What Matters in Linearizing Language Models? A Comparative Study of Architecture, Scale, and Task Adaptation

Anonymous ACL submission

Abstract

Linearization has emerged as a strategy for developing efficient language models (LMs). Starting from an existing Transformer-based LM, linearization replaces the attention component with computationally efficient subquadratic *token mixers*. However, as an increasing number of mixers are proposed, it remains unclear which inductive biases are best suited to inherit the original Transformer's capabilities. Furthermore, it is unknown how linearization is affected by parameter and token budget scaling. To address these questions, we propose a unified setup to compare seven representative architectures, including xLSTM, GLA, and Gated DeltaNet. Our findings reveal that performance hierarchies remain stable from 140M to 1.7B parameters, with error-correcting update rules demonstrating superior scaling exponents. We show that performance gaps are established early and persist through asymptotic maturity at 10B tokens, suggesting that state resolution is a more fundamental bottleneck than the distillation budget. Finally, while most models adapt to instruction tuning, only gated delta-rule formulations maintain the precision necessary for long-context retrieval, whereas additive models suffer from irreversible state saturation. Finally, we show that a cheap mixer-only probe predicts these downstream outcomes before a full run, enabling architecture screening at a small fraction of the cost. These results suggest that for successful linearization, architectural inductive biases remain the primary constraint that cannot be overcome by simply scaling training compute.

1 Introduction

Recent research has increasingly focused on *linearization* as a strategy for developing efficient language models (Bick et al., 2024; Mercat et al., 2024a; Zhang et al., 2024a; Wolf et al., 2020). This process involves taking a pretrained Transformer and converting it into a subquadratic student model by replacing the original $O(L^2)$ self-attention layers with more efficient *token mixers*, such as linear attention or gated recurrences. While the student typically retains the teacher's pretrained feed-forward weights, it must undergo *knowledge distillation* to adapt the new token mixer to the teacher's learned representations. The goal is to retain the performance of large-scale Transformers while gaining the linear-time inference and constant-memory overhead of subquadratic architectures.

However, as the variety of subquadratic token mixers has increased, it has become difficult to determine which specific inductive biases are most effective for successful linearization. Current literature often reports results under heterogeneous conditions—varying in model size, training budget, and distillation objectives—which makes it challenging to isolate whether observed gains stem from architectural design (such as the use of gating, decay, or the delta rule) or from the specific training recipe.

Beyond the choice of token mixer, it remains unclear how linearized students scale with parameter count and distillation budget, whether architectural advantages persist from millions to billions of parameters, and to what extent these students can undergo instruction tuning or manage long-context retrieval after distillation.

A unified evaluation framework. We address these gaps by conducting a systematic empirical study of seven representative subquadratic architectures, including xLSTM (Beck et al., 2024), DeltaNet (Yang et al., 2025b), Gated DeltaNet (Yang et al., 2025a), GLA (Yang et al., 2024), RetNet (Sun et al., 2023), and Kimi (Team et al., 2025).

To isolate the token mixer as the primary variable, we develop a three-stage distillation pipeline that aligns attention-outputs and hidden-states while holding the teacher model, data distribution, and training recipe constant, providing each student

with a dense, architecture-agnostic supervision signal for a rigorous head-to-head comparison of their state-update mechanisms.

Our study yields three primary insights into the linearization of language models:

- We scale student models from 140M to 1.7B parameters and observe that architectural performance hierarchies remain remarkably stable across scales. While gated architectures consistently outperform ungated baselines, we identify a crossover at the 1.7B scale where error-correcting update rules begin to demonstrate superior scaling exponents compared to purely additive alternatives.

- We evaluate the maturity of these architectures by extending distillation significantly beyond standard linearization recipes. Our findings indicate that relative performance gaps are established early in the training process and persist even as models approach their asymptotic limits, suggesting that the inherent state resolution of the token mixer is a more fundamental bottleneck than the distillation token budget.

- We evaluate the viability of linearized models for downstream and long-context tasks by performing instruction tuning on the distilled students. While we find that subquadratic models successfully adapt to instruction following, this process reveals a mixed picture regarding state resolution; specifically, most architectures suffer from a collapse in long-context retrieval, with only gated delta-rule formulations maintaining the precision required for long-range associative recall.

- We introduce a cheap mixer-only probe that predicts a token mixer's downstream recovery before a full distillation run (Pearson $r = -0.80$), enabling architecture screening at a small fraction of the cost of full distillation.

Taken together, these findings establish a clear hierarchy for subquadratic design and demonstrate that architectural choices such as error-correcting update rules are more decisive for successful linearization than simply increasing the distillation token budget.

To facilitate further research into efficient architectures and ensure the reproducibility of our findings, we publicly release all base, instruction-tuned, and instruction-distilled models[1].

2 Preliminaries and Related Work

2.1 Subquadratic Token Mixers

Subquadratic token mixers replace quadratic self-attention with mechanisms that compress history into a compact state that is incrementally updated as tokens arrive, enabling linear-time inference while shifting long-range dependency modeling onto the state update rule. Most architectures differ along one central axis—how they retain, overwrite, or forget information over time—and we group the models in this study by this principle.

State accumulation via linear attention. Linear attention approximates softmax attention by reformulating attention as a sum of key–value outer products stored in a state matrix. The state is updated additively at each step and queried by the current token, yielding linear complexity in sequence length but removing the explicit normalization and token-level selectivity of softmax attention. A known limitation of this formulation is the lack of explicit forgetting, which can cause the state to become dominated by stale or irrelevant information. Several variants (Aksenov et al., 2024; Zhang et al., 2024b) address this by improving the feature map or kernel, but the underlying update remains purely additive.

Gated and decay-based state updates. An alternative design introduces explicit control over memory retention through decay or gating. Rather than accumulating all past information, these models modulate the previous state before adding new content. RetNet (Sun et al., 2023) applies a uniform decay to the entire state, biasing the model toward recent context, Gated Linear Attention (GLA) (Yang et al., 2024) refines this idea by applying channel-wise decay, allowing different components of the state to be forgotten at different rates. More expressive variants introduce input and forget gates, drawing inspiration from recurrent neural networks. Models such as xLSTM (Beck et al., 2024) update their state using multiplicative gates that regulate both how much new information is written and how much of the previous state is preserved. This enables selective retention and controlled overwriting.

[1] Annonymized for submission.

Architecture	Update Type	Input Dep.	State Update Rule	Granularity
Outer Product Based Matrix-Valued State (Additive)				
Linear Attn	Additive	-	$S_t = S_{t-1} + v_t^\top k_t$	Ungated
RetNet	Decay	No	$S_t = \gamma S_{t-1} + v_t k_t^\top$	Channel-wise
GLA	Gated Decay	Yes	$S_t = \mathrm{Diag}(\alpha_t) S_{t-1} + v_t k_t^\top$	Channel-wise (diag.)
mLSTM (xLSTM)	Gated Decay	Yes	$S_t = f_t S_{t-1} + i_t v_t k_t^\top$	Head-wise Scalar
Delta-Rule (Overwriting)				
DeltaNet	Error-Correction	Yes	$S_t = S_{t-1}(I - \beta_t k_t k_t^\top) + \beta_t v_t k_t^\top$	Scalar Overwrite
Gated DeltaNet	Error-Correction	Yes	$S_t = \alpha_t S_{t-1}(I - \beta_t k_t k_t^\top) + \beta_t v_t k_t^\top$	Head-wise Scalar
Kimi (KDA)	Error-Correction	Yes	$S_t = \mathrm{Diag}(\alpha_t) S_{t-1}(I - \beta_t k_t k_t^\top) + \beta_t v_t k_t^\top$	Channel-wise (diag.)
State-Space Update				
Mamba2	Gated Decay	Yes	$S_t = \alpha_t S_{t-1} + v_t k_t^\top$	Head-wise Scalar

Table 1: Taxonomy of subquadratic token mixers. We distinguish between additive accumulation, gated decay, and content-aware overwriting (Delta-rule). Formulas illustrate how the state $S_t \in \mathbb{R}^{d_k \times d_v}$ is updated per timestep.

