PRISM: A PROBE SUITE FOR RETRIEVAL, INTERFERENCE, AND SELECTIVE MEMORY

Anonymous authors
Paper under double-blind review

ABSTRACT

We introduce **PRISM**, a suite of seven synthetic probes for in-context retrieval. The probes vary how key-value bindings are arranged in the context (dispersion) and how many must be held at once (scope), with sequence length and vocabulary fixed across three difficulty tiers. We train six architectures (GDN, GDP, RWKV-7, Mamba2, MLA, Transformer) from scratch with dense next-token cross-entropy at matched parameters and training budget, with per-seed reporting and length and complexity generalisation tests. We control for the most direct state-size confound by matching nominal recurrent state dimension to the hardest binding count. We find that: (i) under this small-scale, fixed-budget regime, RWKV-7 solves nearly the entire suite and GDN/GDP solve task-dependent subsets, while Transformer, MLA, and Mamba2 solve no task; (ii) controlled interventions show that retrieval-supervision density is a major factor in whether softmax attention learns to retrieve, which helps reconcile PRISM with MQAR and MAD, while Mamba2 solves retrieval only at smaller vocabulary and shorter sequences; (iii) the structured-versus-unstructured dispersion axis dissociates models that the difficulty knob alone does not; (iv) when sequence length and the difficulty knob are each scaled by the same factor ($2\times$ or $4\times$), complexity generalisation degrades more than length generalisation on five of seven tasks.

1 INTRODUCTION

Softmax attention (Vaswani et al., 2017) remains the dominant primitive for sequence modelling, but its quadratic time and linear KV-cache cost have driven two lines of work. The first is sub-quadratic alternatives that replace softmax with a fixed-size recurrent state: selective state-space models such as Mamba and Mamba2 (Gu & Dao, 2024; Dao & Gu, 2024); linear-attention models built around the delta rule, including DeltaNet (Yang et al., 2024b), Gated DeltaNet (GDN) (Yang et al., 2025), and Gated DeltaProduct (GDP) (Siems et al., 2025); and recurrent architectures with explicit associative-memory updates such as RWKV-7 (Peng et al., 2025). The second line keeps softmax attention but reduces its memory footprint: multi-head latent attention (MLA) (DeepSeek-AI, 2024), for example, compresses the KV cache through a low-rank latent projection while retaining the $O(L^2)$ softmax computation. The central design question facing both lines is how much of the recall capability of softmax attention can be retained when state is bounded (per-step recurrent state in the first line, cached KV state in the second).

The dominant framework for understanding recall in efficient architectures was set by Zoology (Arora et al., 2024a) and the BASED recall-throughput tradeoff (Arora et al., 2024b). Zoology established that gated-convolution and linear-attention architectures underperform softmax attention on multi-query associative recall; BASED formalised this as a Pareto tradeoff between recurrent state size and recall accuracy. These are well-supported claims about a real tradeoff: when state size is constrained, recall is bounded by it. Subsequent work has extended this analysis in several directions. Yang et al. (2024b) showed that DeltaNet outperforms gated linear attention at *matched* state size. Wang et al. (2025) argue that the rank-1 correction in delta-rule models injects quadratic feature interactions that give the recurrent state nonlinear expressiveness beyond what standard softmax attention provides. Nichani et al. (2025) show that Transformer recall scales with parameter count.

These results all operate within the regime where state size is the binding constraint. The complementary regime, in which state size is verified to be sufficient for the recall task, has received much

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.

less attention. In this regime the question shifts from capacity to acquisition: given a recurrent or linear-attention model whose per-step state can in principle store the required bindings, will it actually learn to retrieve them in context, and how data-efficient is that learning relative to softmax attention? Arora et al. (2025) show that architectures with similar associative-recall accuracy can implement recall through different mechanisms, such as induction in Transformers and short convolutions in Mamba.

Recent sub-quadratic architectures such as GDN, GDP, and RWKV-7 report performance approaching or matching softmax attention on language modelling and recall-intensive tasks at competitive scale. These advances invite a more pointed comparison: how these architectures behave when state capacity is no longer the bottleneck and the question is whether each learns to retrieve. Such a comparison requires a benchmark that probes retrieval through several tasks that each require a different operation, keeps state size at least matched to the binding count, varies dispersion and scope independently, and trains every model from scratch under the same stated objective, since how retrieval is supervised can decide whether it is learned.

We introduce **PRISM**, a seven-task synthetic recall benchmark whose tasks span a dispersion-structure axis, each evaluated at three difficulty tiers along an orthogonal scope axis. Dispersion structure, that is, how key-value bindings are arranged in the context, differentiates the seven tasks. One task scatters bindings uniformly at random and serves as the unstructured baseline; the other six each impose a different structural prior on placement. Scope, the number of bindings the model must hold simultaneously, is varied across the three difficulty tiers, with sequence length and vocabulary held fixed. Together, dispersion and scope realise at the architectural-recall scale the two orthogonal axes that Goldman et al. (2024) identify as governing long-context task difficulty. Beyond these two axes, three tasks additionally expose auxiliary sub-metrics that diagnose specific failure modes, so that two models with identical aggregate scores can still be distinguished by how they fail.

We evaluate six architectures on PRISM: GDN, GDP, RWKV-7, Mamba2, MLA, and a vanilla Transformer. Each model is trained from scratch on each task at three difficulty tiers (easy, moderate, hard) with two random seeds per condition under a fixed training-compute budget, and we report per-seed results. We run two seeds because seed-bimodality is the diagnostic: under matched-compute training, some conditions exhibit one seed solving and one failing, revealing that, near the limit of what the budget allows, whether a task is acquired depends on the seed; this bimodality signal would be destroyed by averaging over seeds or masked by reporting only the best run. To test whether models that solve a task have learned an in-context retrieval capability that transfers beyond their training configuration, we evaluate them on longer sequences (length generalisation) and on a scaled scope knob (complexity generalisation).

Our contributions are: **(1) PRISM**, a capacity-controlled instrument for evaluating in-context retrieval in models trained from scratch: seven operational probes spanning a dispersion axis, each at three scope tiers with sequence length and vocabulary fixed, trained with dense next-token cross-entropy under matched parameter and compute budgets, with per-seed reporting and length and complexity generalisation. **(2)** A protocol-level finding on retrieval-supervision density: under dense next-token training, delta-rule and RWKV-7 models solve most tasks while Transformer, MLA, and Mamba2 solve none, and controlled interventions show that the density of retrieval supervision is a major factor in whether softmax attention learns to retrieve, which helps reconcile PRISM with MQAR and MAD, while Mamba2 solves retrieval only at smaller vocabulary and shorter sequences. **(3)** Two further findings: the structured-versus-unstructured dispersion axis separates models that the difficulty knob alone does not, and when sequence length and the difficulty knob are each scaled by the same factor, complexity generalisation degrades more than length generalisation on five of seven tasks. Code is available at `https://anonymous.4open.science/r/prism_submission-5F3F`.

2 PRISM Benchmark Design

2.1 Design Overview

PRISM consists of seven synthetic in-context retrieval tasks. Every task takes the form of a single sequence of integer tokens drawn from a finite vocabulary $\mathcal{V}$, partitioned into role-specific pools (keys, values, noise fillers, and a small number of structural markers such as anchors, narrative tags, or a fixed query-prefix token). A model is trained to predict the next token at every position, but only

a designated set of value positions in the query block contributes to the evaluation metric, so retrieval answers are a small fraction of the training targets. The seven tasks span a dispersion-structure axis; each is evaluated at three difficulty tiers spanning an orthogonal scope axis. Three tasks additionally expose auxiliary sub-metrics.

Dispersion structure describes how the key-value bindings are distributed in the context. Noisy Context Retrieval (NCR) places bindings uniformly at random in a noise body and is the only test with no exploitable dispersion structure; the remaining six tests place bindings under a structural prior: two phases with disjoint vocabulary pools in Context Shift Retention (CSR), composite-key triples sharing key tokens across clusters in Clustered Associative Recall (CAR), an anchor-then-exemplar two-pass layout in Two-Hop Retrieval (THR), two parallel narratives with a tag-disambiguation token in Narrative Interference Suppression (NIS), K contiguous topic segments in Multi-Topic Retrieval (MTR), or a single early signal followed by late distractors in Ancient Key Retrieval (AKR). Each structural prior is a different exploitable shortcut for routing retrieval; their dissociation is the central use of the suite.

Scope is the number of bindings the model must hold simultaneously. Every task exposes a difficulty knob (N bindings, K topics, S shared keys, or D distractors) that scales scope from easy to moderate to hard, with sequence length and vocabulary size held fixed across difficulties so that the knob isolates a single axis. This realises at the architectural-recall scale the dispersion-scope decomposition identified by Goldman et al. (2024) as governing long-context task difficulty.

Beyond top-level retrieval accuracy, three tasks expose auxiliary sub-metrics that diagnose specific failure modes: CSR reports separate accuracies on Phase 1 (early) versus Phase 2 (late) bindings to expose recency bias; NIS reports per-narrative accuracy to expose interference direction; and MTR reports early-half vs. late-half topic accuracy to expose topic-position bias. These sub-metrics let a model that achieves an identical aggregate accuracy be distinguished from another that fails for a structurally different reason.

Token vocabularies are partitioned in fixed proportions of $|\mathcal{V}| = 256$ across difficulties (e.g., NCR uses $|\mathcal{V}|/4$ each for the key and value pools; the rest of the vocabulary is noise), and key-value mappings, key identities, and noise fillers are freshly sampled per episode from these pools so that no token-ID-to-role association can be learned in-weights. Table 1 summarises the seven tests and their primary structural and scope dimensions; Figure 1 sketches the sequence layout of each.

Table 1: The seven PRISM tasks. *Dispersion* names the structural prior on how key-value bindings are distributed in the context; *Scope knob* names the parameter that controls binding count from easy to hard; ***E / M / H*** gives the knob value at the easy, moderate, and hard tiers, and L the sequence length, which is fixed across tiers; *Eval focus* names the primary metric and any auxiliary sub-metric. All tests use vocabulary size $|\mathcal{V}| = 256$.

ID	Task	Dispersion	Scope knob	E / M / H	L	Eval focus
T1	NCR	uniform random	N bindings	4 / 16 / 32	1024	accuracy
T2	CAR	composite-key triples	$N_c \cdot M$	16 / 32 / 64	512	accuracy
T3	CSR	two phases (disjoint pools)	N per phase	4 / 16 / 32	2048	acc. + P1/P2
T4	THR	anchors-then-exemplars	K categories	4 / 8 / 16	1024	two-hop acc.
T5	NIS	two narratives, tagged	S shared keys	0 / 4 / 8	1024	acc. + per-narr.
T6	MTR	K topic segments	K topics	2 / 4 / 8	1024	acc. + early/late
T7	AKR	early signal, late distractors	D distractors	3 / 16 / 32	2048	sig. acc.

2.2 Test Descriptions

For each task, we describe the sequence layout and the primary evaluation metric; Table 1 lists the difficulty knob, its value at each tier, and the sequence length.

