org/abs/2605.22791.

John J. Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences*, 79(8):2554–2558, 1982. doi: 10.1073/pnas.79.8.2554.

Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, Yang Zhang, and Boris Ginsburg. Ruler: What's the real context size of your long-context language models?, 2024. URL https://arxiv.org/abs/2404.06654.

Ningyuan Huang, Miguel Sarabia, Abhinav Moudgil, Pau Rodriguez, Luca Zappella, and Federico Danieli. Understanding input selectivity in mamba: Impact on approximation power, memorization, and associative recall capacity, 2025. URL https://arxiv.org/abs/2506.11891.

Samy Jelassi, David Brandfonbrener, Sham M. Kakade, and Eran Malach. Repeat after me: Transformers are better than state space models at copying, 2024. URL https://arxiv.org/abs/2402.01032.

Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension, 2017. URL https://arxiv.org/abs/1705.03551.

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are RNNs: Fast autoregressive transformers with linear attention. In Hal Daumé III and Aarti Singh (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 5156–5165. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/katharopoulos20a.html.

Yuval Koren, Assaf Ben-Kish, Raja Giryes, Lior Wolf, and Itamar Zimerman. On the recall scaling laws in mamba: A theoretical and mechanistic study via hashing, 2026. URL https://arxiv.org/abs/2609.07681.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Matthew Kelcey, Jacob Devlin, Kenton Lee, Kristina N. Toutanova, Llion Jones, Ming-Wei Chang, Andrew Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. Natural questions: a benchmark for question answering research. *Transactions of the Association of Computational Linguistics*, 2019.

Aakash Lahoti, Kevin Y. Li, Berlin Chen, Caitlin Wang, Aviv Bick, J. Zico Kolter, Tri Dao, and Albert Gu. Mamba-3: Improved sequence modeling using state space principles, 2026. URL https://arxiv.org/abs/2603.15569.

Itay Lavie, Kirsten Fischer, Andrey Lekov, Frederic Van Maele, Zohar Ringel, and Moritz Helias. Phase transitions in attention: A bayesian theory of copy head emergence, 2026. URL https://arxiv.org/abs/2606.12058.

Kai Liu, Jianfei Gao, and Kai Chen. Scaling up the state size of RNN LLMs for long-context scenarios. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 11516–11529, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.564. URL https://aclanthology.org/2025.acl-long.564/.

Colin Lockard, Prashant Shiralkar, and Xin Luna Dong. OpenCeres: When open information extraction meets the semi-structured web. In Jill Burstein, Christy Doran, and Thamar Solorio (eds.), *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 3047–3056, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1309. URL https://aclanthology.org/N19-1309.

Destiny Okpekpe and Antonio Orvieto. Revisiting associative recall in modern recurrent models, 2025. URL https://arxiv.org/abs/2508.19029.

Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Scott Johnston, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. In-context learning and induction heads. *Transformer Circuits Thread*, 2022. URL https://transformer-circuits.pub/2022/in-context-learning-and-induction-heads/.

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.

Bo Peng, Eric Alcaide, Quentin Anthony, Alon Albalak, Samuel Arcadinho, Stella Biderman, Huanqi Cao, Xin Cheng, Michael Chung, Matteo Grella, Xuzheng He, Huanqi Hou, Jiaju Lin, Przemyslaw Kazienko, Jan Kocon, Jiaming Kong, Bartlomiej Koptyra, Hayden Lau, Ferdinand Mom, Atsushi Saito, Guangyu Song, Johan S. Wind, Ruichong Zhang, Zhenyuan Zhang, Qihang Zhou, and Jian Zhou. RWKV: Reinventing RNNs for the Transformer Era. *arXiv preprint arXiv:2305.13048*, 2023. URL https://arxiv.org/abs/2305.13048.

Bo Peng, Daniel Goldstein, Quentin Anthony, Alon Albalak, Eric Alcaide, Stella Biderman, Eugene Cheah, Xingjian Du, Teddy Ferdinan, Haowen Hou, Przemysław Kazienko, Kranthi Kiran GV, Jan Kocoń, Bartłomiej Koptyra, Satyapriya Krishna, Ronald McClelland Jr., Jiaju Lin, Niklas Muennighoff, Fares Obeid, Atsushi Saito, Guangyu Song, Haoqin Tu, Cahya Wirawan, Stanisław Woźniak, Ruichong Zhang, Bingchen Zhao, Qihang Zhao, Peng Zhou, Jian Zhu, and Rui-Jie Zhu. Eagle and finch: Rwkv with matrix-valued states and dynamic recurrence, 2024. URL https://arxiv.org/abs/2404.05892.