Delta-rule and content-based overwriting. Delta-rule models (Yang et al., 2025b) adopt a content-aware update mechanism that partially overwrites state components aligned with the current input. This induces a form of last-write-wins memory, where new evidence replaces older, related information. Gated variants, such as Gated DeltaNet (Yang et al., 2025a) and Kimi (Team et al., 2025), combine this overwrite rule with decay terms, providing fine-grained control over memory retention. Rather than approximating attention kernels, these models replace token-level selection with structured state updates.

Structured state spaces and duality. Selective State Space Models (SSMs) approach memory retention through input-dependent state transitions that filter or retain context based on the current token. Mamba-2 (Dao and Gu, 2024) bridges this paradigm with linear attention via Structured State Space Duality, showing that its data-dependent decay is equivalent to structured masked linear attention and can thus be computed with attention-like hardware efficiency.

2.2 Linearizing Pretrained Transformers

Rather than training subquadratic language models from scratch, a growing body of work studies how to *linearize* pretrained Transformers by replacing softmax attention with more efficient token mixers and transferring knowledge via distillation (Hinton et al., 2015). Early approaches focused on fine-tuning pretrained models with recurrent or decaying attention mechanisms (Kasai et al., 2021; Mao, 2022), demonstrating that pretrained representations can be partially preserved despite architectural changes.

More recent work has proposed increasingly structured linearization pipelines. SUPRA (Mercat et al., 2024b) introduces a scalable uptraining framework that converts pretrained Transformers into recurrent architectures through progressive adaptation. LoLCATs (Zhang et al., 2024a) combines low-rank adaptation (Hu et al., 2021) with attention transfer to approximate softmax attention efficiently. MOHAWK (Bick et al., 2024) proposes a staged distillation procedure that aligns materialized attention maps between teacher and student, enabling the transfer of quadratic attention behavior into subquadratic models.

Subsequent extensions integrate architectural reuse and instruction tuning. Mamba-LLaMA (Wang et al., 2025b) applies progressive distillation to Mamba-based students, LIGER (Lan et al., 2025) constructs gating modules with additional mechanisms such as sliding-window attention, and Yueyu et al. (2025) linearize Qwen-2.5 using RWKV-style blocks with hidden-state alignment. LIZARA (Nguyen et al., 2025) further augments subquadratic token mixers with adaptive memory modules.

In contrast to prior work that typically targets a single student architecture or emphasizes attention-map reconstruction, our study adopts an *architecture-agnostic* linearization framework based on attention-output and hidden-state alignment. This enables direct comparison of diverse

subquadratic token mixers under identical training conditions, isolating the effects of state-update mechanisms, gating, and scaling behavior. As Mamba-style (Dao and Gu, 2024) models have already been extensively explored, we focus on alternative architectures spanning additive, gated, and delta-rule–based update families.

3 Linearization Framework

We train subquadratic students via a three-stage distillation pipeline that progressively aligns the student with a pretrained Transformer teacher, stabilizing optimization, isolating the token mixer as the primary source of variation, and enabling fair comparison across architectures. Students inherit all teacher parameters by copying; newly introduced parameters are initialized from scratch.

In contrast to prior approaches such as MOHAWK, which align the teacher's materialized attention maps with the student's, we align the full attention output produced by the token mixer. Attention-map alignment is tied to attention-specific inductive biases and primarily constrains query–key interactions; aligning the complete attention output instead provides a denser, model-agnostic supervision signal that remains well-defined across linear attention, gated recurrent, and state-update–based mixers.

Stage 1: Token-mixer (attention-output) alignment. In the first stage, we align the student's token mixer with the teacher's attention output. For each aligned layer ℓ, we minimize a mean squared error loss

$$\mathcal{L}_{\mathrm{AO}}^{\ell} = \left\| a_S^{\ell}(x) - a_T^{\ell}(x) \right\|_2^2 . \tag{1}$$

The overall attention-output alignment objective is

$$\mathcal{L}_{\mathrm{AO}} = \frac{1}{|\mathcal{L}_1|} \sum_{\ell \in \mathcal{L}_1} \mathcal{L}_{\mathrm{AO}}^{\ell}, \tag{2}$$

where $\mathcal{L}_1$ denotes the set of aligned layers. This stage directly supervises the complete token-mixing pathway, including value projections, output projections, and gating mechanisms.

Stage 2: Hidden-state alignment. In the second stage, we align intermediate hidden representations of teacher and student. For each selected layer ℓ, we minimize the mean squared ℓ_2 distance

$$\mathcal{L}_{\mathrm{H2H}}^{\ell} = \frac{1}{T} \sum_{t=1}^{T} \left\| h_S^{\ell}(x)_t - h_T^{\ell}(x)_t \right\|_2 . \tag{3}$$

Aggregating over a set of layers $\mathcal{L}_2$, the hidden-state alignment loss is

$$\mathcal{L}_{\mathrm{H2H}} = \frac{1}{|\mathcal{L}_2|} \sum_{\ell \in \mathcal{L}_2} \mathcal{L}_{\mathrm{H2H}}^{\ell}. \tag{4}$$

Stage 3: End-to-end knowledge distillation. In the final stage, we train the student end-to-end using standard knowledge distillation. The objective combines next-token prediction with distribution matching:

$$\mathcal{L}_{\mathrm{KD}} = \mathcal{L}_{\mathrm{CE}} + \lambda_{\mathrm{KL}} \mathcal{L}_{\mathrm{KL}}, \tag{5}$$

where the distillation loss is defined as

$$\mathcal{L}_{\mathrm{KL}} = \frac{1}{T} \sum_{t=1}^{T} \mathrm{KL}\big(p_T(\cdot \mid x_{\leq t}) \parallel p_S(\cdot \mid x_{\leq t})\big). \tag{6}$$

The three objectives are applied sequentially. This staged optimization improves stability and prevents interference between alignment signals, enabling controlled comparison of token mixers under a unified protocol.

4 Experiments

We select seven representative subquadratic token mixers to enable controlled comparisons along two key architectural axes: the update rule and the presence and granularity of gating. Specifically, we consider two dominant update families - *outer-product additive updates* and *delta-rule–based updates* - and, within each family, architectures with head-wise scalar gating as well as channel-wise gating.

Within the outer-product family, we include Linear Attention as an ungated baseline, GLA (Yang et al., 2024) as a channel-wise gated variant, and xLSTM (Beck et al., 2024) as a head-wise scalar gated model with input-dependent gating. Within the delta-rule family, we include DeltaNet (Yang et al., 2025b) as a channel-wise gated model, Gated DeltaNet (Yang et al., 2025a) as a head-wise scalar gated variant and Kimi (Team et al., 2025) as a channel-wise gated variant. In addition, we include RetNet (Sun et al., 2023), which employs a head-wise scalar but input-independent retention gate and thus occupies a distinct position between ungated and input-dependent gated models.