Noisy Context Retrieval (NCR) is the unstructured baseline. Vocabulary is split into a key pool ($|\mathcal{V}|/4$), a value pool ($|\mathcal{V}|/4$), a noise pool, and a query-prefix token. N key-value pairs are sampled fresh per episode and planted at N uniformly-chosen even positions in a noise-padded body of length L; queries follow the body in random order, each emitted as $\langle \text{QP}, k_i, v_i \rangle$ with the value

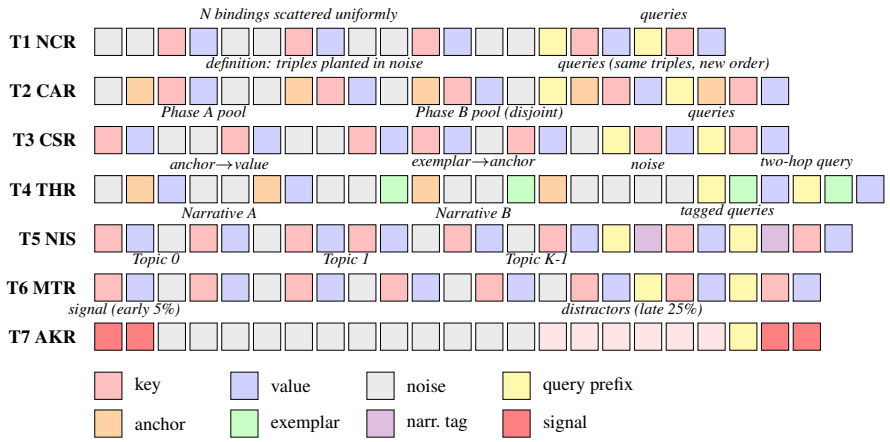


Figure 1: Sequence layouts for the seven PRISM tasks. Each row shows a representative sequence (positions truncated for legibility): role-coloured cells indicate keys, values, noise fillers, and task-specific tokens. Sequence length and vocabulary size are held fixed across difficulties on each task.

position scored. NCR is the only PRISM task whose dispersion is uniform random: it isolates raw multi-binding capacity with no structural shortcut available.

Clustered Associative Recall (CAR) tests retrieval when bindings live in clusters. Vocabulary is split into anchor ($|\mathcal{V}|/8$), shared-key ($|\mathcal{V}|/4$), and per-cluster value pools ($|\mathcal{V}|/2$); N_c clusters each have $M = 4$ shared keys but cluster-specific values, giving $N_c \cdot M$ unique $(\text{anchor}, \text{key}) \to \text{value}$ triples. The sequence has a single definition pass in which the triples are planted at non-overlapping random positions in a noise body, followed by a compact query block in which each binding is queried via the prefixed quadruple $\langle \text{QP}, \text{anchor}, \text{key}, \text{value} \rangle$ and the value position is scored. CAR therefore requires resolving the same key token to different values depending on its anchor.

Context Shift Retention (CSR) tests retention across an explicit context shift. The vocabulary is split into five equal pools: KEY_1, VAL_1, KEY_2, VAL_2, and noise. Phase 1 occupies the first half and Phase 2 the second, each containing N key-value pairs drawn from $(\text{KEY}_1, \text{VAL}_1)$ and $(\text{KEY}_2, \text{VAL}_2)$, respectively. Queries cover all $2N$ pairs, with Phase 1 queries emitted first. Disjoint vocabulary pools mean the phase of any key is decodable from its token ID alone, providing a routing prior. Auxiliary metrics report Phase 1 and Phase 2 accuracy separately and their delta, which flags recency bias (Phase 2$\gg$Phase 1 indicates the older bindings have been overwritten).

Two-Hop Retrieval (THR) tests in-context composition. The context contains both anchor $\to$ value pairs (one for each of K categories) and exemplar $\to$ anchor pairs ($N = 4$ exemplars per category). At query time the model is given a held-out exemplar and must return the value associated with its category, requiring two retrieval hops: exemplar $\to$ anchor $\to$ value. Anchor and value tokens are sampled fresh per episode from disjoint pools; exemplar tokens occupy a fixed per-category range so that the category structure is the semantic signal the model must learn within the episode. The anchor-value pairs are planted in a noise-padded first third of the sequence, the exemplar-anchor pairs in a noise-padded middle third, and the last third is pure noise.

Narrative Interference Suppression (NIS) tests retrieval under controllable interference. Two parallel narratives, A and B, each contain N key-value pairs ($N = 8$ at easy and $N = 16$ at moderate and hard); S of the N keys are shared between narratives but bound to different values, while the remaining $N - S$ keys are unique to each narrative. Each narrative occupies one half of the sequence, with the relevant pairs planted in noise. At query time a narrative-tag token disambiguates which value is being requested: $\langle \text{QP}, \text{NARR-TAG}, k, v \rangle$. Auxiliary metrics report narrative-A and narrative-B accuracy separately; their delta reveals which narrative the model preferentially routes to under interference.

Multi-Topic Retrieval (MTR) tests retrieval over multiple topic segments. The sequence is partitioned into K contiguous segments of approximately equal length, each functioning as a self-contained "topic" with $N = 4$ key-value pairs planted in segment-local noise. Keys and values across all $K \cdot N$

pairs are sampled disjointly so that no token appears in more than one topic. Queries cover all $K \cdot N$ pairs in topic order, from topic 0 (oldest, hardest) to topic $K-1$ (most recent, easiest). Auxiliary metrics report accuracy on the early half of topics versus the late half.

Ancient Key Retrieval (AKR) tests primacy retention against late distractors. A single *signal* key-value pair is planted at a random position in the first 5% of the sequence; D *distractor* key-value pairs are then planted at evenly-spaced positions within the last 25% of the sequence. Queries ask for the signal first (so the signal value is identifiable in evaluation) and then for each distractor in random order. The primary metric is the signal accuracy.

2.3 Protocols

In-distribution evaluation holds out a validation set for each (task, difficulty, model, seed) for model selection (best epoch, best LR) and reports *Test@*L: a fresh evaluation set generated at the training sequence length L with the seeded random number generator advanced past training. The metric is task-specific (Table 1): top-level accuracy on value positions for T1–T6, signal accuracy for T7. Auxiliary sub-metrics are reported alongside the primary metric.

Length generalisation evaluates every checkpoint that solves the in-distribution test on test sets at $L' \in \{512, 1024, 2048, 4096\}$, with the task configuration fixed at the training values. Only the noise body grows; the bindings remain the same in count and structure. This isolates whether the learned solution survives longer noise distances. **Complexity generalisation** scales the task-specific difficulty knob by factors of $2\times$ and $4\times$, with sequence length and vocabulary fixed at training values. This isolates whether the learned solution operates on the structure of the task or only at the binding count it was trained on.

2.4 Relation to Prior Benchmarks

PRISM's probes adapt constructs with clear precedent: multi-query associative recall (MQAR; Arora et al., 2024a), the recall and noisy-recall tasks of MAD (Poli et al., 2024), needle-in-a-haystack and PassKey retrieval (Kamradt, 2023; Mohtashami & Jaggi, 2023), the multi-hop and multi-query variants of RULER (Hsieh et al., 2024), and the variable-binding and serial-position literatures (Smolensky, 1990; Liu et al., 2024). Table 2 lists the closest antecedent of each probe and how the probe differs from it.

Table 2: Closest antecedent of each PRISM probe and the operational difference from it.

Probe	Required operation	Closest antecedent	Operational difference
NCR	one-shot retrieval amid heavy filler noise	MQAR; MAD noisy recall	scattered one-shot bindings; terminal query block
CAR	composite (anchor, key) addressing	variable binding	same key aliased across clusters, resolved by anchor
CSR	retention across a context shift	serial position; interference	no close synthetic predecessor; per-phase scoring
THR	composing learned category structure with a fresh in-context map	RULER variable tracking	held-out exemplars of learned categories, not assignment chains
NIS	selection among conflicting valid bindings	distractor robustness	distractors are conflicting valid bindings, not noise
MTR	query-indexed retrieval over topic segments	multi-query NIAH	contiguous topic segments; early/late scoring
AKR	recovering an early binding against recent writes	PassKey; NIAH	distractors are valid recent bindings; signal scored separately

At the protocol level, PRISM differs from MQAR (Arora et al., 2024a) and the MAD pipeline (Poli et al., 2024) in four respects: (a) per-seed reporting that surfaces training bimodality as a diagnostic rather than averaging over it, (b) complexity generalisation (scaling the task knob) as a separate axis from length generalisation, (c) a verified-sufficient state-capacity regime, and (d) a dense next-token objective with one-shot bindings, whereas MQAR supervises only the answers and MAD repeats keys within a small vocabulary.

Table 3: Model configurations used in all experiments.

Model	Family	dim	heads	n_ℓ	Params	head dim
GDN	Linear attention	256	4	4	4.61M	64
GDP	Linear attention	256	4	4	4.61M	64
Mamba2	Selective SSM	256	4	5	4.43M	256
MLA	Latent attention	256	4	5	4.85M	64 + 32
RWKV-7	Recurrent	256	4	4	4.39M	64
Transformer	Softmax attention	256	4	5	4.39M	64

3 Experiments and Results

3.1 Models

We evaluate six architectures spanning the principal sub-quadratic and softmax-attention families discussed in Appendix A.2: **GDN** (Gated DeltaNet), **GDP** (Gated DeltaProduct), and **RWKV-7** as the sub-quadratic linear-attention and recurrent associative-memory models; **Mamba2** as a selective state-space model; **MLA** (multi-head latent attention) as a softmax-attention KV-compression variant; and a standard **Transformer** as the softmax-attention reference. All models are implemented as drop-in causal-LM modules from the flash-linear-attention library (Yang & Zhang, 2024), with thin wrappers in our codebase that fix the configuration of each architecture to a common $\mathrm{dim} = 256$, $\mathrm{num_heads} = 4$ specification (Table 3); depth $n_\ell \in \{4, 5\}$ is varied per model to keep parameter counts in a narrow 4.4–4.9M band. We treat the library implementations as authoritative: each architecture is instantiated through its canonical config class with no architecture-specific surgery, so that the comparison is between the published mechanisms as their authors defined them.

The matched-budget regime is deliberately small-scale: we want per-head state capacity to be at-or-above the demands of every difficulty tier in PRISM. For the three linear recurrent models with head dimension 64 (GDN, GDP, RWKV-7), each head's outer-product associative state has rank-bound capacity of 64 bindings. With 4 heads per layer, aggregate per-layer capacity reaches 256 bindings only if heads specialise to disjoint binding subsets. Effective per-layer capacity therefore lies somewhere in the range $[64, 256]$. The hardest task in the suite is CAR-Hard with $N_c \cdot M = 64$ bindings, which sits at the per-head lower bound; NCR-Hard requires 32 bindings, comfortably below even a single head's capacity. For Mamba2, the SSM hidden state with expand factor 4 and head dim 256 carries an even larger nominal capacity. In other words, no failure observed below can be ascribed to insufficient state at the per-head level.

3.2 Training

All models are trained from scratch on each (task, difficulty) pair under a fixed compute budget. Training uses 100 epochs, batch size 128, and a cross-entropy next-token loss applied at every position so that the in-context retrieval circuit must form jointly with the language-model objective. We sweep three learning rates per seed in decreasing order, $\{10^{-3}, 5 \times 10^{-4}, 10^{-4}\}$, with no weight decay, and run two random seeds per condition. All training runs are conducted on a single NVIDIA H100 GPU.

For each seed and each (task, difficulty) pair, the LR sweep is governed by two thresholds derived from validation accuracy at the training sequence length: the *full-accuracy stop* (≥ 0.995), in which we treat the task as essentially solved once a learning rate reaches this accuracy and terminate the sweep early; and the *solved threshold* (≥ 0.9), in which we call a seed *solved* on a condition if its best validation accuracy across the sweep is at least 0.9. The solved threshold separates three regimes that are visible across our results: chance behaviour (typically below 0.05, i.e., $\approx 3\times$ the uniform-random rate of $1/64$ over $|\mathcal{V}|/4$ value tokens), partial-credit / shortcut solutions (0.05–0.9), and full circuit acquisition (≥ 0.9).

3.3 Results

We report results in two blocks: a categorical evaluation-accuracy summary across all six architectures (Figure 2), and length and complexity generalisation on solved seeds (Figure 3). The figures present

a categorical view, showing whether each (model, task, eval-point) combination was solved by both seeds, by one seed only, or by neither, so that the dissociation patterns are visible at a glance; the exact per-seed accuracies (including bimodality information collapsed by the categorical encoding) are tabulated in full in Appendix B (Tables 6–12). The primary metric is task-specific (Table 1): top-level evaluation accuracy on value positions for T1–T6, and signal accuracy for T7.