Bo Peng, Ruichong Zhang, Daniel Goldstein, Eric Alcaide, Haowen Hou, Janna Lu, William Merrill, Guangyu Song, Kaifeng Tan, Saiteja Utpala, Nathan Wilce, Johan S. Wind, Tianyi Wu, Daniel Wuttke, and Christian Zhou-Zheng. RWKV-7 "Goose" with Expressive Dynamic State Evolution. *arXiv preprint arXiv:2503.14456*, 2025. URL https://arxiv.org/abs/2503.14456.

Reiner Pope, Sholto Douglas, Aakanksha Chowdhery, Jacob Devlin, James Bradbury, Anselm Levskaya, Jonathan Heek, Kefan Xiao, Shivani Agrawal, and Jeff Dean. Efficiently scaling transformer inference, 2022. URL https://arxiv.org/abs/2211.05102.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. Technical report, OpenAI, 2019.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100,000+ questions for machine comprehension of text, 2016. URL https://arxiv.org/abs/1606.05250.

Maxim Rebguns. Wonderwords, 2025. URL https://pypi.org/project/wonderwords/3.0.1/. Python software package.

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

Noam Shazeer. GLU variants improve transformer. *arXiv preprint arXiv:2002.05202*, 2020. URL https://arxiv.org/abs/2002.05202.

Weijia Shi, Sewon Min, Maria Lomeli, Chunting Zhou, Margaret Li, Gergely Szilvasy, Rich James, Xi Victoria Lin, Noah A. Smith, Luke Zettlemoyer, Scott Yih, and Mike Lewis. In-context pretraining: Language modeling beyond document boundaries, 2024. URL https://arxiv.org/abs/2310.10638.

Konrad Staniszewski, Szymon Tworkowski, Sebastian Jaszczur, Yu Zhao, Henryk Michalewski, Łukasz Kuciński, and Piotr Miłoś. Structured packing in llm training improves long context utilization, 2025. URL https://arxiv.org/abs/2312.17296.

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.

Wilson L. Taylor. Cloze Procedure: A new tool for measuring readability. *Journalism Quarterly*, 30(4):415–433, 1953. doi: 10.1177/107769905303000401.

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.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. Llama 2: Open foundation and fine-tuned chat models, 2023. URL https://arxiv.org/abs/2307.09288.

Asher Trockman, Hrayr Harutyunyan, J. Zico Kolter, Sanjiv Kumar, and Srinadh Bhojanapalli. Mimetic initialization helps state space models learn to recall, 2024. URL https://arxiv.org/abs/2410.11135.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2023. URL https://arxiv.org/abs/1706.03762.

Roger Waleffe, Wonmin Byeon, Duncan Riach, Brandon Norick, Vijay Korthikanti, Tri Dao, Albert Gu, Ali Hatamizadeh, Sudhakar Singh, Deepak Narayanan, Garvit Kulshreshtha, Vartika Singh, Jared Casper, Jan Kautz, Mohammad Shoeybi, and Bryan Catanzaro. An empirical study of mamba-based language models, 2024. URL https://arxiv.org/abs/2406.07887.

Kaiyue Wen, Zhiyuan Li, Jason Wang, David Hall, Percy Liang, and Tengyu Ma. Understanding warmup-stable-decay learning rates: A river valley loss landscape perspective, 2024. URL https://arxiv.org/abs/2410.05192.

E. Wu, K. Wu, R. Daneshjou, et al. How medical ai devices are evaluated: limitations and recommendations from an analysis of FDA approvals. *Nature Medicine*, 27:582–584, April 2021. doi: 10.1038/s41591-021-01312-x.

Songlin Yang and Yu Zhang. Fla: A triton-based library for hardware-efficient implementations of linear attention mechanism, January 2024. URL https://github.com/fla-org/flash-linear-attention.

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

Songlin Yang, Bailin Wang, Yu Zhang, Yikang Shen, and Yoon Kim. Parallelizing linear transformers with the delta rule over sequence length. In *Advances in Neural Information Processing Systems*, volume 37, 2024b. URL https://arxiv.org/abs/2406.06484.

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

Yu Zhang and Songlin Yang. Flame: Flash language modeling made easy, January 2025. URL https://github.com/fla-org/flame.

Nicolas Zucchet, Francesco d'Angelo, Andrew K. Lampinen, and Stephanie C. Y. Chan. The emergence of sparse attention: impact of data distribution and benefits of repetition, 2025. URL https://arxiv.org/abs/2505.17863.