This selection spans ungated, input-independent gated, and input-dependent gated architectures across both update families, enabling systematic analysis of which inductive biases most affect distillation and instruction tuning. For the additive

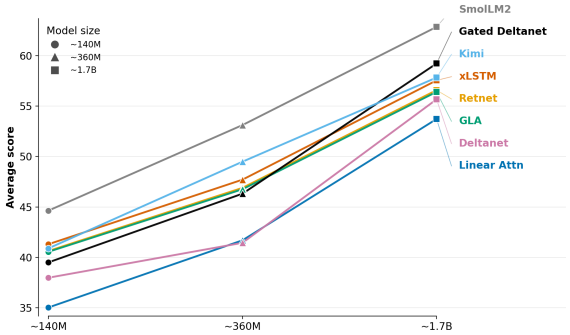

Figure 1: Average evaluation score for all 7 (+ teacher) considered token mixers at three model sizes. All architectures markedly improve at larger model sizes, though some architectures, such as Gated DeltaNet, benefit most from model scaling.

Model	Avg.	Recov.	Rank
Scale: ∼140M Parameters			
Teacher: SmolLM2-135M	*44.6*	*100%*	–
xLSTM	41.3	92.6%	1
Kimi	40.9	91.6%	2
RetNet	40.6	91.0%	3
GLA	40.6	90.9%	4
Gated DeltaNet	39.5	88.5%	5
DeltaNet	38.0	85.1%	6
Linear Attn	35.0	78.5%	7
Scale: ∼360M Parameters			
Teacher: SmolLM2-360M	*53.1*	*100%*	–
Kimi	49.5	93.2%	1
xLSTM	47.7	89.8%	2
RetNet	46.9	88.3%	3
GLA	46.8	88.0%	4
Gated DeltaNet	46.3	87.2%	5
Linear Attn	41.7	78.5%	6
DeltaNet	41.5	78.0%	7
Scale: ∼1.7B Parameters			
Teacher: SmolLM2-1.7B	*62.9*	*100%*	–
Gated DeltaNet	59.2	94.2%	1
Kimi	57.8	92.0%	2
xLSTM	57.6	91.6%	3
RetNet	56.6	90.0%	4
GLA	56.4	89.8%	5
DeltaNet	55.7	88.6%	6
Linear Attn	53.7	85.5%	7

Table 2: Summary of zero-shot downstream performance across scales. Recovery (Recov.) denotes the percentage of teacher performance retained by the linearized student. Models are ranked per scale by average performance.

and decay-based mixers (Linear Attention, GLA, RetNet), we prepend a short convolution to the token-mixing inputs; an ablation across all scales (Appendix B, Table 7) shows it consistently improves downstream performance, so we adopt it by default. Computational efficiency is validated in Appendix G.

4.1 Experiment 1: Comparison Across Model Size

Our first inquiry establishes the relative ranking of these eight subquadratic architectures under a unified framework and investigates whether these performance hierarchies remain stable as model capacity increases.

4.1.1 Experimental Setup

Token budget. We train all architectures on a fixed budget of 3B tokens sampled from FineWeb (Penedo et al., 2024), a budget shown to yield state-of-the-art linearized models (Bick et al., 2024). Text is chunked into fixed 512-token sequences; we select this context length via an ablation over 512, 1024, and 2048 tokens (Appendix C, Table 8), which shows shorter contexts yield consistently better zero-shot performance. Following Bick et al. (2024), we allocate 80M tokens for matrix-mixing distillation and 160M for hidden-state alignment.

Model sizes. We instantiate all seven subquadratic architectures at three scales: 140M, 360M, and 1.7B parameters. As teachers, we use the SmolLM2 family (Allal et al., 2025), a series of Llama-style Transformer models. Each teacher corresponds in parameter count to our student sizes: SmolLM2-135M, SmolLM2-360M, and SmolLM2-1.7B[2].

Evaluation. Student and teacher models are evaluated on seven zero-shot tasks using the LM-Eval-Harness (Gao et al., 2023) and LM-PubQuiz (Ploner et al., 2024) frameworks: LAMBADA (Paperno et al., 2016), WinoGrande (Sakaguchi et al., 2019), ARC-Easy and ARC-Challenge (Clark et al., 2018), PIQA (Bisk et al., 2019), HellaSwag (Zellers et al., 2019) and BEAR (Wiland et al., 2024). LAMBADA results are reported as the mean of its Standard and OpenAI variants.

4.1.2 Results

Our results are summarized in Figure 1 and Table 2. We observe smooth, monotonic scaling across all eight architectures with no divergence or early saturation, confirming that these subquadratic models train stably via distillation across a wide parameter range.

However, scaling efficiency—the ability to close

[2]Note that since specific token mixers utilize different internal layer structures, parameter counts are matched as closely as possible but are not exactly identical across architectures.

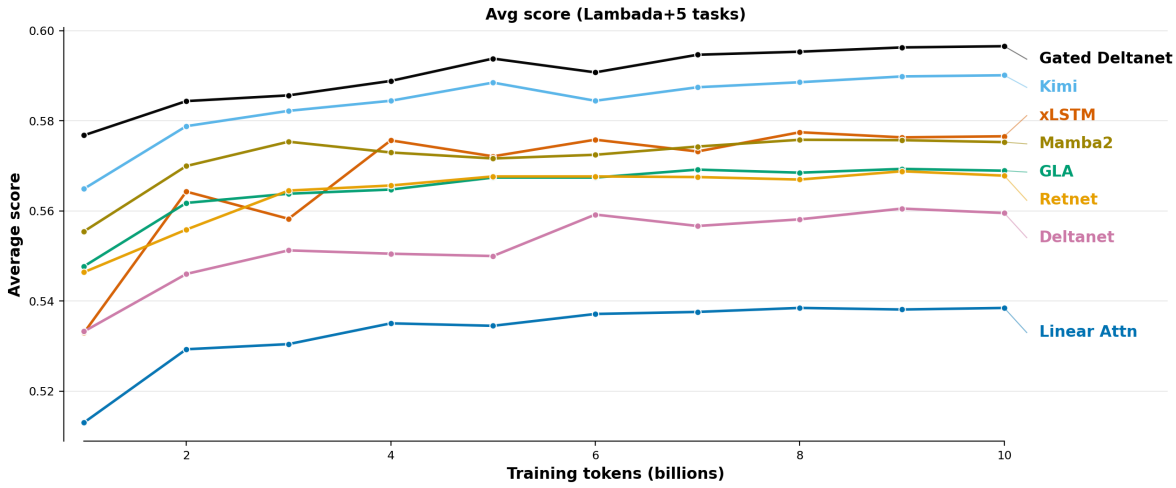

Figure 2: Token scaling behavior during training. We evaluate each 1B tokens of training and report average scores across all benchmarks.

the gap to the teacher—differs markedly between architectures. Gated state-update models define the Pareto frontier, though the leader shifts with scale: xLSTM has the highest recovery at 140M and 360M, while a notable "crossover" occurs at 1.7B, where Gated DeltaNet leads at 94.2% recovery. This suggests error-correction mechanisms such as the Delta-rule exhibit superior scaling behavior as capacity increases.

Among additive and decay-based mixers, GLA and RetNet follow nearly identical trajectories, with RetNet holding a narrow lead. The short convolutions adopted for these architectures (Appendix B) make Linear Attention substantially more competitive at 1.7B, though it remains the weakest student overall. Kimi scales strongly, reaching 92.0% recovery at 1.7B—second only to Gated DeltaNet.

4.2 Experiment 2: Asymptotic Performance and State Resolution

Experiment 1 established scaling behavior relative to parameter count but did not address whether weaker architectures simply need more data to converge. We test this by fixing the size at ∼1.7B parameters and extending training to 10B tokens, observing both the asymptotic knowledge limit (downstream benchmarks) and state resolution (retrieval tasks) of each architecture.