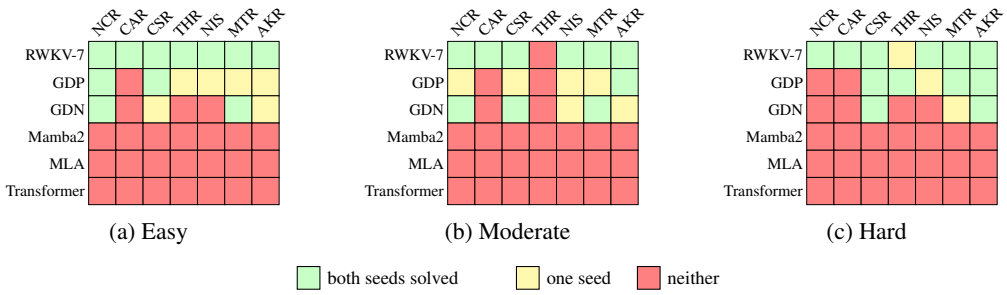


Figure 2: In-distribution evaluation accuracy summary across all six architectures, organised by difficulty tier. In each panel, columns are tasks and rows are models. Each cell encodes the outcome on one (model, difficulty, task) condition: green if both random seeds reached ≥ 0.9 best validation accuracy at training L; yellow if exactly one seed did; red if neither. The top three rows are mostly green/yellow at every difficulty; the bottom three rows are uniformly red. Per-seed exact accuracies appear in Table 6 (Appendix B).

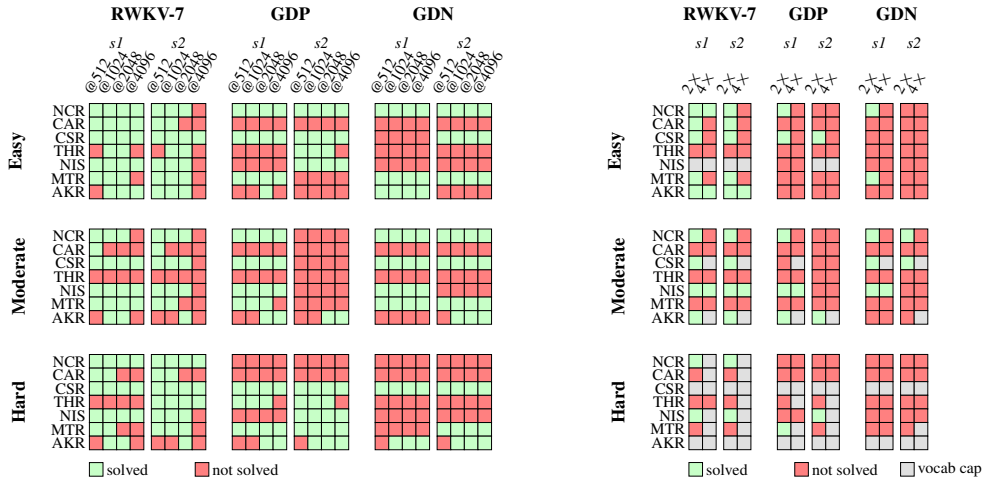
In-distribution accuracy is summarised in Figure 2 for all six architectures on all seven tasks at all three difficulty tiers; Table 6 (Appendix B) reports the per-seed best validation accuracy at training sequence length. At every difficulty Mamba2 sits at chance across the entire suite. MLA and the vanilla Transformer pass nothing decisively at any difficulty, with isolated single-seed partial credit on THR (Transformer s1 0.248, MLA s1 0.412 at easy), on CSR (Transformer s1 0.117 at moderate; Transformer s1 0.112 and s2 0.097 at hard), and on MTR (Transformer s2 0.521 at moderate, 0.411 at hard) that never survives a second seed and never crosses the solved threshold. By contrast RWKV-7 solves all seven tasks at easy and at hard, and six of seven at moderate (THR fails in both seeds, at 0.349 and 0.348). GDP and GDN show task-dependent solve patterns at each difficulty, with strong seed-bimodality on several conditions (e.g., GDP NIS-easy s1 0.020 vs. s2 0.999; GDN MTR-hard s1 0.018 vs. s2 0.993; GDP NCR-moderate s1 0.996 vs. s2 0.035).

Length and complexity generalisation (§2.3) are shown in Figure 3 for every seed that solved a task in distribution; per-seed accuracies are listed in Tables 7–12 (Appendix B). At twice the training value, 72 of 81 solved checkpoints remain solved under length scaling but only 38 of 66 under complexity scaling; at four times, 29 of 50 and 7 of 41. When sequence length and the difficulty knob are scaled by the same factor, complexity generalisation degrades more than length generalisation on five of seven tasks, comparably on NIS, and less on AKR (Appendix F.1.3).

Auxiliary sub-metrics diagnose specific failure modes for three tasks (CSR, NIS, and MTR; §2.1). Table 13 (Appendix C) reports CSR sub-metrics per seed, with Δ flagging direction-of-failure; the analogous NIS and MTR tables (Tables 14 and 15) appear in the same appendix. On CSR, the Phase-2−Phase-1 delta is large and positive in three conditions where the top-level accuracy is degraded: GDN-easy s1 ($\Delta = +0.681$ with P1 $= 0.025$, P2 $= 0.706$), GDP-moderate s2 ($\Delta = +0.955$ with P1 $= 0.040$, P2 $= 0.995$), and Transformer-moderate s1 ($\Delta = +0.191$ with P1 $= 0.021$, P2 $= 0.212$). On NIS and MTR, $|\Delta|$ is small across all conditions.

4 Why Do Softmax Attention and Mamba2 Fail?

Transformer, MLA, and Mamba2 solve no PRISM task at any tier (§3.3). We first rule out implementation and pipeline errors (§4.1) and then test two explanations: the density of retrieval supervision for the softmax-attention models (§4.2) and task scale for Mamba2 (§4.3). All results in this section are behavioural: they locate where acquisition fails but do not identify internal mechanisms.



(a) Length generalisation: columns are eval lengths $\{512, 1024, 2048, 4096\}$.

(b) Complexity generalisation: columns are knob-scaling factors $\{2\times, 4\times\}$.

Figure 3: Per-seed generalisation outcomes for the three architectures that solved at least one task at training (RWKV-7, GDP, GDN). Within each subfigure, models are laid out horizontally and difficulty tiers are stacked vertically; each task occupies one row, with the row split into two seed groups (s1 left, s2 right). Green = seed solved (≥ 0.9) at the indicated evaluation point; red = not solved (or the seed did not solve at training, in which case all of that seed's cells are red); gray (complexity only) = condition skipped because the scaled knob exceeds the $|\mathcal{V}| = 256$ vocabulary headroom. Mamba2, MLA, and Transformer are absent because they did not solve any task at any difficulty (Figure 2).

4.1 Positive Controls

The training harness and the model implementations succeed on a published benchmark. On the easiest MQAR condition of Zoology (Arora et al., 2024a), trained through PRISM's complete harness, PRISM's Mamba2 reaches 0.994, PRISM's MLA 0.991, and Zoology's attention baseline 0.998 (Table 16). The same PRISM models also solve PRISM tasks in a reduced setting with $|\mathcal{V}| = 64$, $L = 128$, and a smaller difficulty knob: the Transformer solves NIS and AKR in both seeds, MLA solves AKR, and Mamba2 solves CAR and NIS (Table 17). The failures in Figure 2 therefore reflect the conditions PRISM imposes (vocabulary size, sequence length, dense supervision, and training budget) rather than errors in the pipeline.

4.2 Softmax Attention and Supervision Density

Attention models acquire retrieval on MQAR and on MAD's recall tasks, and PRISM differs from both in how retrieval is supervised (Table 4). MQAR computes the loss only on the answers. MAD noisy recall uses dense next-token loss, but its small vocabulary makes keys repeat, so about a third of the targets carry retrieval signal. PRISM combines dense loss with one-shot bindings and a larger vocabulary, so retrieval answers make up only 0.39% of the targets on NCR-easy.

Table 4: Settings in which attention models acquire retrieval.

	MQAR	MAD noisy recall	PRISM NCR-easy
Training loss	answers only	dense next-token	dense next-token
Answers among training targets	100%	≈33%	0.39%
Vocabulary (key / value pools)	8192 (4096 / 4096)	32 (8 / 8)	256 (64 / 64)
Binding presentation	once, queried nearby	keys repeat	once, queried at the end
Length / training examples	64 / 100k	128 / 12.8k	1024 / 12.8k

To test whether this density matters, we restrict the loss to the answer positions; the data, models, training budget, and learning-rate sweep remain unchanged (Table 5). Transformer and MLA go from

chance to solving AKR-easy in both seeds (≥ 0.999), and from chance to about 0.52 on NCR-easy (see Appendix E.1 for details). Dense training is not impossible under PRISM: GDN, GDP, and RWKV-7 solve most tasks with the same objective. The architectures differ in how efficiently they acquire retrieval from sparse supervision, and sequence length also limits the Transformer (Table 20).

Table 5: Best in-distribution accuracy (seed 1 / seed 2) at the easy tier with the dense objective and with the loss restricted to answer positions. AKR reports signal accuracy.

	NCR-easy		AKR-easy	
Model	dense	answers only	dense	answers only
Transformer	0.019 / 0.021	0.527 / 0.518	0.027 / 0.086	0.999 / 0.999
MLA	0.241 / 0.058	0.529 / 0.529	0.034 / 0.024	1.000 / 1.000
Mamba2	0.020 / 0.023	0.022 / 0.021	0.025 / 0.022	0.027 / 0.030

4.3 Mamba2 and Task Scale

Mamba2 responds to none of the changes that help attention. A wider search over learning rates and weight decays finds no solve on NCR-easy (best 0.033) or AKR-easy (best 0.065; Table 18), and answer-only training leaves it at chance (Table 5). To locate where acquisition fails, we vary vocabulary size, sequence length, and binding count on NCR, keeping the models and training budget unchanged. Mamba2 solves $|\mathcal{V}| = 64$, $L = 128$, $N = 4$ in both seeds (0.999 and 0.983) but stays at chance at every setting with $|\mathcal{V}| = 256$ (see Appendix E.3 for details).

5 Discussion

We discuss the findings listed in the contributions; Appendix F gives extended analyses.

Supervision density and acquisition. Under dense next-token training, the six architectures fall into three groups. GDN, GDP, and RWKV-7 solve most tasks even though answers are a small fraction of the training targets. Transformer and MLA solve AKR-easy only when the loss is restricted to the answers (§4.2), consistent with evidence that properties of the training data affect whether attention models acquire in-context learning (Chan et al., 2022; Kawata et al., 2025). Mamba2 is rescued neither by answer-only training nor by a wider hyperparameter search, and solves NCR only at a smaller vocabulary and shorter sequences (§4.3). These groups help explain why PRISM and earlier benchmarks disagree: MQAR concentrates supervision on the answers and MAD repeats bindings within a small vocabulary, while PRISM removes both aids. Our results therefore do not contradict the associative-recall successes reported by Arora et al. (2025) and Okpekpe & Orvieto (2025); they suggest that such successes depend on how retrieval is supervised and on the scale of the task.

Dispersion structure. GDN and GDP solve CSR, MTR, and AKR at the hard tier but fail NCR-Hard, despite both having a binding count of 32; the structural prior in the first three may convert the task into a routing problem the architectures can exploit, while NCR offers no such prior.

Complexity versus length generalisation. When sequence length and the difficulty knob are each scaled by the same factor, the solved seeds lose more accuracy under complexity scaling on NCR, CAR, CSR, THR, and MTR (for example, MTR at $4\times$: 0.474 versus 0.862), comparable accuracy on NIS, and less on AKR (0.932 versus 0.759 at $2\times$). On most tasks, the learned solutions therefore remain more sensitive to the binding count they were trained on than to the noise distance.