A Sequence Mixer Details

The table shows the update rule for a set of different SSMs, RNNs and Linear Attention models. Several of them overlap significantly with only minor differences separating how the update works. They do however deviate significantly from the standard transformer layout in their overall layer structure and are not explored in detail in this study.

The layer structure of linear transformer variants adds several additional components to the standard transformer layout, partly to account for the additional input dependences but also as they have been shown to improve performance generally. To isolate any differences between models to only the sequence mixer operation, we apply a standard setup following the convention used in gated delta net for all architectures across experiments. This allows for fair comparison between only the sequence mixers.

For an input $x_t = \bar{x}_t^{(\ell)}$, the mixers compute

$$\tilde{q}_t = W_q x_t, \tilde{k}_t = W_k x_t, \tilde{v}_t = W_v x_t, \beta_t = \sigma(W_\beta x_t) \tag{7}$$

$$\tilde{v}_t = \beta v_t \tag{8}$$

$$\mathcal{C}_4(u)_t = \sum_{\tau=0}^{3} c_\tau \odot u_{t-\tau}, \qquad u_s = 0 \text{ for } s < 1 \tag{9}$$

$$q_t = \mathrm{SiLU}(\mathcal{C}_4^q(\tilde{q})_t)\,, \tag{10}$$

$$k_t = \mathrm{SiLU}\Big(\mathcal{C}_4^k(\tilde{k})_t\Big)\,, \tag{11}$$

$$v_t = \mathrm{SiLU}(\mathcal{C}_4^v(\tilde{v})_t)\,, \tag{12}$$

$$o_t = \mathrm{SequenceMixer}(q_t, k_t, v_t) \tag{13}$$

$$\mathcal{O}_\phi(o_t, x_t) = W_o\ \mathrm{vec}[\mathrm{RMSNorm}_\varepsilon(o_t) \odot \mathrm{SiLU}(W_g x_t)] \tag{14}$$

Left out are the individual input projections for the additional components, e.g. the decay projection. A particularly important component is the short convolution $\mathcal{C}_4$, a causal depthwise 1D convolution with a kernel width of four that is applied along the sequence dimension to capture local interactions between neighboring tokens. As in Gated Delta Net, it is applied to the query, key, and value representations before the recurrent sequence-mixing operation. This local mixing is particularly

Table 2: Recurrent state-update rules for linear-attention architectures. The common readout is $\mathbf{o}_t = \mathbf{S}_t^\top \mathbf{q}_t$.

Architecture	State update	Mechanism
LA[a]	$\mathbf{S}_t = \mathbf{S}_{t-1} + \mathbf{k}_t \mathbf{v}_t^\top$	Additive write
RWKV-5[b]	$\mathbf{S}_t = \mathrm{Diag}(\boldsymbol{\alpha}) \mathbf{S}_{t-1} + \mathbf{k}_t \mathbf{v}_t^\top$	Fixed channel-wise decay
Mamba-2[c]	$\mathbf{S}_t = a_t \mathbf{S}_{t-1} + \mathbf{k}_t \mathbf{v}_t^\top$	Scalar decay per head
GLA[d]	$\mathbf{S}_t = \mathrm{Diag}(\boldsymbol{\alpha}_t) \mathbf{S}_{t-1} + \mathbf{k}_t \mathbf{v}_t^\top$	Input-dependent channel-wise decay
RWKV-6[e]	$\mathbf{S}_t = \mathrm{Diag}(\boldsymbol{\alpha}_t) \mathbf{S}_{t-1} + \mathbf{k}_t \mathbf{v}_t^\top$	Scalar decay per head
DN[f]	$\mathbf{S}_t = \left(\mathbf{I} - \beta_t \mathbf{k}_t \mathbf{k}_t^\top\right) \mathbf{S}_{t-1} + \beta_t \mathbf{k}_t \mathbf{v}_t^\top$	Delta update
GDN[g]	$\mathbf{S}_t = \alpha_t \left(\mathbf{I} - \beta_t \mathbf{k}_t \mathbf{k}_t^\top\right) \mathbf{S}_{t-1} + \beta_t \mathbf{k}_t \mathbf{v}_t^\top$	Scalar decay and delta update
KDA[h]	$\mathbf{S}_t = \left(\mathbf{I} - \beta_t \mathbf{k}_t \mathbf{k}_t^\top\right) \mathrm{Diag}(\boldsymbol{\alpha}_t) \mathbf{S}_{t-1} + \beta_t \mathbf{k}_t \mathbf{v}_t^\top$	Channel-wise decay and delta update
GDN2[i]	$\mathbf{S}_t = \left(\mathbf{I} - \mathbf{k}_t (\mathbf{b}_t \odot \mathbf{k}_t)^\top\right) \mathrm{Diag}(\boldsymbol{\alpha}_t) \mathbf{S}_{t-1} + \mathbf{k}_t (\mathbf{w}_t \odot \mathbf{v}_t)^\top$	Channel-wise decay, erase, and write
RWKV-7[j]	$\mathbf{S}_t = \left[\mathrm{Diag}(\boldsymbol{\alpha}_t) - (\mathbf{a}_t \odot \hat{\mathbf{k}}_t) \hat{\mathbf{k}}_t^\top\right] \mathbf{S}_{t-1} + \mathbf{k}_t \mathbf{v}_t^\top$	Channel-wise decay, rank-one erase and write