4.2.1 Asymptotic Knowledge Scaling

We train over 10B tokens sampled from FineWeb and use the same token splits for the three linearization phases. We checkpoint the models after each 1B tokens of training. Each checkpoint is evaluated on the same benchmarks as in Experiment 1.

Results. As shown in Figure 2, all architectures benefit from additional training. However, the results confirm that more data does not close the architectural gap.

Gated state-update models (Gated DeltaNet, xLSTM) maintain their lead throughout, plateauing after 6–7B tokens, while ungated models such as RetNet and DeltaNet saturate earlier and at a lower level. The Experiment 1 gaps thus reflect not undertraining but fundamental limits in how token mixers compress information into their recurrent states.

	Context Length			
Model	512	1k	2k	4k
Gated DeltaNet	**71.6**	**64.8**	**46.0**	**15.3**
Kimi	68.7	57.3	29.8	8.4
xLSTM	66.9	53.5	19.1	5.7
DeltaNet	66.6	57.7	8.2	0.0
Mamba2	32.0	6.8	2.4	1.3
GLA	20.2	4.7	1.0	0.5
RetNet	13.5	3.4	0.0	0.0
Linear Attn	0.0	0.0	0.0	0.0

Table 3: Needle-In-A-Haystack (NIAH) retrieval accuracy (%) of distilled models at 10B tokens, averaged over the three S-NIAH subtasks at each context length. Gated DeltaNet retains markedly higher state resolution at extended lengths. Per-subtask results are in Appendix E (Table 10).

4.2.2 Retrieval and State Resolution

A model's ability to accumulate knowledge over 10B tokens is inextricably linked to its ability to

manage its internal state. To probe the "resolution" of these states, we evaluate the 10B-token checkpoints on three "Needle-In-A-Haystack" (NIAH) tasks (Hsieh et al., 2024). In NIAH, models are tasked to retrieve a specific piece of information (the "needle") from a long sequence of tokens.

Results. Table 3 reveals a pervasive collapse in state resolution across most subquadratic architectures as sequence length increases. Linear Attention and RetNet fail almost entirely beyond 512 tokens, while gated models like xLSTM and DeltaNet lose precision dramatically by 2k tokens, dropping to mean retrieval of 19.1% and 8.2%. This suggests their fixed-size state saturates as it accumulates information, lacking the precision for long-range associative recall.

In contrast, the gated delta-rule models preserve usable retrieval at 2k and 4k contexts: Gated DeltaNet leads (46.0% and 15.3%), with Kimi a clear second (29.8% and 8.4% at 2k/4k)—both a substantial margin over the next-best student (xLSTM, 19.1% at 2k). While still short of the near-perfect retrieval of Transformer teachers with explicit KV-caches, this indicates the gated Delta-rule provides a more effective mechanism for targeted state updates.

4.3 Experiment 3: Instruction Adaption: Post-Training vs. Instruction-Aware Distillation

Finally, we examine whether linearized students can inherit instruction-following behaviour, and whether the timing of supervision matters. At the ∼1.7B scale we compare two regimes, both starting from the 10B-token checkpoints of Experiment 2: (i) **post-training** SFT on *SmolTalk* (1.1M pairs, ∼1B tokens; no teacher), using the recipe of *SmolLM2-1.7B-Instruct* (Allal et al., 2025); and (ii) **instruction-aware distillation** from *SmolLM2-1.7B-Instruct* (Stage 1–2 alignment then a Stage 3 distillation loss on *SmolTalk*). All models are evaluated on zero-shot, retrieval (NIAH), and instruction-following/reasoning benchmarks — IFEval (Zhou et al., 2023), GSM8K (Cobbe et al., 2021), SWDE, and LongBench (Bai et al., 2024)[3]. Table 4 summarizes the headline metrics; full per-task results (including SWDE, LongBench, and the Mamba2 reference) are in Appendix F (Tables 11, 12, 13). The headline carries over from the base models: instruction adaptation *amplifies* rather than bridges architectural gaps. Post-training reliably improves every student but benefits the already-strong gated state-update models (xLSTM, Gated DeltaNet) most, while instruction-aware distillation is inconsistent and further *degrades* long-context retrieval. Across both regimes, only the gated delta-rule retains usable long-range retrieval; additive and simply-gated mixers collapse, mirroring their pre-adaptation NIAH failures. Instruction supervision thus does not overcome the state-resolution limits set during linearization.

[3]We use five subsets of LongBench: WikiMQA, MultiFieldQA, NarrativeQA, TREC, and TriviaQA.

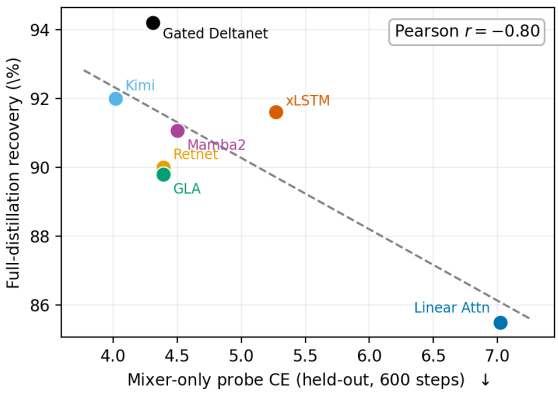

Figure 3: The probe as a predictor. Held-out probe CE (mixer-only, 600 steps) tracks full-distillation recovery across token mixers: lower probe CE predicts higher downstream recovery (cf. Table 5).

4.4 Experiment 4: Predicting Linearization Quality

The preceding experiments require a full multi-stage distillation run to rank an architecture. We close with a lightweight diagnostic that **predicts** a token mixer's downstream recovery from a short, inexpensive probe.

A mixer-only attention-transfer probe. The probe isolates the token mixer from the rest of the network. Starting from the teacher, we replace attention with the candidate mixer, *freeze* the inherited feed-forward, embedding, and output weights, and train only the new mixer parameters under the Stage 3 objective (Section 3) for a short budget (600 steps). Because everything except the mixer is held fixed, the held-out distillation loss measures how well the mixer alone reproduces the teacher's computation, at a fraction of a percent of the full token budget.

The probe predicts recovery. Table 5 and Figure 3 show that the probe's held-out loss ranks architectures consistently with their full downstream recov-

	Regime A: Post-Training SFT				Regime B: Instruction-Aware Distillation			
MODEL	LLM	NIAH	IFEVAL	GSM8K	LLM	NIAH	IFEVAL	GSM8K
Teacher	*70.2*	*99.3*	*56.7*	*42.1*	*70.6*	*98.1*	*60.2*	*49.5*
Gated DeltaNet	**61.1**	**57.3**	40.9	8.3	**55.4**	**47.3**	**47.1**	**20.0**
xLSTM	60.5	47.2	**45.1**	**10.2**	51.6	10.4	41.7	13.5
GLA	56.6	10.8	40.9	8.3	49.1	4.4	29.5	5.8
DeltaNet	56.3	39.4	36.2	9.7	55.3	40.1	42.2	13.1
RetNet	56.3	6.3	32.6	6.4	53.7	10.6	31.1	2.8
Kimi	56.7	45.0	38.0	8.5	50.0	41.2	42.0	15.0
Linear Attn	44.7	0.0	24.5	1.6	31.3	0.0	22.8	0.0

Table 4: Instruction adaptation at $\sim$1.7B (condensed). LLM is the mean zero-shot score and NIAH the mean retrieval accuracy; best student per column within each regime in bold. Full metrics (SWDE, LongBench, Mamba2, per-task breakdowns) are in Appendix F.