6 Conclusion

We introduced **PRISM**, a seven-task synthetic probe suite for in-context retrieval paired with a controlled experimental protocol under which we evaluated six architectures. All models have about 4.5M parameters and share one training recipe, so the results may change at larger scale or under per-architecture tuning. Two seeds per condition expose seed-dependent acquisition but cannot estimate solve probabilities. The interventions of Section 4 are behavioural: they show where acquisition fails but not why. PRISM's random filler is less structured than natural text, and the 256-token vocabulary limits how far the difficulty knob can be scaled. Future work will address these limitations.

AI USE STATEMENT

We used generative AI tools to assist with writing, to edit the paper for readability, to search for and summarise related literature, and to suggest how results could be summarised in the text; the authors checked all numbers and conclusions against the experimental records. We did not use them to develop the methodology, design or implement the experiments, or generate or process data. The authors verified all cited work, reviewed and revised all AI-assisted text, and take full responsibility for the content of this paper.

ETHICS STATEMENT

The research uses procedurally generated synthetic integer-token sequences and involves no human subjects, no scraped data, and no pretrained models with dual-use potential. PRISM is a controlled synthetic diagnostic benchmark for in-context retrieval research; it releases only integer-token sequences with no semantic content, no privacy-sensitive data, and no pretrained models, and therefore poses no realistic privacy, fairness, or misuse risk. We have read the ICLR Code of Ethics and the work conforms with it.

REPRODUCIBILITY STATEMENT

The PRISM benchmark, training and evaluation code, and configuration files for all reported conditions are released anonymously at `https://anonymous.4open.science/r/prism_submission-5F3F`; the repository ships with a README and scripts to regenerate the synthetic data and reproduce the reported runs. The benchmark is fully specified in §2.1 and §2.2, and the test-set generation procedure in §2.3. §3.1 specifies the model configurations, and §3.2 specifies the training protocol (epochs, batch size, learning-rate sweep, seeds, and validation-accuracy thresholds) and the compute used.

REFERENCES

Cem Anil, Yuhuai Wu, Anders Andreassen, Aitor Lewkowycz, Vedant Misra, Vinay Ramasesh, Ambrose Slone, Guy Gur-Ari, Ethan Dyer, and Behnam Neyshabur. Exploring length generalization in large language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Aryaman Arora, Neil Rathi, Nikil Roashan Selvam, Róbert Csordás, Dan Jurafsky, and Christopher Potts. Mechanistic evaluation of Transformers and state space models. *arXiv preprint arXiv:2505.15105*, 2025.

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. In *International Conference on Learning Representations (ICLR)*, 2024a.

Simran Arora, Sabri Eyuboglu, Michael Zhang, Aman Timalsina, Silas Alberti, James Zou, Atri Rudra, and Christopher Ré. Simple linear attention language models balance the recall–throughput tradeoff. In *International Conference on Machine Learning (ICML)*, 2024b.

Stephanie C. Y. Chan, Adam Santoro, Andrew K. Lampinen, Jane X. Wang, Aaditya K. Singh, Pierre H. Richemond, Jay McClelland, and Felix Hill. Data distributional properties drive emergent in-context learning in Transformers. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Tri Dao and Albert Gu. Transformers are SSMs: Generalized models and efficient algorithms through structured state space duality. In *International Conference on Machine Learning (ICML)*, 2024.

DeepSeek-AI. DeepSeek-V2: A strong, economical, and efficient mixture-of-experts language model. *arXiv preprint arXiv:2405.04434*, 2024.

Benjamin L. Edelman, Ezra Edelman, Surbhi Goel, Eran Malach, and Nikolaos Tsilivis. The evolution of statistical induction heads: In-context learning Markov chains. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Omer Goldman, Alon Jacovi, Aviv Slobodkin, Aviya Maimon, Ido Dagan, and Reut Tsarfaty. Is it really long context if all you need is retrieval? towards genuinely difficult long context NLP. In *Empirical Methods in Natural Language Processing (EMNLP)*, 2024.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. In *Conference on Language Modeling (COLM)*, 2024.

Albert Gu, Karan Goel, and Christopher Ré. Efficiently modeling long sequences with structured state spaces. In *International Conference on Learning Representations (ICLR)*, 2022.

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? In *Conference on Language Modeling (COLM)*, 2024.

Greg Kamradt. Needle in a haystack – pressure testing LLMs. `https://github.com/gkamradt/LLMTest_NeedleInAHaystack`, 2023.

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are RNNs: Fast autoregressive transformers with linear attention. In *International Conference on Machine Learning (ICML)*, 2020.

Ryotaro Kawata, Yujin Song, Alberto Bietti, Naoki Nishikawa, Taiji Suzuki, Samuel Vaiter, and Denny Wu. From shortcut to induction head: How data diversity shapes algorithm selection in Transformers. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Mosh Levy, Alon Jacoby, and Yoav Goldberg. Same task, more tokens: the impact of input length on the reasoning performance of large language models. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2024.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173, 2024.

Amirkeivan Mohtashami and Martin Jaggi. Landmark attention: Random-access infinite context length for Transformers. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Eshaan Nichani, Jason D. Lee, and Alberto Bietti. Understanding factual recall in transformers via associative memories. In *International Conference on Learning Representations (ICLR)*, 2025.

Destiny Okpekpe and Antonio Orvieto. Revisiting associative recall in modern recurrent models. *arXiv preprint arXiv:2508.19029*, 2025.

Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, et al. In-context learning and induction heads. *Transformer Circuits Thread*, 2022. URL `https://transformer-circuits.pub/2022/in-context-learning-and-induction-heads/index.html`.

Bo Peng, Eric Alcaide, Quentin Anthony, Alon Albalak, Samuel Arcadinho, Stella Biderman, et al. RWKV: Reinventing RNNs for the transformer era. In *Findings of the Association for Computational Linguistics: EMNLP*, 2023.

Bo Peng, Ruichong Zhang, Daniel Goldstein, Eric Alcaide, Xingjian Du, Haowen Hou, et al. RWKV-7 "goose" with expressive dynamic state evolution. *arXiv preprint arXiv:2503.14456*, 2025.

Michael Poli, Armin W. Thomas, Eric Nguyen, Pragaash Ponnusamy, Björn Deiseroth, Kristian Kersting, Taiji Suzuki, Brian Hie, Stefano Ermon, Christopher Ré, Ce Zhang, and Stefano Massaroli. Mechanistic design and scaling of hybrid architectures. In *International Conference on Machine Learning (ICML)*, 2024.

Gautam Reddy. The mechanistic basis of data dependence and abrupt learning in an in-context classification task. In *International Conference on Learning Representations (ICLR)*, 2024.

Imanol Schlag, Kazuki Irie, and Jürgen Schmidhuber. Linear transformers are secretly fast weight programmers. In *International Conference on Machine Learning (ICML)*, 2021.

Julien Siems, Timur Carstensen, Arber Zela, Frank Hutter, Massimiliano Pontil, and Riccardo Grazzi. DeltaProduct: Improving state-tracking in linear RNNs via householder products. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Aaditya K. Singh, Stephanie C. Y. Chan, Ted Moskovitz, Erin Grant, Andrew M. Saxe, and Felix Hill. The transient nature of emergent in-context learning in Transformers. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Aaditya K. Singh, Ted Moskovitz, Felix Hill, Stephanie C. Y. Chan, and Andrew M. Saxe. What needs to go right for an induction head? a mechanistic study of in-context learning circuits and their formation. In *International Conference on Machine Learning (ICML)*, 2024.

Paul Smolensky. Tensor product variable binding and the representation of symbolic structures in connectionist systems. *Artificial Intelligence*, 46(1–2):159–216, 1990.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, pp. 5998–6008, 2017.

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

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

Songlin Yang, Bailin Wang, Yikang Shen, Rameswar Panda, and Yoon Kim. Gated linear attention transformers with hardware-efficient training. In *International Conference on Machine Learning (ICML)*, pp. 56501–56523, 2024a.

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 (NeurIPS)*, 2024b.

Songlin Yang, Jan Kautz, and Ali Hatamizadeh. Gated delta networks: Improving mamba2 with delta rule. In *International Conference on Learning Representations (ICLR)*, 2025.

Liang Zhao, Xiachong Feng, Xiaocheng Feng, Weihong Zhong, Dongliang Xu, Qing Yang, Hongtao Liu, Bing Qin, and Ting Liu. Length extrapolation of transformers: A survey from the perspective of positional encoding. In *Findings of the Association for Computational Linguistics: EMNLP*, 2024. arXiv preprint arXiv:2312.17044.

Yongchao Zhou, Uri Alon, Xinyun Chen, Xuezhi Wang, Rishabh Agarwal, and Denny Zhou. Transformers can achieve length generalization but not robustly. *arXiv preprint arXiv:2402.09371*, 2024.

A Related Work

A.1 Transformer attention and the in-context recall circuit

The softmax-attention Transformer (Vaswani et al., 2017) solves associative recall by composing two attention operations into an *induction head*: one head shifts each value token's information back to its preceding key, and a later head matches a query against the shifted keys to copy the bound value. Olsson et al. (2022) showed that this circuit emerges abruptly during training, marked by a small bump in the loss curve, and is the principal mechanism behind in-context learning in large Transformers. Singh et al. (2024) pinpointed the underlying dynamics: three subcircuits must align via gradient descent for an induction head to form, which both explains the abruptness and predicts that the timing varies across seeds. Reddy (2024) reproduced the abruptness in a small parametric model and traced it to nested nonlinearities being learned sequentially. Edelman et al.

(2024) quantified *when* the transition occurs, finding that high training-data diversity pushes its onset to a critical step count.

The data-diversity dependence has direct consequences for benchmark design. Chan et al. (2022) established that in-context learning emerges only under specific distributional properties of training data (burstiness, class diversity, Zipfian frequency) and competes with in-weights learning (IWL), in which the model memorises input-output mappings into its embeddings and MLPs. Singh et al. (2023) showed that this competition produces a double-descent learning curve: with low task diversity the model takes the IWL shortcut, and with high diversity it must form the in-context circuit, slowing initial convergence. Kawata et al. (2025) extended this to retrieval circuits specifically, showing that high diversity is required to push Transformers from shortcut solutions toward genuine induction-head behaviour. Nichani et al. (2025) provided a theoretical counterpart: a single-layer Transformer with an MLP can attain perfect factual recall, but only when its parameter count scales (up to log factors) with the number of facts to be stored, which punctures the assumption that softmax attention has unbounded recall capacity.

DeepSeek-V2's multi-head latent attention (MLA) modifies softmax attention by compressing the KV cache through a low-rank latent projection while retaining the softmax non-linearity (DeepSeek-AI, 2024). MLA preserves the induction-head mechanism and inherits its data-efficiency profile.

A.2 Sub-quadratic architectures and the convergence to delta-rule updates

Selective state-space models. Mamba and Mamba2 (Gu & Dao, 2024; Dao & Gu, 2024) introduced input-dependent state-space recurrences, replacing the time-invariant transition matrices of S4-style SSMs (Gu et al., 2022) with token-conditioned A_t, B_t. The state is multiplicatively gated and additively updated by the input. Under the state-space duality of Dao & Gu (2024), the Mamba2 update is an additive write indexed by an input-dependent key projection, lacking only the key-indexed erasure of the delta rule. Arora et al. (2025) find that Mamba solves associative recall by binding keys to values through its short convolution, whereas the recall gap analysed by Arora et al. (2024a) concerns gated-convolution models.

Linear attention with the delta rule. Linear-attention models replace softmax with a kernel feature map and accumulate an outer-product associative memory $S_t = \sum_{i \le t} \phi(k_i) v_i^\top$ (Katharopoulos et al., 2020). Schlag et al. (2021) reformulated this as a fast-weight memory and introduced the delta-rule update, which subtracts the memory's current prediction at k_t before writing v_t, making each write equivalent to a single SGD step on a least-squares objective. Yang et al. (2024b) parallelised the delta-rule update over sequence length (DeltaNet) and showed it outperforms vanilla linear attention on recall at matched state size. Yang et al. (2025) added an input-dependent gate that decays the memory between writes (Gated DeltaNet, GDN). Siems et al. (2025) replaced the gating with a product of Householder reflections per token (Gated DeltaProduct, GDP), which permits higher-rank updates per step at the cost of a more constrained update geometry. Wang et al. (2025) evaluated six linear-attention variants in a hybrid setup and argue that delta-rule architectures inject rank-1 nonlinear interactions absent from softmax attention, which is a TC^0 circuit.