[a] Linear Attention (Katharopoulos et al., 2020).
[b] RWKV-5 (Eagle) (Peng et al., 2024).
[c] Mamba 2 (Dao & Gu, 2024).
[d] Gated Linear Attention (Yang et al., 2024a).
[e] RWKV-6 (Finch) (Peng et al., 2024).
[f] DeltaNet (Yang et al., 2024b).
[g] Gated DeltaNet (Yang et al., 2025).
[h] Kimi Delta Attention (Team et al., 2025).
[i] Gated DeltaNet-2 (Hatamizadeh et al., 2026).
[j] RWKV-7 (Peng et al., 2025).

relevant for associative recall since it allows for local token shifts, which provides a shortcut for forming induction heads in a single layer (Yang et al., 2024b) enabling it to perform MQAR without additional layers.

The layers were implemented using the FLA library Yang & Zhang (2024) with modifications to standardize layer parameters.

B Model and technical details

The models consist of 20 layers with model dimension 1024, using 8 heads, yielding a total state size of 131072 per layer. Each layer uses a sequence mixer interleaved with a SwiGLU (Shazeer, 2020) MLP with an intermediate dimension of 2752. The models are not parameter matched, as some add additional input necessary for their update rules, for example per diagonal decay requires more parameters than head-wise decay.

Each model is trained on 15B tokens, corresponding to approximately 44 tokens per parameter for a 340M-parameter model. Each step used a batch size of 128, with packed sequences of length 4096 resulting in 524288 tokens per step for 28672 steps. The WSD (Wen et al., 2024) scheduler with a 10 % warmup and 20 % cooldown phase, all models were trained with a learning rate of 4.5e-4. The training framework was implemented with FLAME Zhang & Yang (2025).

Table 3: Zoology-style AR-hit rate (% of tokens) as a function of corpus and tokenizer. A token is an AR hit if it completes a bigram that appeared earlier in the context and occurs $\leq$1250 times per 10B tokens. Documents are chunked at 2048 tokens; bigram counts are taken from held-out data of the same corpus and rescaled to 10B tokens. Pile = `pile-uncopyrighted`.

Corpus	Tokenizer	AR hits (%)	Repeated bigrams (%)
FineWeb-Edu	Llama-2 (32k)	1.7	22.70
	GPT-2 BPE (50k)	2.39	19.88
Pile	Llama-2 (32k)	3.8	40.84
	GPT-2 BPE (50k)	4.31	43.65
Zoology (Arora et al., 2023), Pile		6.4	–

To run the evaluations we use the (Gao et al., 2024) with minor modifications.

C Data and Tokenizer Impact on Zoology analysis

Reusing the exact Zoology method we get far fewer AR hits than they report. The original study found 6.4 % tokens considered as AR-hits. The difference depends on both datasets and tokenizer. The original study used the Pile (Gao et al., 2020) has since become unavailable due to copyright issues on and as such we can only infer the difference from the uncopyrighted data and the documentation of the original paper. About 25 % of training data was removed specifically, Books3, BookCorpus2, OpenSubtitles, YouTube Subtitles, and OpenWebText2.

We find that the untokenized Pile contains roughly twice as many bigrams as FineWeb-Edu. Computing rarity for these, this results in about twice as many AR-hits. As we do not have access to the copyrighted Pile, our statistics for the dataset differs from the original paper which we speculate accounts for the remaining difference.