MODEL	PROBE CE ↓	RECOV. (%)
Gated DeltaNet	4.31	94.2
RetNet	4.39	90.0
GLA	4.39	89.8
Mamba2	4.50	91.1
xLSTM	5.27	91.6
Linear Attn	7.02	85.5
Kimi	**4.02**	92.0

Table 5: The mixer-only probe predicts downstream recovery (Pearson $r = -0.80$ between held-out probe CE and full-distillation recovery, $n = 7$): lower probe CE corresponds to higher recovery across token mixers.

ery (Pearson $r = -0.80$): the delta-rule and decay-based mixers that lead at 1.7B (Gated DeltaNet, RetNet, GLA) also achieve the lowest probe loss, while Linear Attention is correctly flagged as the weakest student. The probe is thus a useful screen at a small fraction of the cost of a full run. Its one notable miss is xLSTM, which the mixer-only probe under-rates relative to its strong end-to-end recovery—likely because xLSTM recovers capabilities through the inherited network rather than the mixer in isolation—so we treat the probe as a strong-but-imperfect screen rather than an exact predictor.

The figure also hints at a gating trade-off: channel-wise (granular) gates such as Kimi's optimize the probe objective best (lowest CE), yet head-wise scalar gates—most clearly Gated DeltaNet—tend to recover better downstream, suggesting that finer-grained gating eases optimization more than it improves generalization. A plausible explanation is that per-channel gates add degrees of freedom that let the mixer match the teacher's attention output more easily during distillation—lowering probe loss—without a commensurate gain in the downstream behaviour that full distillation ultimately rewards. In effect, the finer gate optimizes the distillation proxy more than the end task it is meant to approximate. A practical consequence is that probe loss is best read as a positive screen rather than a precise ranking: a very low probe loss from a granular-gated mixer, as for Kimi, does not by itself guarantee the highest recovery.

5 Conclusion

We present a systematic empirical evaluation of subquadratic language models distilled from Transformer teachers, comparing diverse token mixers under a unified setup that isolates architectural design from confounds such as data, optimization, and post-training. Architectural choices induce persistent performance gaps: delta-rule and gated recurrent mechanisms consistently outperform outer-product–based linear attention, and head-wise or scalar gating enables more effective memory control and long-range retention than channel-wise diagonal gating.

Beyond this comparison, we contribute a lightweight predictive probe (Section 4.4): a short, mixer-only run whose held-out loss anticipates a token mixer's full-distillation recovery (Pearson $r = -0.80$), letting architectures be screened at a fraction of the cost of a full run. The same analysis surfaces a trade-off in gate design—channel-wise gates optimize the distillation objective most readily, whereas head-wise scalar gates tend to generalize better downstream—a gap between optimization ease and downstream quality that future linearization recipes might exploit. Taken together, our results offer both a clear empirical ranking of subquadratic token mixers and a practical tool for navigating the design space before committing to expensive distillation runs.

Limitations

While our study provides a comprehensive empirical comparison of subquadratic language models, several limitations should be noted.
First, we restrict our analysis to a **fixed Transformer teacher family**. Although this enables controlled comparisons across student architectures, different teacher models or pretraining objectives may induce different transfer dynamics.

Second, we do not investigate **hybrid architectures** (intra- or inter-layer) that combine multiple token mixers within or across layers. Such designs would substantially expand the design space beyond the scope of this work. Moreover, while hybrid token-mixer architectures are an active area of research, prior work suggests that hybrid performance cannot be predicted from standalone components (Wang et al., 2025a), motivating our focus on isolating single-mixer architectural effects.

Finally, our analysis focuses primarily on downstream performance and scaling behavior. We do not conduct a detailed **mechanistic or qualitative analysis** of internal dynamics, such as gating behavior, which could provide additional insight into why certain architectures align more effectively than others.

References

Yaroslav Aksenov, Nikita Balagansky, Sofia Maria Lo Cicero Vaina, Boris Shaposhnikov, Alexey Gorbatovski, and Daniil Gavrilov. 2024. Linear transformers with learnable kernel functions are better in-context models. *Preprint*, arXiv:2402.10644.

Loubna Ben Allal, Anton Lozhkov, Elie Bakouch, Gabriel Martín Blázquez, Guilherme Penedo, Lewis Tunstall, Andrés Marafioti, Hynek Kydlíček, Agustín Piqueres Lajarín, Vaibhav Srivastav, Joshua Lochner, Caleb Fahlgren, Xuan-Son Nguyen, Clémentine Fourrier, Ben Burtenshaw, Hugo Larcher, Haojun Zhao, Cyril Zakka, Mathieu Morlon, and 3 others. 2025. Smollm2: When smol goes big – data-centric training of a small language model. *Preprint*, arXiv:2502.02737.

Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Zeng, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. 2024. LongBench: A bilingual, multitask benchmark for long context understanding. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3119–3137, Bangkok, Thailand. Association for Computational Linguistics.

Maximilian Beck, Korbinian Pöppel, Markus Spanring, Andreas Auer, Oleksandra Prudnikova, Michael Kopp, Günter Klambauer, Johannes Brandstetter, and Sepp Hochreiter. 2024. xlstm: Extended long short-term memory. In *Thirty-eighth Conference on Neural Information Processing Systems*.

Aviv Bick, Kevin Y. Li, Eric P. Xing, J. Zico Kolter, and Albert Gu. 2024. Transformers to ssms: Distilling quadratic knowledge to subquadratic models. *Preprint*, arXiv:2408.10189.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. 2019. Piqa: Reasoning about physical commonsense in natural language. *Preprint*, arXiv:1911.11641.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *Preprint*, arXiv:1803.05457.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *Preprint*, arXiv:2110.14168.

Tri Dao and Albert Gu. 2024. Transformers are ssms: Generalized models and efficient algorithms through structured state space duality. *Preprint*, arXiv:2405.21060.

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, and 5 others. 2023. A framework for few-shot language model evaluation.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *Preprint*, arXiv:1503.02531.

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

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *Preprint*, arXiv:2106.09685.

Jungo Kasai, Hao Peng, Yizhe Zhang, Dani Yogatama, Gabriel Ilharco, Nikolaos Pappas, Yi Mao, Weizhu Chen, and Noah A. Smith. 2021. Fine-tuning pretrained transformers into rnns. *Preprint*, arXiv:2103.13076.

Disen Lan, Weigao Sun, Jiaxi Hu, Jusen Du, and Yu Cheng. 2025. Liger: Linearizing large language models to gated recurrent structures. *Preprint*, arXiv:2503.01496.

Huanru Henry Mao. 2022. Fine-tuning pre-trained transformers into decaying fast weights. *Preprint*, arXiv:2210.04243.

Jean Mercat, Igor Vasiljevic, Sedrick Scott Keh, Kushal Arora, Achal Dave, Adrien Gaidon, and Thomas Kollar. 2024a. Linearizing large language models. In *First Conference on Language Modeling*.

Jean-Pierre Mercat, Igor Vasiljevic, Sedrick Scott Keh, Kushal Arora, Achal Dave, Adrien Gaidon, and Thomas Kollar. 2024b. Linearizing large language models. *ArXiv*, abs/2405.06640.

Chien Van Nguyen, Ruiyi Zhang, Hanieh Deilamsalehy, Puneet Mathur, Viet Dac Lai, Haoliang Wang, Jayakumar Subramanian, Ryan A. Rossi, Trung Bui, Nikos Vlassis, Franck Dernoncourt, and Thien Huu Nguyen. 2025. Lizard: An efficient linearization framework for large language models. *Preprint*, arXiv:2507.09025.

Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Quan Ngoc Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernández. 2016. The lambada dataset: Word prediction requiring a broad discourse context. *Preprint*, arXiv:1606.06031.

Guilherme Penedo, Hynek Kydlíček, Loubna Ben allal, Anton Lozhkov, Margaret Mitchell, Colin Raffel, Leandro Von Werra, and Thomas Wolf. 2024. The fineweb datasets: Decanting the web for the finest text data at scale. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Max Ploner, Jacek Wiland, Sebastian Pohl, and Alan Akbik. 2024. Lm-pub-quiz: A comprehensive framework for zero-shot evaluation of relational knowledge in language models. *Preprint*, arXiv:2408.15729.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2019. Winogrande: An adversarial winograd schema challenge at scale. *Preprint*, arXiv:1907.10641.

Yutao Sun, Li Dong, Shaohan Huang, Shuming Ma, Yuqing Xia, Jilong Xue, Jianyong Wang, and Furu Wei. 2023. Retentive network: A successor to transformer for large language models. *Preprint*, arXiv:2307.08621.

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, and 41 others. 2025. Kimi linear: An expressive, efficient attention architecture. *Preprint*, arXiv:2510.26692.

Dustin Wang, Rui-Jie Zhu, Steven Abreu, Yong Shan, Taylor Kergan, Yuqi Pan, Yuhong Chou, Zheng Li, Ge Zhang, Wenhao Huang, and Jason Eshraghian. 2025a. A systematic analysis of hybrid linear attention. *Preprint*, arXiv:2507.06457.

Junxiong Wang, Daniele Paliotta, Avner May, Alexander M. Rush, and Tri Dao. 2025b. The mamba in the llama: Distilling and accelerating hybrid models. *Preprint*, arXiv:2408.15237.

Jacek Wiland, Max Ploner, and Alan Akbik. 2024. Bear: A unified framework for evaluating relational knowledge in causal and masked language models. *Preprint*, arXiv:2404.04113.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, and 3 others. 2020. Huggingface's transformers: State-of-the-art natural language processing. *Preprint*, arXiv:1910.03771.

Songlin Yang, Jan Kautz, and Ali Hatamizadeh. 2025a. Gated delta networks: Improving mamba2 with delta rule. *Preprint*, arXiv:2412.06464.

Songlin Yang, Bailin Wang, Yikang Shen, Rameswar Panda, and Yoon Kim. 2024. Gated linear attention transformers with hardware-efficient training. *Preprint*, arXiv:2312.06635.

Songlin Yang, Bailin Wang, Yu Zhang, Yikang Shen, and Yoon Kim. 2025b. Parallelizing linear transformers with the delta rule over sequence length. *Preprint*, arXiv:2406.06484.

Lin Yueyu, Li Zhiyuan, Peter Yue, and Liu Xiao. 2025. Arwkv: Pretrain is not what we need, an rnn-attention-based language model born from transformer. *Preprint*, arXiv:2501.15570.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800, Florence, Italy. Association for Computational Linguistics.

Michael Zhang, Simran Arora, Rahul Chalamala, Alan Wu, Benjamin Spector, Aaryan Singhal, Krithik Ramesh, and Christopher Ré. 2024a. Lolcats: On low-rank linearizing of large language models. *Preprint*, arXiv:2410.10254.

Michael Zhang, Kush Bhatia, Hermann Kumbong, and Christopher Ré. 2024b. The hedgehog & the porcupine: Expressive linear attentions with softmax mimicry. *Preprint*, arXiv:2402.04347.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. 2023. Instruction-following evaluation for large language models. *Preprint*, arXiv:2311.07911.

Hyperparameter	Value
Optimizer	AdamW
Weight Decay	0.1
β	(0.9, 0.95)
Stage 1	
LR Scheduler	WarmUp-Stable-Decay
Learning Rate	3×10^{-4}
Batch Size	80
Loss Function	MSE
Stage 2	
LR Scheduler	WarmUp-Stable-Decay
Learning Rate	2×10^{-4}
Batch Size	80
Loss Function	L2-Distance Loss
Stage 3	
LR Scheduler	Linear
Learning Rate	$[3 \times 10^{-4}, 2 \times 10^{-4}, 1 \times 10^{-4}]$
Batch Size	120
Loss Function	KLD + CE

Table 6: Training hyperparameters and stage-wise settings.

A Hyperparameters

B Effect of Short Convolutions

Table 7 shows the effect of adding short convolutions to Linear Attention, GLA, and RetNet across three model scales. Short convolutions consistently improve downstream performance, with particularly pronounced gains on Lambada.

C Effect of Training Context Window

We investigate how the training context window T_{train} affects model quality across two axes: zero-shot downstream performance and in-context retrieval (S-NIAH-1). Table 8 reports results for RetNet and Gated DeltaNet trained with context lengths of 512, 1024, and 2048 tokens for 3B tokens following the 3-stage pipeline described in Section 3. Shorter training contexts yield consistently better zero-shot performance for both architectures. Based on these results, we use $T_{\text{train}} = 512$ as the default throughout the main experiments.

D Full Results: Experiment 1

This section reports full downstream evaluation results for the model size scaling experiment (Experiment 1). All models were trained on 3B distillation tokens. We include per-model and per-task scores for all student architectures at each parameter scale, corresponding to the averaged results summarized in the main text.

E Full Results: Experiment 2 (NIAH)

Table 10 reports the per-subtask Needle-In-A-Haystack accuracy underlying the averaged results in Table 3.

F Full Results: Experiment 3

This section provides detailed results for Experiment 3. We report results for the standard post-trained student models, aswell as the chat-distilled models. For reference, we include results for a SmolLM2-1.7B model trained with the same regime (1 Epoch on Smoltalk) and the teacher model for the chat distillation.

G Efficiency Comparison: Prefilling and Decoding Speed

While each subquadratic architecture should scaling linearly with sequence length during decoding, we validate trainind and decoding times for all models and compare them against a standard transformer architecture (teacher model) in Figure 4. All models are based on the SmollM2-1.7B teacher model with comparable parameter count.

MODEL	LAMB. acc.	WINO. acc.	ARC-E acc.n.	ARC-C acc.n.	PIQA acc.n.	HELL. acc.n.	BEAR acc.	AVG.
Scale: ∼140M Parameters								
Linear Attn	9.51	50.75	42.85	22.35	61.86	30.88	10.56	32.68
+ ShortConv	14.18	50.99	47.35	25.43	64.69	33.52	9.15	35.04
GLA	22.04	52.25	51.52	27.13	65.29	37.19	18.35	39.11
+ ShortConv	26.09	52.33	52.44	27.73	67.08	38.36	19.91	40.56
RetNet	17.02	53.35	49.96	26.54	64.69	35.81	21.54	38.42
+ ShortConv	25.11	52.72	53.28	28.33	65.78	37.70	21.54	40.64
Scale: ∼360M Parameters								
Linear Attn	16.95	50.99	49.33	25.60	65.51	37.74	14.37	37.21
+ ShortConv	24.98	52.72	53.45	28.33	68.12	43.78	20.52	41.70
GLA	29.85	55.88	59.81	31.31	70.08	48.13	24.15	45.60
+ ShortConv	33.35	56.99	60.10	32.34	70.29	48.86	25.36	46.76
RetNet	26.06	54.06	56.86	30.29	68.99	46.23	23.23	43.67
+ ShortConv	31.30	55.17	60.31	38.19	70.84	48.24	24.28	46.90
Scale: ∼1.7B Parameters								
Linear Attn	30.90	53.70	55.10	31.30	69.60	52.50	18.94	44.58
+ ShortConv	48.49	58.96	63.64	36.69	74.10	60.42	33.73	53.72
GLA	50.10	57.20	64.60	37.20	74.60	62.70	34.29	54.38
+ ShortConv	52.82	59.98	67.34	39.33	75.30	63.71	36.37	56.41
RetNet	50.00	58.60	65.80	38.80	74.80	62.90	32.44	54.76
+ ShortConv	53.45	61.01	67.13	39.68	75.14	63.38	36.26	56.58

Table 7: Effect of short convolutions on Linear Attention, GLA, and RetNet across three model scales. Each architecture is shown with and without a short convolutional filter prepended to the input. AVG. is the mean across all seven benchmarks.