Recurrent associative-memory models. RWKV (Peng et al., 2023) introduced a token-mixing recurrence with explicit decay and channel-mixing, evolving through several generations. RWKV-7 (Peng et al., 2025) adopts an associative-memory parameterisation in which the recurrent state is updated by a low-rank set of projection paths that effectively implement a delta-rule-like write. The convergence theme across these families has been articulated in several recent surveys: under appropriate parameterisation, gated linear attention, delta-rule linear attention, and modern recurrent models such as RWKV-7 can be written as variants of the same outer-product recurrence with different update geometries (Yang et al., 2024a; Wang et al., 2025).

A.3 Recall benchmarks: difficulty axes and generalisation tests

Synthetic recall benchmarks. Single-needle synthetic tests, such as Needle-in-a-Haystack (Kamradt, 2023) and PassKey retrieval (Mohtashami & Jaggi, 2023), measure whether a model can locate one piece of information in a long context. Arora et al. (2024a) introduced Multi-Query Associative Recall (MQAR), which probes the simultaneous storage of many key-value bindings and has become

the standard mechanistic recall test for sub-quadratic models. Hsieh et al. (2024) built RULER, a configurable long-context suite with multi-needle, multi-key and aggregation variants. Poli et al. (2024) extended this approach with the Mechanistic Architecture Design (MAD) pipeline: a battery of synthetic token-manipulation tasks (recall, compression, and others) used as capability unit tests, with state dimension normalised across recurrent models for fair comparison; they further show that performance on these small-scale synthetics correlates with compute-optimal scaling laws for models trained from 70M to 7B parameters, providing evidence that controlled synthetic recall tests are predictive of large-scale behaviour. Okpekpe & Orvieto (2025) re-examined recurrent models on associative recall and attribute their learnability gap to optimisation dynamics rather than representational limits, while reaffirming that Transformers solve MQAR reliably under standard training.

Difficulty axes for controllable testing. Goldman et al. (2024) propose two orthogonal axes that govern long-context task difficulty: *dispersion* (how hard the relevant information is to find) and *scope* (how much of it is needed). They show that most published benchmarks sit in the low-dispersion, low-scope corner, while the high-dispersion, high-scope regime is severely under-represented. Synthetic benchmarks can vary these axes independently: scope by raising the number of bindings to be tracked simultaneously, and dispersion by varying needle position, by interleaving distractors of varying similarity to the keys, or by partitioning the context into multiple structured sections. Levy et al. (2024) demonstrated this empirically: in NIAH-style tasks, needle position dominates raw context length as a difficulty driver.

Generalisation tests. Anil et al. (2022) and Zhou et al. (2024) showed that Transformers often fail to length-generalise on synthetic tasks even when they fit the training distribution, and that the gap is highly architecture-dependent. Zhao et al. (2024) surveyed positional-encoding interventions that close part of the length-extrapolation gap. Beyond length, *complexity generalisation*, which scales task knobs such as binding count, distractor count, or composition depth at fixed sequence length, probes whether a learned circuit operates on the structure of the task or has merely memorised a distribution.

B FULL PER-SEED RESULTS

This appendix reports the complete per-seed results: in-distribution accuracy for all six architectures (Table 6), and length- and complexity-generalisation results across all three difficulty tiers, restricted to seeds that solved the in-distribution task (≥ 0.9 best validation accuracy at training L). Seeds that did not solve at training are omitted because they stay at chance under any out-of-distribution scaling and contribute only noise. The main paper (§3.3) reports a representative subset of these numbers; the tables here are the full record.

For each generalisation axis we present three separate tables (easy, moderate, hard) so that within-difficulty patterns are easier to read. Within each table, models are separated by horizontal rules and ordered RWKV-7 / GDP / GDN; only models with at least one solved seed at the given (task, difficulty) appear. Mamba2, MLA, and the vanilla Transformer are absent from every table because they failed to solve any task at any difficulty (Table 6).

Conventions: *trL* gives the training sequence length; subsequent columns give evaluation accuracy at $L' \in \{512, 1024, 2048, 4096\}$. The metric is signal accuracy for AKR and top-level evaluation accuracy for all other tasks. In the complexity tables, *knob* names the task-specific scope parameter scaled at $2\times$ and $4\times$; $--$ marks an unavailable condition: the scaled knob exceeds the $|\mathcal{V}| = 256$ vocabulary headroom or cannot be sampled, or scaling leaves the condition unchanged (NIS-easy, where $S = 0$).

B.1 IN-DISTRIBUTION ACCURACY

Table 6: In-distribution evaluation accuracy (best validation accuracy at training L) at the easy, moderate and hard difficulty tiers, per seed, $|\mathcal{V}| = 256$.

		RWKV-7		GDP		GDN		Mamba2		MLA		Transformer	
Difficulty	Test	s1	s2	s1	s2	s1	s2	s1	s2	s1	s2	s1	s2
Easy	NCR	1.000	0.996	0.985	0.986	0.985	0.981	0.020	0.023	0.241	0.058	0.019	0.021
	CAR	0.991	0.995	0.010	0.860	0.009	0.010	0.009	0.010	0.012	0.035	0.010	0.010
	CSR	0.998	0.997	0.994	0.994	0.366	0.981	0.023	0.023	0.023	0.023	0.023	0.023
	THR	0.994	0.995	0.037	0.996	0.045	0.419	0.039	0.038	0.412	0.230	0.248	0.191
	NIS	0.997	1.000	0.020	0.999	0.458	0.080	0.018	0.018	0.024	0.061	0.025	0.019
	MTR	0.996	0.998	0.984	0.410	0.993	0.990	0.019	0.019	0.019	0.075	0.018	0.085
	AKR	1.000	1.000	0.920	0.023	0.998	0.484	0.025	0.022	0.034	0.024	0.027	0.086
Moderate	NCR	0.998	1.000	0.996	0.035	0.996	0.991	0.017	0.018	0.045	0.028	0.030	0.022
	CAR	0.979	0.984	0.790	0.009	0.009	0.009	0.013	0.009	0.009	0.009	0.009	0.009
	CSR	1.000	0.998	0.992	0.518	0.996	0.995	0.021	0.021	0.029	0.056	0.117	0.022
	THR	0.349	0.348	0.035	0.344	0.344	0.156	0.037	0.038	0.038	0.044	0.349	0.276
	NIS	0.998	0.996	0.995	0.018	0.997	0.301	0.018	0.018	0.022	0.041	0.057	0.017
	MTR	0.996	0.999	0.957	0.019	0.974	0.995	0.018	0.017	0.081	0.064	0.018	0.521
	AKR	1.000	0.999	0.989	0.995	0.024	0.995	0.021	0.023	0.061	0.025	0.025	0.030
Hard	NCR	1.000	0.996	0.468	0.019	0.021	0.017	0.018	0.019	0.031	0.028	0.036	0.019
	CAR	0.990	0.980	0.009	0.009	0.009	0.009	0.008	0.008	0.008	0.008	0.009	0.009
	CSR	1.000	0.999	0.985	0.997	0.996	0.987	0.032	0.030	0.033	0.021	0.112	0.097
	THR	0.215	0.995	0.997	0.999	0.154	0.046	0.044	0.044	0.060	0.068	0.072	0.072
	NIS	0.998	0.998	0.889	0.992	0.049	0.017	0.017	0.018	0.038	0.034	0.038	0.018
	MTR	0.995	0.997	0.995	0.996	0.018	0.993	0.021	0.018	0.017	0.031	0.062	0.411
	AKR	1.000	1.000	0.997	0.992	0.998	0.995	0.023	0.023	0.026	0.028	0.027	0.026

B.2 LENGTH GENERALISATION

Tables 7–9 report per-seed length-generalisation accuracy at evaluation lengths $L' \in \{512, 1024, 2048, 4096\}$, with one table per difficulty tier (easy, moderate, hard). The training length used for each task appears in the *trL* column; columns to its right give the test-set accuracy at the indicated L'.

Table 7: Full length-generalisation results, **easy** difficulty, solved seeds only.

Model	Test	seed	trL	@512	@1024	@2048	@4096
RWKV-7	NCR	s1	1024	0.999	1.000	0.999	0.996
RWKV-7	NCR	s2	1024	0.996	0.997	0.975	0.679
RWKV-7	CAR	s1	512	0.991	0.990	0.981	0.911
RWKV-7	CAR	s2	512	0.994	0.978	0.722	0.308
RWKV-7	CSR	s1	2048	0.996	0.997	0.998	0.997
RWKV-7	CSR	s2	2048	0.995	0.997	0.998	0.998
RWKV-7	THR	s1	1024	0.830	0.991	0.926	0.160
RWKV-7	THR	s2	1024	0.878	0.995	0.989	0.798
RWKV-7	NIS	s1	1024	0.995	0.998	0.996	0.982
RWKV-7	NIS	s2	1024	0.999	1.000	0.991	0.487
RWKV-7	MTR	s1	1024	0.994	0.994	0.964	0.584
RWKV-7	MTR	s2	1024	0.995	0.998	0.986	0.822
RWKV-7	AKR	s1	2048	0.835	0.999	1.000	0.995
RWKV-7	AKR	s2	2048	0.988	0.997	1.000	0.701
GDP	NCR	s1	1024	0.985	0.982	0.979	0.903
GDP	NCR	s2	1024	0.975	0.983	0.976	0.920
GDP	CSR	s1	2048	0.978	0.990	0.994	0.988
GDP	CSR	s2	2048	0.970	0.986	0.993	0.985
GDP	THR	s2	1024	0.990	0.994	0.956	0.538
GDP	NIS	s2	1024	0.999	0.999	0.999	0.996
GDP	MTR	s1	1024	0.978	0.982	0.976	0.939
GDP	AKR	s1	2048	0.512	0.732	0.934	0.359
GDN	NCR	s1	1024	0.985	0.981	0.984	0.968
GDN	NCR	s2	1024	0.975	0.978	0.974	0.931
GDN	CSR	s2	2048	0.955	0.973	0.978	0.969
GDN	MTR	s1	1024	0.993	0.992	0.988	0.975
GDN	MTR	s2	1024	0.986	0.989	0.983	0.967
GDN	AKR	s1	2048	0.995	0.995	0.995	0.919

Table 8: Full length-generalisation results, **moderate** difficulty, solved seeds only.

Model	Test	seed	trL	@512	@1024	@2048	@4096
RWKV-7	NCR	s1	1024	0.995	0.998	0.988	0.854
RWKV-7	NCR	s2	1024	0.999	1.000	0.989	0.819
RWKV-7	CAR	s1	512	0.981	0.978	0.962	0.835
RWKV-7	CAR	s2	512	0.985	0.936	0.586	0.257
RWKV-7	CSR	s1	2048	0.998	1.000	1.000	1.000
RWKV-7	CSR	s2	2048	0.960	0.992	0.998	0.875
RWKV-7	NIS	s1	1024	0.997	0.998	0.997	0.994
RWKV-7	NIS	s2	1024	0.996	0.996	0.983	0.869
RWKV-7	MTR	s1	1024	0.990	0.996	0.983	0.911
RWKV-7	MTR	s2	1024	0.923	0.998	0.891	0.635
RWKV-7	AKR	s1	2048	0.307	0.975	1.000	0.273
RWKV-7	AKR	s2	2048	0.039	0.023	1.000	0.643
GDP	NCR	s1	1024	0.995	0.995	0.994	0.986
GDP	CSR	s1	2048	0.982	0.990	0.991	0.989
GDP	NIS	s1	1024	0.989	0.995	0.996	0.995
GDP	MTR	s1	1024	0.952	0.959	0.903	0.834
GDP	AKR	s1	2048	0.038	0.648	0.992	0.989
GDP	AKR	s2	2048	0.273	0.550	0.995	0.975
GDN	NCR	s1	1024	0.997	0.996	0.997	0.995
GDN	NCR	s2	1024	0.990	0.990	0.990	0.972
GDN	CSR	s1	2048	0.990	0.995	0.996	0.997
GDN	CSR	s2	2048	0.993	0.994	0.994	0.994
GDN	NIS	s1	1024	0.997	0.998	0.998	0.998
GDN	MTR	s1	1024	0.969	0.973	0.973	0.968
GDN	MTR	s2	1024	0.995	0.995	0.993	0.987
GDN	AKR	s2	2048	0.719	0.960	1.000	0.977

Table 9: Full length-generalisation results, **hard** difficulty, solved seeds only.