Additionally we use the Llama-2 tokenizer, while the zoology (Arora et al., 2023) study which used the GPT2BPETokenizer (Radford et al., 2019) which produces a slightly higher ratio of AR-hits. However, we still do not reach the same percentage of AR-hits, which we attribute to the different thresholds for rarity computing statistics for the uncopyrighted Pile leads to.

D Examples cues from the recall suite

To test whether the models solve the task through in-context retrieval rather than memorization, we run two ablations on softmax attention and KDA (Table 4), for both the cloze and the prefix cue. In *No ctx*, the context is removed and the model sees only the generation prompt, so it must rely on the prompt and on knowledge stored in its weights. In *Sub.*, the answer in the context is replaced by a random word of similar length drawn from `wonderwords`; we then measure how often the model outputs the substituted word (Ctx) or the original answer (Mem.).

Removing the context lowers accuracy on every dataset, for both cue types and both architectures, so the models have not simply memorized the answers. The size of the drop depends on the dataset. TQA remains the easiest to answer from memory ($\approx$14–15% for both cues). On DROP, memory-based recall is only non-trivial with the prefix cue ($\approx$20% vs. $\approx$2% for cloze). In all cases, accuracy with context is far higher.

Substitution separates the two cue types. With the prefix cue, both models mostly copy the substituted word from the context, often close to their original accuracy (e.g., TQA: 91.3 vs. 93.0 for softmax attention). They rarely output the original answer, and do so less often than without any context. This suggests the context overrides stored knowledge. With the cloze cue, copying drops sharply, and the rate of recovering the original answer roughly matches the No ctx rate. Here the models fall back on memory rather than on the context.

Table 4: Recall under context ablations for softmax attention and KDA (340M). **Orig.**: accuracy with the full context. **No ctx**: the context is removed and only the generation prompt is kept. **Sub.**: the answer in the context is replaced by a random word of similar length; we report how often the model outputs the *substituted* word (Ctx) or the *original* answer (Mem.). **Cloze** uses the standard question cue; **Prefix** repeats the exact prefix preceding the answer in the context.

		Softmax attention				KDA			
				Sub.				Sub.	
Dataset	Cue	Orig.	No ctx	Ctx	Mem.	Orig.	No ctx	Ctx	Mem.
NQ	Cloze	22.6	4.7	8.8	3.8	15.1	5.0	2.7	5.3
	Prefix	70.8	3.8	64.0	1.0	27.8	4.4	18.9	2.7
TQA	Cloze	44.5	13.6	16.6	13.7	43.7	15.3	15.1	14.7
	Prefix	93.0	14.5	91.3	2.4	84.8	15.4	81.1	3.2
DROP	Cloze	19.7	2.4	7.1	4.6	21.7	1.6	6.5	6.6
	Prefix	87.4	19.8	77.2	5.5	75.7	19.5	54.7	8.8
SWDE	Cloze	54.8	3.8	37.0	3.6	29.6	3.7	11.2	3.7
	Prefix	70.1	3.2	66.3	1.6	42.8	4.7	32.6	2.7
FDA	Cloze	52.9	0.0	51.8	0.0	9.5	0.0	5.7	0.1
	Prefix	75.2	0.0	80.6	0.1	11.5	0.0	7.0	0.3

E Complete LM Eval Results

Table 5: Non-NIAH evaluation results with standard training. Base uses the original evaluation prompt; Prefix adds the four-word prefix. Higher accuracy is better for document extraction and QA; lower word perplexity is better for language modeling. "–" indicates results not evaluated.

		Document extraction		In-context QA				Language modeling	
Model	Prompt	SWDE	FDA	SQuAD	TQA	NQ	DROP	LMB	Wiki
Softmax	Base	53.7	50.5	28.4	50.2	19.4	19.1	35.0	26.2
	Prefix	65.8	69.5	74.0	84.0	48.9	67.7	–	–
DN	Base	20.9	8.2	29.1	45.0	11.9	19.3	37.8	27.9
	Prefix	26.6	11.9	58.5	70.4	17.5	51.3	–	–
GLA	Base	20.0	4.2	31.2	47.5	13.4	18.0	30.5	27.7
	Prefix	20.9	5.1	51.5	65.0	14.7	50.4	–	–
GDN	Base	21.9	7.9	31.9	46.1	14.2	17.8	33.1	27.2
	Prefix	31.7	10.0	59.5	70.0	19.2	50.4	–	–
KDA	Base	29.4	12.6	34.8	49.8	14.2	21.3	29.7	25.9
	Prefix	36.7	14.3	64.3	70.3	18.8	57.1	–	–
Sequence length		1,302	2,271	306	346	1,832	309	87	5,195

Table 6: Non-NIAH evaluation results with repeat training. Base uses the original evaluation prompt; Prefix adds the four-word prefix. Higher accuracy is better for document extraction and QA; lower word perplexity is better for language modeling. "–" indicates results not evaluated.