MODEL	LAMB. acc.	WINO. acc.	ARC-E acc.n.	ARC-C acc.n.	PIQA acc.n.	HELL. acc.n.	AVG.	S-NIAH-1 (pass-key retrieval)			
								512	1k	2k	4k
RetNet											
$T_{\text{train}} = 512$	52.25	61.01	67.13	39.68	75.14	63.38	59.77	13.0	1.5	0.1	0.0
$T_{\text{train}} = 1024$	47.91	58.09	66.04	38.05	75.52	63.27	58.15	11.0	1.6	0.0	0.0
$T_{\text{train}} = 2048$	47.97	56.99	65.19	37.97	75.79	63.42	57.88	27.4	7.6	1.2	0.0
Gated DeltaNet											
$T_{\text{train}} = 512$	55.90	64.80	69.10	41.90	76.10	66.80	62.43	99.0	95.8	68.4	21.2
$T_{\text{train}} = 1024$	46.90	62.04	69.28	41.98	75.35	65.97	60.25	96.2	82.0	41.8	17.8
$T_{\text{train}} = 2048$	52.69	62.19	66.41	39.85	75.46	65.48	60.34	91.2	82.6	38.2	15.4

Table 8: Downstream results for RetNet and Gated DeltaNet trained with different training context lengths for 3B tokens, following the 3-stage pipeline described in Section 3.

MODEL	LAMB. acc.	WINO. acc.	ARC-E acc.n.	ARC-C acc.n.	PIQA acc.n.	HELL. acc.n.	BEAR acc.	AVG.	RECOV.
Scale: ∼140M Parameters									
SmolLM2-135M (Teacher)	39.16	51.93	58.67	29.61	68.01	43.23	21.84	44.64	100.0%
Gated DeltaNet	22.92	51.85	51.89	28.24	65.40	38.10	18.17	39.51	88.52%
Mamba2	22.41	51.78	53.62	27.90	66.49	38.18	18.94	39.90	89.38%
xLSTM	27.39	50.91	54.12	28.84	66.54	39.10	22.28	41.31	92.55%
Kimi	29.51	51.22	54.92	28.24	66.81	39.89	15.75	40.90	91.62%
GLA	26.09	52.33	52.44	27.73	67.08	38.36	19.91	40.56	90.86%
RetNet	25.11	52.72	53.28	28.33	65.78	37.70	21.54	40.64	91.04%
DeltaNet	17.09	51.38	49.96	27.90	65.29	35.66	18.63	37.99	85.10%
Linear Attn	14.18	50.99	47.35	25.43	64.69	33.52	9.15	35.04	78.50%
Scale: ∼360M Parameters									
SmolLM2-360M (Teacher)	49.29	58.72	68.22	37.88	71.98	56.33	29.45	53.12	100.0%
Gated DeltaNet	31.79	55.17	60.19	32.34	69.80	49.80	25.14	46.32	87.19%
xLSTM	37.30	56.43	60.06	32.76	71.38	51.24	24.80	47.71	89.81%
Mamba2	28.84	53.51	59.76	32.00	71.00	48.98	18.12	44.60	83.96%
Kimi	39.90	56.99	63.85	34.30	71.87	52.14	27.39	49.49	93.17%
GLA	33.35	56.99	60.10	32.34	70.29	48.86	25.36	46.76	88.03%
RetNet	31.3	55.17	60.31	38.19	70.84	48.24	24.28	46.90	88.29%
DeltaNet	23.02	51.78	54.38	28.92	68.34	44.61	19.07	41.45	78.02%
Linear Attn	24.98	52.72	53.45	28.33	68.12	43.78	20.52	41.70	78.50%
Scale: ∼1.7B Parameters									
SmolLM2-1.7B (Teacher)	64.95	65.82	73.32	47.18	77.37	71.40	39.93	62.85	100.0%
Gated DeltaNet	55.90	64.80	69.10	41.90	76.10	66.80	39.93	59.22	94.22%
Mamba2	53.62	60.14	67.17	40.10	75.79	65.47	38.37	57.24	91.07%
xLSTM	52.70	63.30	67.10	39.80	75.80	64.90	39.25	57.55	91.56%
Kimi	52.10	63.40	68.00	42.30	76.60	65.60	36.80	57.83	92.01%
GLA	52.82	59.98	67.34	39.33	75.30	63.71	36.37	56.41	89.75%
RetNet	53.45	61.01	67.13	39.68	75.14	63.38	36.26	56.58	90.02%
DeltaNet	51.00	61.20	67.70	39.80	73.70	63.20	33.11	55.67	88.58%
Linear Attn	48.49	58.96	63.64	36.69	74.10	60.42	33.73	53.72	85.47%

Table 9: Experiment 1: Complete Zero-shot downstream benchmarks across all scales.

	S-NIAH-1 (Pass-key)				S-NIAH-2 (Number)				S-NIAH-3 (UUID)			
Model	512	1k	2k	4k	512	1k	2k	4k	512	1k	2k	4k
Gated DeltaNet	**99.4**	**97.8**	**81.8**	**29.8**	**99.0**	**93.4**	**55.0**	**15.6**	**16.4**	**3.2**	**1.2**	**0.4**
xLSTM	95.8	78.0	25.2	8.4	98.6	82.2	32.0	8.6	6.2	0.4	0.0	0.0
DeltaNet	98.8	96.2	0.0	0.0	99.2	77.0	24.6	0.0	1.8	0.0	0.0	0.0
Kimi	97.0	84.0	45.0	14.0	98.0	86.0	44.0	11.0	11.0	1.8	0.5	0.2
Mamba2	14.8	4.2	0.0	0.0	81.2	16.2	7.2	3.8	0.0	0.0	0.0	0.0
GLA	10.6	1.0	0.0	0.0	50.0	13.0	3.0	1.6	0.0	0.0	0.0	0.0
RetNet	10.4	0.0	0.0	0.0	30.2	10.2	0.0	0.0	0.0	0.0	0.0	0.0
Linear Attn	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

Table 10: Experiment 2: Full per-subtask Needle-In-A-Haystack (NIAH) performance of distilled models at 10B tokens. Scores represent retrieval accuracy (%).