Model	Test	seed	trL	@512	@1024	@2048	@4096
RWKV-7	NCR	s1	1024	1.000	1.000	1.000	1.000
RWKV-7	NCR	s2	1024	0.993	0.996	0.997	0.990
RWKV-7	CAR	s1	512	0.990	0.963	0.531	0.081
RWKV-7	CAR	s2	512	0.980	0.956	0.865	0.523
RWKV-7	CSR	s1	2048	1.000	1.000	1.000	1.000
RWKV-7	CSR	s2	2048	0.991	0.999	1.000	0.993
RWKV-7	THR	s2	1024	0.988	0.995	0.990	0.959
RWKV-7	NIS	s1	1024	0.999	0.999	0.999	0.997
RWKV-7	NIS	s2	1024	0.925	0.998	0.975	0.859
RWKV-7	MTR	s1	1024	0.959	0.996	0.717	0.131
RWKV-7	MTR	s2	1024	0.952	0.997	0.956	0.688
RWKV-7	AKR	s1	2048	0.867	0.938	1.000	0.074
RWKV-7	AKR	s2	2048	0.470	0.766	1.000	0.037
GDP	CSR	s1	2048	0.969	0.979	0.985	0.979
GDP	CSR	s2	2048	0.967	0.995	0.997	0.997
GDP	THR	s1	1024	0.940	0.997	0.977	0.638
GDP	THR	s2	1024	0.992	0.999	0.996	0.561
GDP	NIS	s2	1024	0.988	0.992	0.993	0.989
GDP	MTR	s1	1024	0.993	0.995	0.993	0.984
GDP	MTR	s2	1024	0.996	0.996	0.994	0.988
GDP	AKR	s1	2048	0.587	0.861	0.997	0.988
GDP	AKR	s2	2048	0.651	0.928	0.987	0.937
GDN	CSR	s1	2048	0.981	0.993	0.997	0.998
GDN	CSR	s2	2048	0.966	0.981	0.988	0.990
GDN	MTR	s2	1024	0.983	0.994	0.996	0.997
GDN	AKR	s1	2048	0.066	0.984	0.998	0.996
GDN	AKR	s2	2048	0.312	0.991	0.995	0.995

B.3 Complexity generalisation

Tables 10–12 report per-seed complexity-generalisation accuracy at scope-knob scaling factors of $2\times$ and $4\times$ relative to training, with one table per difficulty tier (easy, moderate, hard). Sequence length and vocabulary are held fixed at training values; the *knob* column names the task-specific scope parameter that is scaled.

Table 10: Full complexity-generalisation results, **easy** difficulty, solved seeds only.

Model	Test	seed	knob	train	2×	4×
RWKV-7	NCR	s1	N	1.000	0.995	0.961
RWKV-7	NCR	s2	N	0.997	0.976	0.813
RWKV-7	CAR	s1	M	0.991	0.925	0.664
RWKV-7	CAR	s2	M	0.995	0.962	0.765
RWKV-7	CSR	s1	N	0.998	0.959	0.572
RWKV-7	CSR	s2	N	0.998	0.921	0.487
RWKV-7	THR	s1	N	0.991	0.343	0.080
RWKV-7	THR	s2	N	0.995	0.275	0.044
RWKV-7	NIS	s1	S	0.998	—	—
RWKV-7	NIS	s2	S	1.000	—	—
RWKV-7	MTR	s1	N	0.994	0.955	0.535
RWKV-7	MTR	s2	N	0.998	0.916	0.426
RWKV-7	AKR	s1	D	1.000	0.999	0.941
RWKV-7	AKR	s2	D	1.000	0.997	0.946
GDP	NCR	s1	N	0.982	0.907	0.673
GDP	NCR	s2	N	0.983	0.896	0.659
GDP	CSR	s1	N	0.994	0.931	0.657
GDP	CSR	s2	N	0.993	0.942	0.697
GDP	THR	s2	N	0.994	0.205	0.016
GDP	NIS	s2	S	0.999	—	—
GDP	MTR	s1	N	0.982	0.881	0.573
GDP	AKR	s1	D	0.934	0.735	0.579
GDN	NCR	s1	N	0.981	0.925	0.723
GDN	NCR	s2	N	0.978	0.887	0.642
GDN	CSR	s2	N	0.978	0.837	0.443
GDN	MTR	s1	N	0.992	0.929	0.723
GDN	MTR	s2	N	0.989	0.805	0.433
GDN	AKR	s1	D	0.995	0.877	0.146

Table 11: Full complexity-generalisation results, **moderate** difficulty, solved seeds only.

Model	Test	seed	knob	train	2×	4×
RWKV-7	NCR	s1	N	0.998	0.977	0.783
RWKV-7	NCR	s2	N	1.000	0.926	0.316
RWKV-7	CAR	s1	M	0.979	0.809	0.323
RWKV-7	CAR	s2	M	0.984	0.877	0.491
RWKV-7	CSR	s1	N	1.000	0.997	—
RWKV-7	CSR	s2	N	0.998	0.915	—
RWKV-7	NIS	s1	S	0.998	0.999	0.998
RWKV-7	NIS	s2	S	0.996	0.996	0.996
RWKV-7	MTR	s1	N	0.996	0.819	0.264
RWKV-7	MTR	s2	N	0.999	0.837	0.406
RWKV-7	AKR	s1	D	1.000	1.000	—
RWKV-7	AKR	s2	D	0.999	0.938	—
GDP	NCR	s1	N	0.996	0.936	0.651
GDP	CSR	s1	N	0.992	0.893	—
GDP	NIS	s1	S	0.995	0.993	0.988
GDP	MTR	s1	N	0.957	0.761	0.438
GDP	AKR	s1	D	0.989	0.985	—
GDP	AKR	s2	D	0.995	0.969	—
GDN	NCR	s1	N	0.996	0.964	0.816
GDN	NCR	s2	N	0.991	0.914	0.649
GDN	CSR	s1	N	0.996	0.962	—
GDN	CSR	s2	N	0.995	0.926	—
GDN	NIS	s1	S	0.997	0.997	0.995
GDN	MTR	s1	N	0.974	0.816	0.485
GDN	MTR	s2	N	0.995	0.824	0.461
GDN	AKR	s2	D	0.995	0.888	—

Table 12: Full complexity-generalisation results, **hard** difficulty, solved seeds only.

Model	Test	seed	knob	train	2×	4×
RWKV-7	NCR	s1	N	1.000	1.000	—
RWKV-7	NCR	s2	N	0.996	0.953	—
RWKV-7	CAR	s1	M	0.990	0.830	—
RWKV-7	CAR	s2	M	0.980	0.813	—
RWKV-7	CSR	s1	N	1.000	—	—
RWKV-7	CSR	s2	N	1.000	—	—
RWKV-7	THR	s2	N	0.995	0.377	—
RWKV-7	NIS	s1	S	0.999	0.998	—
RWKV-7	NIS	s2	S	0.998	0.998	—
RWKV-7	MTR	s1	N	0.996	0.712	—
RWKV-7	MTR	s2	N	0.997	0.546	—
RWKV-7	AKR	s1	D	1.000	—	—
RWKV-7	AKR	s2	D	1.000	—	—
GDP	CSR	s1	N	0.985	—	—
GDP	CSR	s2	N	0.997	—	—
GDP	THR	s1	N	0.997	0.417	—
GDP	THR	s2	N	0.999	0.252	—
GDP	NIS	s2	S	0.992	0.988	—
GDP	MTR	s1	N	0.995	0.905	—
GDP	MTR	s2	N	0.996	0.831	—
GDP	AKR	s1	D	0.997	—	—
GDP	AKR	s2	D	0.987	—	—
GDN	CSR	s1	N	0.997	—	—
GDN	CSR	s2	N	0.988	—	—
GDN	MTR	s2	N	0.994	0.696	—
GDN	AKR	s1	D	0.998	—	—
GDN	AKR	s2	D	0.995	—	—

C SUB-METRIC DIAGNOSTICS FOR CSR, NIS, AND MTR

Table 13 reports per-seed CSR sub-metrics (Phase-1 accuracy P1, Phase-2 accuracy P2, and the Phase-2−Phase-1 delta Δ); the three large positive Δ values discussed in §3.3 appear here alongside all other conditions. Tables 14 and 15 report the analogous NIS sub-metrics (narrA, narrB, Δ) and MTR sub-metrics (early, late, Δ); for both tasks, $|\Delta|$ is small across all conditions and the tables are reported here for completeness.

Table 13: CSR sub-metrics: top-level evaluation accuracy (acc), Phase-1 accuracy (P1), Phase-2 accuracy (P2), and the Phase-2 minus Phase-1 delta (Δ). Positive Δ indicates recency bias (Phase-2 bindings retrieved better than Phase-1).

	seed 1				seed 2			
Model	acc	P1	P2	Δ	acc	P1	P2	Δ
Easy								
RWKV-7	0.998	0.999	0.997	-0.002	0.998	0.998	0.997	-0.002
GDP	0.994	0.993	0.994	+0.001	0.994	0.992	0.997	+0.005
GDN	0.366	0.025	0.706	+0.681	0.981	0.985	0.977	-0.008
Mamba2	0.023	0.025	0.022	-0.003	0.023	0.023	0.023	+0.000
MLA	0.023	0.021	0.025	+0.004	0.023	0.023	0.022	-0.001
Transformer	0.023	0.023	0.022	-0.000	0.023	0.024	0.023	-0.001
Moderate								
RWKV-7	1.000	1.000	1.000	-0.000	0.998	0.997	0.999	+0.002
GDP	0.992	0.991	0.993	+0.002	0.517	0.040	0.995	+0.955
GDN	0.996	0.999	0.993	-0.006	0.995	0.994	0.996	+0.002
Mamba2	0.021	0.021	0.022	+0.001	0.021	0.022	0.021	-0.001
MLA	0.029	0.024	0.034	+0.010	0.056	0.087	0.026	-0.060
Transformer	0.117	0.021	0.212	+0.191	0.022	0.021	0.022	+0.001
Hard								
RWKV-7	1.000	1.000	1.000	+0.000	1.000	1.000	0.999	-0.001
GDP	0.985	0.980	0.989	+0.008	0.998	0.997	0.998	+0.002
GDN	0.996	0.994	0.998	+0.004	0.987	0.991	0.982	-0.009
Mamba2	0.032	0.031	0.032	+0.001	0.030	0.029	0.031	+0.002
MLA	0.033	0.033	0.033	-0.001	0.021	0.021	0.021	-0.000
Transformer	0.112	0.104	0.120	+0.016	0.097	0.087	0.106	+0.019

Table 14: NIS sub-metrics: top-level evaluation accuracy (acc), narrative-A accuracy (narrA), narrative-B accuracy (narrB), and the narrative-B minus narrative-A delta (Δ). Δ exposes interference direction; small magnitude indicates symmetric interference handling.