		Document extraction		In-context QA				Language modeling	
Model	Prompt	SWDE	FDA	SQuAD	TQA	NQ	DROP	LMB	Wiki
Softmax	Base	53.6	62.7	25.6	50.2	19.2	18.4	36.4	26.3
	Prefix	71.7	84.2	81.3	87.6	63.7	80.2	–	–
DN	Base	27.1	22.6	31.7	45.2	12.7	18.1	38.6	28.8
	Prefix	45.4	37.3	72.3	78.8	30.0	67.1	–	–
GLA	Base	28.7	18.9	33.6	47.9	15.2	17.9	32.2	28.5
	Prefix	38.1	26.1	69.1	75.5	25.2	63.4	–	–
GDN	Base	30.0	23.5	33.0	46.5	15.1	18.1	33.4	28.0
	Prefix	47.9	41.2	75.3	80.1	30.9	69.1	–	–
KDA	Base	34.7	27.4	37.3	48.9	16.5	20.1	29.8	26.5
	Prefix	52.0	44.3	75.9	78.2	32.2	69.8	–	–
Sequence length		1,302	2,271	306	346	1,832	309	87	5,195

F MQAR DETAILS

The synthetic MQAR task forms a sequence by sampling N distinct keys without replacement, pairing each with a value, and placing the resulting pairs consecutively at the start of the sequence. The remainder of the sequence is filled with a designated noise token, and a subset of these positions is replaced by queries, each a repetition of one of the keys seen in the prefix. Writing keys as lowercase letters, values as digits, and the noise token as `0`, a sequence takes the form

```
input:   a 3 b 7 c 1 0 0 0 b 0 0 a 0 0 c 0
target:  - - - - - - - - - - 7 - - 3 - - 1 -
```

where − denotes a masked position. The name multi-query refers to the several lookups occurring within a single sequence, which prevents the model from succeeding by retaining only a single retrieved item. Difficulty is controlled by the sequence length, the number of key–value pairs and the vocabulary size.

G AR-HITS-REPEAT

The effect of repeating in context improves results at AR-hits, particularly at a recall distances greater than 512 where the gap on average shrinks by about 28 %. Shorter distances than that don't shift noticeably, as the gap to softmax attention was rather small and the average distance between the original and repeat is usually greater than that.