	Zero-Shot	Retrieval	Instruction Following & Reasoning			
Model	LLM Avg	NIAH-S Avg	IFEval	GSM8K	SWDE	LongB.
Regime A: Post-Training Adaptation (Teacher: *SmolLM2-1.7B + SmolTalk* SFT 1 Epoch)						
Teacher Baseline	*70.24*	*99.31*	*56.71*	*42.08*	*23.85*	*17.45*
Gated DeltaNet	**61.07**	**57.33**	40.89	8.28	13.33	**17.74**
xLSTM	60.52	47.15	**45.08**	**10.17**	14.03	16.68
GLA	56.59	10.78	40.89	8.28	13.33	**17.74**
DeltaNet	56.29	39.38	36.21	9.72	3.62	2.27
RetNet	56.29	6.28	32.61	6.39	9.67	2.05
Kimi	56.70	45.04	38.00	8.50	**14.50**	15.00
Linear Attn	44.70	0.00	24.46	1.62	2.21	0.00
Linear Attn (conv)	44.70	0.00	26.02	0.2	1.5	0.2
Mamba2	57.88	14.75	37.89	11.14	6.75	13.61
Regime B: Instruction-Aware Distillation (Teacher: *SmolLM2-1.7B-Instruct*)						
Teacher Baseline	*70.57*	*98.06*	*60.19*	*49.51*	*22.32*	*25.64*
Gated DeltaNet	**55.35**	**47.33**	**47.12**	**20.02**	**10.08**	**15.24**
Deltanet	55.34	40.10	42.17	13.12	9.36	13.23
RetNet	53.69	10.61	31.06	2.81	7.11	7.98
xLSTM	51.64	10.41	41.73	13.50	8.10	14.91
GLA	49.11	4.35	29.50	5.84	5.13	10.79
Kimi	50.00	41.25	42.00	15.00	9.00	13.00
Linear Attn	31.28	0.00	22.78	0.00	1.40	2.00
Mamba2	52.27	15.60	41.01	17.15	7.92	13.82

Table 11: Comparison of instruction-following performance across two adaptation strategies. Regime A evaluates the effectiveness of standard post-training SFT on linearized base models, while Regime B examines a holistic instruction-aware distillation pipeline. Each regime is contrasted against its respective Transformer teacher to measure relative recovery of specialized capabilities.

Model	Lamb. acc.	WinoG. acc.	Arc-E acc. norm.	Arc-C acc. norm.	PIQA acc. norm.	HellaS. acc. norm.	BEAR acc.
Post Training							
Teacher (SmolLM2-1.7B, fine-tuned)	61.61	67.25	66.46	42.58	76.55	70.82	36.94
Linear Attn	26.11	52.09	45.16	29.35	66.81	48.67	25.21
Retnet	48.01	61.56	55.85	36.01	74.05	62.26	32.19
GLA	52.04	63.69	58.84	36.26	74.97	64.04	32.32
xLSTM	**54.62**	**65.43**	63.59	37.63	**75.30**	66.58	37.49
Deltanet	47.33	58.09	59.89	36.95	73.23	62.25	29.57
Gated Deltanet	54.41	65.11	**64.77**	**39.85**	**75.30**	**66.97**	36.87
Kimi	48.20	58.60	62.90	39.10	70.80	60.60	32.00
Mamba2	52.38	63.14	55.64	37.88	73.78	64.45	32.73
Chat Distilled							
Teacher (SmolLM2-1.7B-Instruct)	61.10	68.19	63.13	44.03	76.12	71.82	39.06
Linear Attn	0.38	50.12	33.08	20.39	55.55	28.16	5.49
Retnet	43.14	59.98	60.19	34.04	72.63	52.16	25.97
GLA	41.90	58.33	48.99	34.30	71.65	60.78	19.46
xLSTM	41.19	60.14	52.10	31.14	68.17	57.07	20.51
Deltanet	41.89	58.41	59.97	37.80	72.42	61.70	27.21
Gated Deltanet	41.93	57.93	60.10	37.97	72.36	61.83	27.04
Kimi	42.50	51.70	55.40	34.50	62.40	53.50	15.00
Mamba2	29.57	58.17	59.85	34.30	71.82	59.91	18.95

Table 12: Experiment 3: Performance of linearized models after post-training instruction tuning and instruction-aware distillation. Results are reported across instruction-following, reasoning, and long-context benchmarks.

Model	S-NIAH-1 (pass-key retrieval)				S-NIAH-2 (number in haystack)				S-NIAH-3 (uuid in haystack)			
	512	1k	2k	4k	512	1k	2k	4k	512	1k	2k	4k
Post Training												
Teacher (SmolLM2-1.7B, fine-tuned)	100	100	100	100	100	100	99.8	99.8	100	99.8	97.4	95.0
Linear Attn	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Retnet	12.4	1.0	0.0	0.0	47.8	13.8	0.4	0.0	0.0	0.0	0.0	0.0
xLSTM	99.2	93.8	64.6	17.6	99.4	89.4	48.8	15.2	30.2	3.8	3.2	0.6
GLA	50.6	7.0	1.0	0.0	81.2	42.6	11.0	4.8	1.6	0.0	0.0	0.0
Deltanet	98.6	**99.4**	0.0	0.0	**99.6**	92.8	**62.4**	0.0	18.2	1.6	0.0	0.0
Gated Deltanet	**99.8**	**99.4**	**98.0**	**65.8**	99.4	**95.0**	58.2	**21.6**	**32.2**	**12.0**	**4.6**	**2.0**
Kimi	98.0	92.0	62.0	16.0	98.5	88.0	50.0	14.0	18.0	3.0	0.8	0.2
Mamba2	37.2	7.4	2.6	1.4	95.0	21.6	8.2	3.4	0.2	0.0	0.0	0.0
Chat Distilled												
Teacher (SmolLM2-1.7B-Instruct)	99.8	99.8	98.2	100	100	86.6	98.6	99.6	100	94.4	100	99.8
Retnet	13.4	0.2	0.0	0.0	89.0	24.8	0.0	0.2	0.0	0.0	0.0	0.0
GLA	8.6	0.0	0.0	0.0	92.4	11.6	2.8	2.0	0.0	0.0	0.0	0.0
xLSTM	19.8	3.4	1.0	0.8	70.0	26.0	2.2	1.8	0.0	0.0	0.0	0.0
Deltanet	98.0	94.2	63.2	22.2	99.0	90.8	50.6	11.6	0.6	0.0	0.0	0.0
Gated Deltanet	98.6	98.0	72.4	24.0	99.6	91.8	59.0	17.0	7.6	0.0	0.0	0.0
Kimi	95.0	85.0	52.0	14.0	98.0	86.0	48.0	12.0	5.0	0.0	0.0	0.0
Mamba2	38.4	3.4	0.0	0.0	97.4	41.4	3.4	3.2	0.0	0.0	0.0	0.0

Table 13: Experiment 3: Full results for the NIAH-S task of the RULER benchmark.

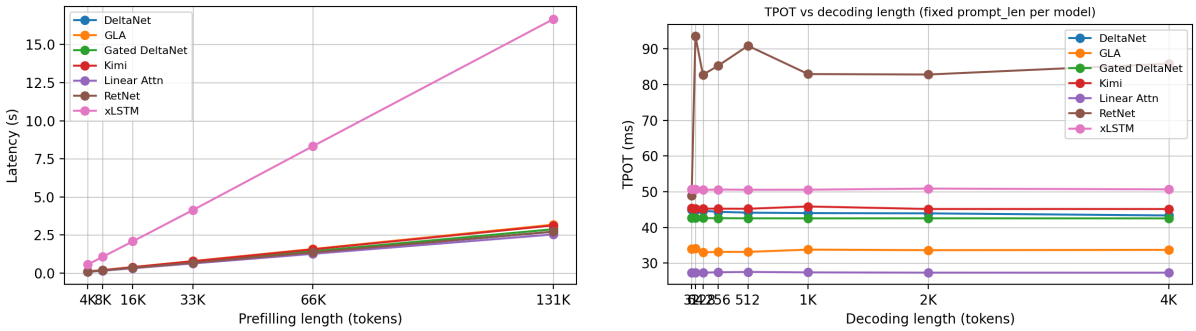

Figure 4: The prefilling and decoding time of all 1.7B distilled models for different input and decoding lengths. All models exhibit linear scaling for ingesting input and constant time for generating tokens.