	seed 1				seed 2			
Model	acc	narrA	narrB	Δ	acc	narrA	narrB	Δ
Easy								
RWKV-7	0.998	0.998	0.997	-0.001	1.000	1.000	1.000	-0.000
GDP	0.020	0.017	0.024	+0.007	0.999	0.999	1.000	+0.001
GDN	0.458	0.449	0.467	+0.018	0.080	0.050	0.110	+0.060
Mamba2	0.018	0.015	0.020	+0.004	0.018	0.019	0.018	-0.001
MLA	0.024	0.017	0.031	+0.015	0.061	0.105	0.017	-0.088
Transformer	0.025	0.035	0.015	-0.021	0.019	0.019	0.018	-0.001
Moderate								
RWKV-7	0.998	0.998	0.999	+0.001	0.996	1.000	0.993	-0.007
GDP	0.995	0.994	0.996	+0.002	0.018	0.018	0.018	+0.001
GDN	0.997	0.998	0.997	-0.000	0.301	0.296	0.305	+0.009
Mamba2	0.018	0.017	0.018	+0.001	0.018	0.019	0.017	-0.002
MLA	0.022	0.017	0.028	+0.010	0.041	0.015	0.067	+0.053
Transformer	0.057	0.052	0.062	+0.011	0.017	0.019	0.016	-0.002
Hard								
RWKV-7	0.998	0.998	0.999	+0.001	0.998	0.998	0.998	-0.001
GDP	0.889	0.898	0.881	-0.017	0.992	0.990	0.994	+0.004
GDN	0.049	0.043	0.054	+0.011	0.017	0.018	0.017	-0.001
Mamba2	0.017	0.017	0.017	+0.000	0.018	0.018	0.018	+0.000
MLA	0.038	0.059	0.017	-0.042	0.034	0.029	0.039	+0.010
Transformer	0.038	0.017	0.059	+0.042	0.018	0.019	0.018	-0.001

Table 15: MTR sub-metrics: top-level evaluation accuracy (acc), early-half topic accuracy (early), late-half topic accuracy (late), and the late minus early delta (Δ). Positive Δ indicates topic-position bias toward more-recent topics, the expected pattern for fixed-state recurrent models when older topics fall outside the effective state.

	seed 1				seed 2			
Model	acc	early	late	Δ	acc	early	late	Δ
Easy								
RWKV-7	0.996	0.995	0.998	+0.003	0.998	0.998	0.998	-0.001
GDP	0.984	0.975	0.994	+0.019	0.410	0.384	0.436	+0.052
GDN	0.993	0.990	0.996	+0.006	0.990	0.986	0.993	+0.007
Mamba2	0.019	0.021	0.017	-0.004	0.019	0.020	0.018	-0.003
MLA	0.019	0.021	0.016	-0.005	0.075	0.074	0.075	+0.001
Transformer	0.018	0.021	0.016	-0.004	0.085	0.094	0.076	-0.018
Moderate								
RWKV-7	0.996	0.996	0.996	-0.001	0.999	0.999	0.998	-0.001
GDP	0.957	0.941	0.973	+0.032	0.019	0.018	0.020	+0.002
GDN	0.974	0.975	0.973	-0.002	0.995	0.993	0.997	+0.004
Mamba2	0.018	0.016	0.019	+0.002	0.017	0.016	0.019	+0.003
MLA	0.081	0.111	0.051	-0.061	0.064	0.069	0.059	-0.010
Transformer	0.018	0.018	0.018	+0.001	0.521	0.520	0.521	+0.001
Hard								
RWKV-7	0.995	0.997	0.994	-0.003	0.997	0.997	0.997	+0.001
GDP	0.995	0.992	0.998	+0.005	0.996	0.993	0.999	+0.006
GDN	0.018	0.018	0.017	-0.001	0.993	0.993	0.993	-0.001
Mamba2	0.021	0.019	0.023	+0.004	0.018	0.017	0.019	+0.003
MLA	0.017	0.018	0.016	-0.001	0.031	0.028	0.033	+0.005
Transformer	0.062	0.063	0.061	-0.003	0.411	0.403	0.418	+0.015

D PIPELINE, IMPLEMENTATION, AND HYPERPARAMETER CONTROLS

This appendix supports §4.1 and §4.3. It tests whether the failures of Transformer, MLA, and Mamba2 could come from the training harness, the model implementations, or too narrow a hyperparameter search.

D.1 MQAR CONTROLS

We use the easiest published MQAR condition of Zoology (Arora et al., 2024a): $|\mathcal{V}| = 8192$, $L = 64$, four key-value pairs, 100,000 training and 3,000 test examples, answer-only labels, AdamW with weight decay 0.1, batch size 512, epoch-level cosine decay, a 64-epoch limit, width 256, and two layers. A run solves when its best validation accuracy reaches 0.99. Table 16 reports three checks at two learning rates, 2.15×10^{-3} and 10^{-2}. The upstream Zoology code reproduces the published Mamba result, which validates the environment. PRISM's Mamba2 solves MQAR inside Zoology's own trainer, which validates the implementation independently of our harness. The unchanged Zoology data trained through PRISM's complete harness (model builder, data loaders, loss, metrics, checkpoint selection, and training loop) are solved by PRISM's Mamba2, PRISM's MLA, and Zoology's attention configuration (one head, learned absolute positions, no MLP). Zoology has no MLA configuration, so the MLA run is an extension rather than a reproduction.

Table 16: Best MQAR validation accuracy per learning rate.

Model	Implementation	Trainer	2.15×10^{-3}	10^{-2}
Mamba (v1)	Zoology	Zoology	0.991	0.991
Mamba2	PRISM	Zoology	0.993	0.993
Mamba2	PRISM	PRISM	0.994	0.993
Attention (1 head)	Zoology	PRISM	0.998	0.983
MLA	PRISM	PRISM	0.990	0.991

D.2 REDUCED-SETTING CONTROLS

We also train the PRISM models, unchanged, with $|\mathcal{V}| = 64$, $L = 128$, and a smaller difficulty knob, using the same pipeline, dense objective, learning-rate sweep, and two seeds (Table 17). The knob values are $N = 2$ for NCR; $N_c = 2$ and $M = 2$ for CAR; $N = 2$ per phase for CSR; $K = 2$ and $N = 2$ for THR; $N = 4$ and $S = 0$ for NIS; $K = 2$ and $N = 2$ for MTR; and $D = 1$ for AKR. Because vocabulary, length, and scope change together, this setting shows that the models can solve the tasks but does not isolate which factor matters; the factorial of Appendix E.3 (Table 20) separates them for NCR, with Mamba2 and the Transformer.

Table 17: Best in-distribution accuracy (seed 1 / seed 2) in the reduced setting.

Model	NCR	CAR	CSR	THR	NIS	MTR	AKR
Transformer	0.763 / 0.936	0.526 / 0.525	0.757 / 0.964	0.764 / 0.770	1.000 / 0.999	0.839 / 0.998	0.998 / 1.000
MLA	0.758 / 0.880	0.524 / 0.533	0.996 / 0.792	0.763 / 0.764	0.748 / 0.936	0.949 / 0.838	0.998 / 1.000
Mamba2	0.956 / 0.730	0.995 / 0.967	0.424 / 0.094	0.145 / 0.148	0.999 / 0.997	0.997 / 0.661	0.995 / 0.076

D.3 MAMBA2 HYPERPARAMETER SEARCH

For the Mamba2 of Table 3, trained with the dense objective, we search five learning rates (10^{-2}, $3{\times}10^{-3}$, 10^{-3}, $3{\times}10^{-4}$, and 10^{-4}), three weight decays (0, 0.01, and 0.1), and two seeds on NCR-easy and AKR-easy, 30 runs per task. No run solves. Table 18 lists the best run for each task and seed.

Table 18: Best Mamba2 run in the hyperparameter search for each task and seed. For reference, Mamba2 reaches 0.020–0.025 on these tasks with the standard learning-rate sweep (Table 6).

Task	Seed	Best accuracy	Learning rate	Weight decay
NCR-easy	1	0.033	10^{-2}	0.01
NCR-easy	2	0.022	$3{\times}10^{-3}$	0.1
AKR-easy	1	0.048	$3{\times}10^{-3}$	0.1
AKR-easy	2	0.065	10^{-3}	0.01

E INTERVENTION DETAILS

This appendix gives the setups behind §4.2 and §4.3, and the full results of the dense MQAR and factorial experiments.

E.1 ANSWER-ONLY TRAINING ON PRISM

The answer-only runs of Table 5 use the easy-tier generators of NCR ($|\mathcal{V}| = 256$, $L = 1024$, $N = 4$) and AKR ($|\mathcal{V}| = 256$, $L = 2048$, $D = 3$), 12,800 training examples, 100 epochs, the learning-rate sweep of §3.2, the same two seeds, and the models of Table 3. The only change is the loss mask, which restricts the cross-entropy to the answer positions of the query block; evaluation is unchanged.

Table 5 reports the results. On AKR-easy, answer-only training raises Transformer and MLA from near chance under the dense objective (at most 0.086) to at least 0.999 in both seeds. On NCR-easy, both models reach about 0.52 (0.518 to 0.529) but do not solve the task; under the dense objective, the Transformer is at chance, while MLA reaches 0.241 in seed 1 and 0.058 in seed 2. Mamba2 stays near chance on both tasks under either objective (at most 0.030). Restricting the loss to the answers is therefore sufficient for the softmax-attention models to acquire AKR-easy but not NCR-easy, and it does not help Mamba2. For NCR, the factorial of Appendix E.3 shows that sequence length also limits the Transformer.

E.2 DENSE TARGETS ON MQAR

To test whether the objective matters on a benchmark where acquisition normally succeeds, we train PRISM's Mamba2 in Zoology's MQAR setup (Appendix D.1) and change only the training targets, from answer-only labels to next-token targets at every non-final position; the generator, trainer, optimiser, schedule, budget, and answer-only evaluation are unchanged. Table 19 reports three initialisations (Zoology's default seed 123, used in Table 16, and PRISM's seeds 1 and 2) at the two learning rates that solve MQAR. All six answer-only runs solve; no dense run reaches 0.99 (best 0.948 to 0.972). Dense training still acquires substantial retrieval, so the effect is a consistent reduction rather than a collapse to chance.

Table 19: Best MQAR validation accuracy of PRISM's Mamba2 with answer-only and dense next-token targets.

	2.15×10^{-3}		10^{-2}	
Seed	answers only	dense	answers only	dense
123	0.993	0.972	0.993	0.955
1	0.991	0.954	0.992	0.964
2	0.991	0.962	0.992	0.948

E.3 FACTORIAL ON NCR

The factorial of Table 20 varies vocabulary size ($|\mathcal{V}| \in \{64, 256\}$), sequence length ($L \in \{128, 1024\}$), and binding count ($N \in \{2, 4\}$) on NCR, with PRISM's Mamba2 and Transformer, the dense objective, and the training budget and learning-rate sweep of §3.2; reported values are the best in-distribution accuracy over the sweep.

Mamba2 solves $|\mathcal{V}| = 64$, $L = 128$, $N = 4$ in both seeds (0.999 and 0.983) but stays at chance at every setting with $|\mathcal{V}| = 256$, and at $|\mathcal{V}| = 64$ increasing the length to 1024 removes robust solving. The binding count is not the limiting factor: at $|\mathcal{V}| = 64$ and $L = 128$, Mamba2 does at least as well with $N = 4$ as with $N = 2$. The Transformer has a different boundary: it reaches substantial accuracy at $L = 128$ for both vocabulary sizes but fails at nearly every setting with $L = 1024$, so length is its main barrier. PRISM's easy tier combines the larger vocabulary with long sequences and therefore lies beyond both boundaries under this budget. These results cover one task, two values per factor, and two seeds, and they describe where Mamba2 fails without explaining why. Mamba2's solve at $|\mathcal{V}| = 64$, $L = 128$, $N = 4$ is sensitive to the learning rate: seed 2 reaches 0.983 at 10^{-3}, while 5×10^{-4} and 10^{-4} remain near chance.