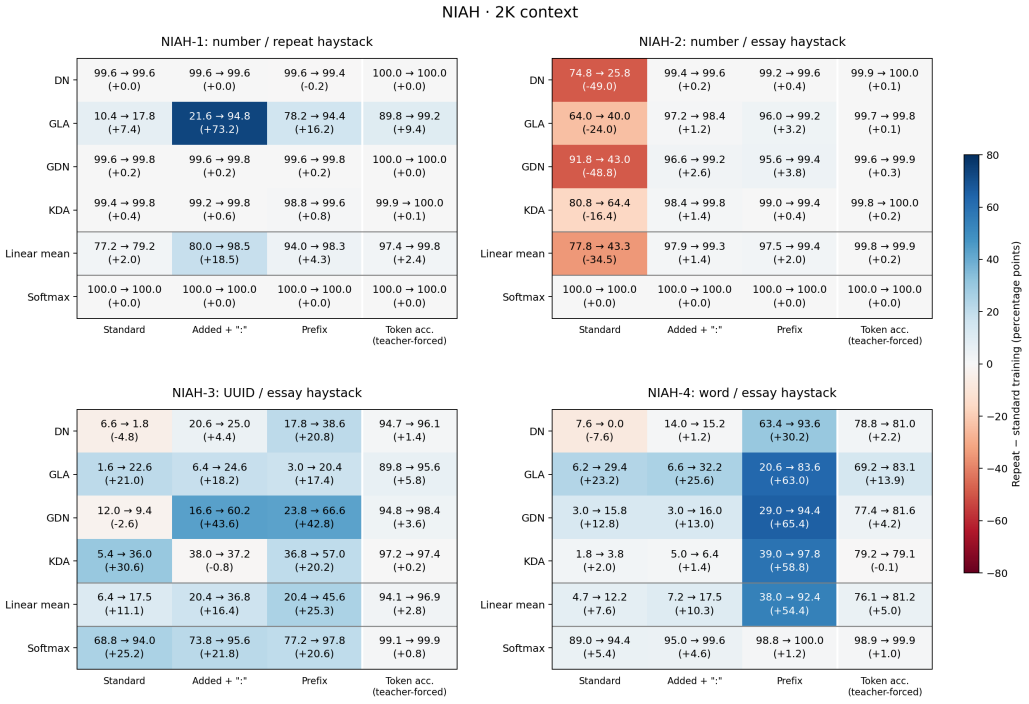
Group	Number of tokens	DN	GLA	GDN	KDA
Gap to softmax, standard run					
AR-hits, 1–511 back	226,922	+0.122	+0.153	+0.094	+0.072
AR-hits, 512+ back	46,148	+0.462	+0.559	+0.391	+0.334
Rare tokens					
New prefix, 1–511 back	16,158	+0.135	+0.071	+0.042	−0.106
New prefix, 512+ back	3,541	+0.148	+0.197	+0.135	+0.019
Same prefix, 1–511 back	12,472	+0.219	+0.235	+0.176	+0.127
Same prefix, 512+ back	3,158	+0.657	+0.850	+0.628	+0.521
Gap to softmax, Repeat run					
AR-hits, 1–511 back	226,922	+0.132	+0.153	+0.092	+0.074
AR-hits, 512+ back	46,148	+0.352	+0.372	+0.295	+0.229
Rare tokens					
New prefix, 1–511 back	16,158	+0.190	+0.145	+0.107	−0.067
New prefix, 512+ back	3,541	+0.204	+0.254	+0.200	+0.059
Same prefix, 1–511 back	12,472	+0.230	+0.235	+0.176	+0.146
Same prefix, 512+ back	3,158	+0.463	+0.551	+0.411	+0.342
Effect of Repeat (Repeat − standard)					
AR-hits, 1–511 back	226,922	+0.010	+0.000	−0.002	+0.001
AR-hits, 512+ back	46,148	−0.110	−0.187	−0.096	−0.105
Rare tokens					
New prefix, 1–511 back	16,158	+0.054	+0.074	+0.065	+0.039
New prefix, 512+ back	3,541	+0.056	+0.057	+0.065	+0.040
Same prefix, 1–511 back	12,472	+0.010	+0.000	+0.000	+0.018
Same prefix, 512+ back	3,158	−0.194	−0.299	−0.217	−0.179

Table 7: Effect of the splicemix variants on both recall tasks, held-out FineWeb-Edu, split at 512 tokens. The top and middle panels give each mixer's loss gap to softmax attention in nats for standard and repeat training, respectively. The bottom panel gives the change in that gap (repeat minus standard) in nats; negative values indicate a narrower gap.

H Complete Single NIAH Results

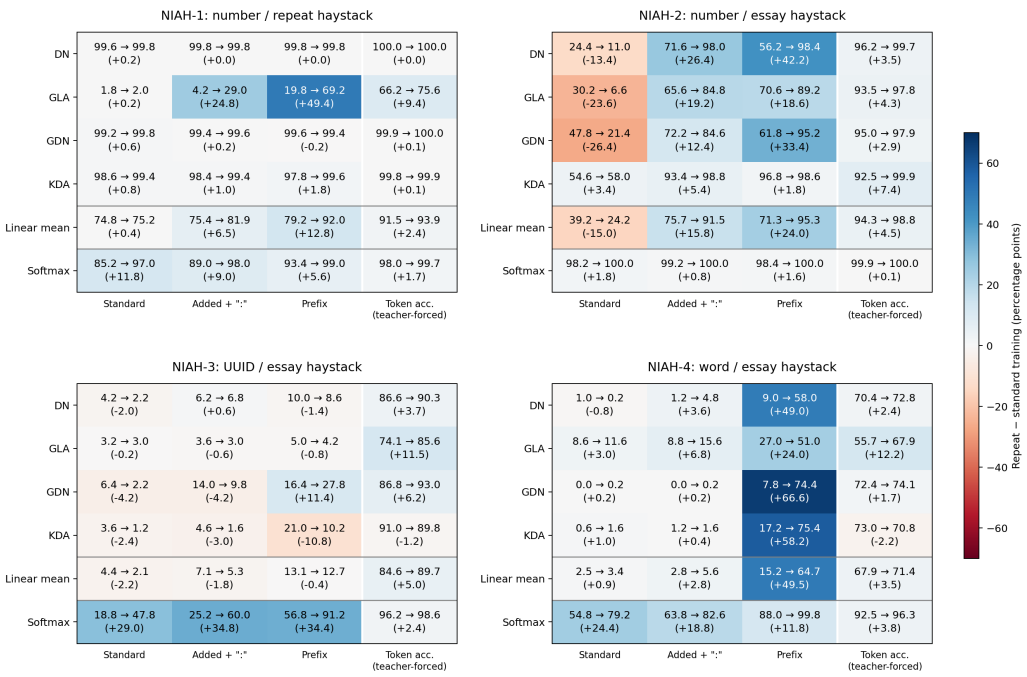
The trend across sequence lengths remains approximately the same, with improvements on the modified generation prompts, and decreases on standard prompts for most linear attention models.

A noteworthy phenomenon is the exceptionally strong performance on architectures using delta-rule on NIAH-1. As the delta rule erases previous associations bound to the same key, repeated inputs of the same tokens are trivially ignored, which means the needle is well preserved even at a context length of 8192 tokens, while GLA without the Delta Rule struggles.



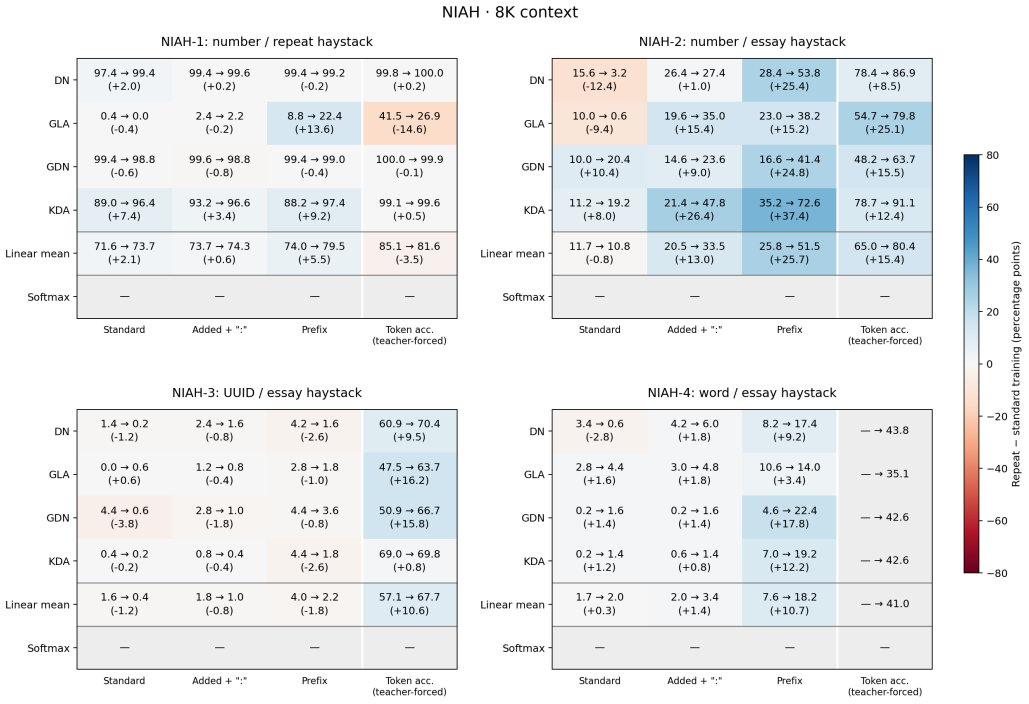
2K only. Each cell: standard-training score → repeat-training score; gain in parentheses. Color encodes gain.
First three columns: generated contains accuracy (%). Last column: teacher-forced token accuracy (%) under the Added + ":" prompt.
Linear mean: equal weight for DN, GLA, GDN, and KDA. Shared color scale across 2K and 8K.

Figure 9: Complete S-NIAH results at 2k context length



4K only. Each cell: standard-training score → repeat-training score; gain in parentheses. Color encodes gain.
First three columns: generated contains accuracy (%). Last column: teacher-forced token accuracy (%) under the Added + ":" prompt.

Figure 10: Complete S-NIAH results at 4k context length



8K only. Each cell: standard-training score → repeat-training score; gain in parentheses. Color encodes gain.
First three columns: generated contains accuracy (%). Last column: teacher-forced token accuracy (%) under the Added + ":" prompt.
Linear mean: equal weight for DN, GLA, GDN, and KDA. Shared color scale across 2K and 8K.
Gray: change unavailable. Softmax was not evaluated at 8K; NIAH-4 standard-training token accuracy was not evaluated.

Figure 11: Complete S-NIAH results at 8k context length