Table 20: NCR accuracy (seed 1 / seed 2) as vocabulary size $|\mathcal{V}|$, sequence length L, and binding count N vary, with PRISM's models, dense objective, and training budget. NCR-easy corresponds to $|\mathcal{V}| = 256$, $L = 1024$, $N = 4$.

		Mamba2		Transformer	
$\lvert\mathcal{V}\rvert$	L	$N = 2$	$N = 4$	$N = 2$	$N = 4$
64	128	0.956 / 0.730	0.999 / 0.983	0.763 / 0.936	0.573 / 0.557
64	1024	0.074 / 0.660	0.079 / 0.078	0.072 / 0.508	0.073 / 0.073
256	128	0.023 / 0.022	0.020 / 0.020	0.701 / 0.790	0.522 / 0.515
256	1024	0.022 / 0.023	0.020 / 0.023	0.025 / 0.022	0.019 / 0.021

F EXTENDED DISCUSSION OF FINDINGS

We restate three findings discussed in §5 with their per-seed evidence; the evidence on supervision density is in §4 and Appendix E. We then catalogue three additional findings: a recency pattern in the CSR sub-metrics, and two dissociations between models within a mechanism family.

F.1 MAJOR FINDINGS

F.1.1 UNDER DENSE TRAINING, THE ARCHITECTURES FALL INTO THREE GROUPS

Under this matched-compute, small-scale regime with the dense objective, Mamba2, MLA, and the vanilla Transformer solve no task at any difficulty, RWKV-7 solves nearly every task, and GDN and GDP solve task-dependent subsets. Section 4 examines why the softmax-attention models and Mamba2 fail. RWKV-7 dominates the comparison overall (Table 6): it solves all seven tasks at easy; at moderate it solves six, with THR failing in both seeds (0.349 and 0.348); and at hard it solves all seven, although THR in one seed only (s1 0.215, s2 0.995).

F.1.2 THE STRUCTURED-VERSUS-UNSTRUCTURED DISPERSION AXIS DISSOCIATES MODELS THAT THE DIFFICULTY KNOB ALONE DOES NOT

GDN and GDP solve CSR, MTR, and AKR at the hard tier but fail NCR-Hard, even though the unstructured task is no larger in scope. NCR-Hard plants 32 bindings uniformly in a noise body, and both architectures fail in every seed, three of four at chance (GDP s1=0.468, s2=0.019; GDN s1=0.021, s2=0.017). At matched scope, the structurally-primed tasks are solved in all but one seed: MTR-Hard, with 8 contiguous topic segments carrying 32 bindings in total (GDP s1=0.995, s2=0.996; GDN s1=0.018, s2=0.993, with the bimodal seed solving), and AKR-Hard, with 32 late distractors against a single early signal (GDP s1=0.997, s2=0.992; GDN s1=0.998, s2=0.995). The strongest contrast is CSR-Hard, which carries 64 bindings in total (32 simultaneously held bindings per phase, drawn from disjoint vocabulary pools) and is nevertheless solved by every seed of both architectures (GDP s1=0.985, s2=0.997; GDN s1=0.996, s2=0.987), a far better outcome than NCR-Hard despite a higher total binding count. One interpretation is that the structural prior in CSR, MTR, and AKR turns retrieval into a routing problem the architectures can exploit, while NCR offers no such prior. The dissociation is specific to the hard tier: NCR is solved by both architectures at easy (GDN s1=0.985, s2=0.981; GDP s1=0.985, s2=0.986) and at moderate is solved by both GDN seeds (s1=0.996, s2=0.991) and one GDP seed (s1=0.996, s2=0.035), suggesting that the dispersion axis separates these models only once scope is large enough to expose the difference.

A weaker form of the asymmetry appears in the Transformer, which solves no task: best Transformer NCR accuracy across all tiers is 0.036 (s1 hard), flat at chance, while Transformer reaches measurable partial credit on the structurally-primed tasks at moderate and hard, including MTR-moderate s2=0.521, MTR-hard s2=0.411, and CSR-hard s1=0.112 / s2=0.097. The MTR-moderate s2 sub-metrics are symmetric across early/late topic halves (0.520 / 0.521, Table 15), so the elevation reflects partial topic-segment routing rather than recency. The CSR-hard elevation also shows no recency pattern (P2$-$P1 $= +0.016$ and $+0.019$, Table 13), unlike Transformer CSR-moderate s1 (§F.2.1).

F.1.3 COMPLEXITY GENERALISATION DEGRADES MORE THAN LENGTH GENERALISATION ON MOST TASKS

Comparing seeds that were evaluated on both axes at the same scaling factor, complexity accuracy falls further than length accuracy on NCR (0.699 versus 0.911 at $4\times$), CAR (0.561 versus 0.813 at $4\times$), CSR (0.928 versus 0.979 at $2\times$), THR (0.311 versus 0.972 at $2\times$), and MTR (0.474 versus 0.862 at $4\times$); it is comparable on NIS (0.994 versus 0.964 at $4\times$, where a single seed drives the length drop) and smaller on AKR (0.932 versus 0.759 at $2\times$). On most tasks, the learned solutions therefore remain more sensitive to the binding count they were trained on than to the noise distance. AKR behaves differently under length scaling. Its models are trained at $L = 2048$, and accuracy drops both for shorter and for longer sequences (Tables 7–9). At $L' = 512$, 13 of the 15 solved seeds fall below the 0.9 threshold, across all three architectures. At $L' = 4096$, the drop depends on the architecture: five of six RWKV-7 seeds fall below the threshold, but only one of nine GDP and GDN seeds does.

F.2 ADDITIONAL FINDINGS

Three further findings are not discussed in the main text. The first concerns the CSR sub-metrics. The other two dissociate models *within* a family: they distinguish GDN from GDP and isolate CAR as a task that GDN and GDP do not solve under this protocol while RWKV-7 does.

F.2.1 THE RECENCY BIAS PATTERN OBSERVED ON CONTEXT-SHIFT RETENTION APPEARS ACROSS MECHANISMS

The CSR sub-metrics (Table 13) show that when models partially learn CSR, Phase-1 accuracy stays near chance (0.021 to 0.040) while Phase-2 accuracy is higher (0.212 to 0.995). This pattern appears in both delta-rule (GDN-easy s1, GDP-moderate s2) and softmax-attention (Transformer-moderate s1) models, suggesting it reflects something structural about the context-shift task rather than a delta-rule-specific failure. The NIS and MTR sub-metric tables (Appendix C) show no such asymmetry: $|\Delta|$ is small in all conditions, so at the scope (S) and topic count (K) tested here these sub-metrics do not separate models.

F.2.2 WITHIN DELTA-RULE LINEAR ATTENTION, GDN AND GDP DISSOCIATE BY TASK DIFFICULTY

GDN and GDP are both gated delta-rule linear-attention models with matched parameter budgets (Table 3), differing in update structure. Their solve patterns diverge in opposite directions depending on difficulty. At the hard tier, GDP shows a higher ceiling on the structurally hardest tasks: THR-Hard (GDP s1=0.997, s2=0.999 vs. GDN s1=0.154, s2=0.046), NIS-Hard (GDP s1=0.889, s2=0.992 vs. GDN s1=0.049, s2=0.017), and MTR-Hard (GDP 0.995 / 0.996 vs. GDN bimodal 0.018 / 0.993). On easier conditions the direction reverses: GDN is more seed-reliable on tasks both architectures can in principle solve: NCR-Moderate (GDN 0.996 / 0.991 vs. GDP bimodal 0.996 / 0.035), CSR-Moderate (GDN 0.996 / 0.995 vs. GDP s1-only 0.992 / 0.518), and MTR-Moderate (GDN 0.974 / 0.995 vs. GDP bimodal 0.957 / 0.019). Two models from the same family can therefore differ markedly in which conditions they solve.

F.2.3 CAR RESISTS DELTA-RULE LINEAR ATTENTION BUT IS SOLVED BY RWKV-7 AT EVERY DIFFICULTY

RWKV-7 solves CAR cleanly at every difficulty tier (Easy s1=0.991, s2=0.995; Moderate s1=0.979, s2=0.984; Hard s1=0.990, s2=0.980). GDN sits at chance ($\approx 1/128$ given the value-pool size of $|\mathcal{V}|/2 = 128$) at every tier (0.009 / 0.010, 0.009 / 0.009, and 0.009 / 0.009 for Easy, Moderate, and Hard, respectively). GDP only ever reaches partial credit and never crosses the 0.9 solved threshold on any seed: Easy s2=0.860 (s1=0.010), Moderate s1=0.790 (s2=0.009), Hard 0.009 / 0.009. This is a within-family dissociation: all three architectures are sub-quadratic linear-attention or recurrent associative-memory mechanisms, yet only RWKV-7 solves CAR. CAR is also the one structurally-primed task whose dispersion prior (composite-key triples clustered around per-cluster anchors) does *not* translate into a solve for GDN and GDP, distinguishing it from CSR, MTR, and AKR (§F.1.2). What CAR uniquely demands is composite-key resolution: the same key token bound to different values depending on the anchor preceding it. All three architectures maintain outer-product associative-memory state with delta-rule-like writes (§A.2); they differ in the geometry of the per-token update: GDN's single gated delta step, GDP's product of Householder reflections, and RWKV-7's low-rank multi-path projection.

F.3 INTERPRETATION: PER-EPISODE RESAMPLING AND IN-CONTEXT STATE UPDATES

The cross-architecture solve patterns documented in §F.1.1 are empirical; this subsection offers an interpretive frame for them rather than new evidence. PRISM resamples key tokens, value tokens, and key-value mappings freshly per episode (§2.1), so no token-ID can carry role information across episodes. Retrieval must therefore be solved *in-context*, within a single forward pass, rather than stored in the model weights. Architectures whose per-token state update writes each binding directly into the state may need less training signal to acquire retrieval than architectures whose retrieval solution must first be learned from the training data.

Delta-rule and RWKV-7 implement per-token associative-memory writes. The gated delta-rule update used by GDN and GDP performs, at each token, what Schlag et al. (2021) characterise as a single SGD step on a least-squares associative-memory objective. RWKV-7's low-rank multi-path projection implements a delta-rule-like write under the same template (§A.2). Under this view, the per-token state update already performs the in-context binding, so less must be learned before

retrieval becomes available. This view is consistent with the broad solve coverage of GDN, GDP, and RWKV-7 under the dense objective.

Transformer and MLA acquire retrieval when supervision is concentrated on the answers. Softmax attention is known to solve associative recall with induction heads (Olsson et al., 2022), which emerge through a phase transition whose timing depends on the training data (Chan et al., 2022; Singh et al., 2023; Kawata et al., 2025; Edelman et al., 2024); MLA keeps the softmax non-linearity (DeepSeek-AI, 2024). Under PRISM's dense objective, answers are 0.39% of the training targets on NCR-easy (Table 4), and restricting the loss to the answers lets both models solve AKR-easy (Table 5). This is consistent with the dependence on training data reported in this literature, but we do not test whether induction heads form. On NCR-easy, answer-only training reaches only about 0.52, and the factorial of Appendix E.3 shows that sequence length also limits the Transformer.

Mamba2's failure is not explained by a missing key-indexed write. The selective state-space recurrence is multiplicatively gated and additively updated by the input (Gu & Dao, 2024; Dao & Gu, 2024). Under the state-space duality of Dao & Gu (2024), the Mamba2 update is an additive write indexed by an input-dependent key projection, lacking only the key-indexed erasure of the delta rule. Arora et al. (2025) find that Mamba solves associative recall by binding keys to values through its short convolution, and because PRISM places each value next to its key, this route is available here. The state-size capacity argument can also be ruled out: as established in §3.1, Mamba2's per-head SSM hidden state is comfortably larger than the binding count required at any difficulty tier. We therefore describe Mamba2's failure only behaviourally (§4.3).