PCAF: BOUNDED EXACT-SUCCESSOR MEMORY FOR EFFICIENT LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Efficient causal models compress long histories, which can erase token-successor associations. We study PCAF-Exact, a bounded per-key memory that retains recent observed successors and mixes their sparse vote with a causal model's prediction. Capacity-matched experiments show that exact memory improves PG-19 perplexity across three model scales; at 40M/64K, PPL falls from 129.72 to 114.54. A parameter-free reciprocal-recency reader retains much of this benefit. Under equal record storage, continuous retrieval gives lower perplexity, while exact lookup reads about 2.3× faster; measured teacher-forced pipelines show a smaller speed advantage. A compact hybrid improves on a payload-matched continuous reader across 19 PG-19 test books at both evaluated scales. Same-target tests show gains through 128K context. Together, these results position bounded exact-successor memory as an efficient complement to contextual retrieval for repeated transitions.

1 INTRODUCTION

A history of the form $A, B, \ldots, A$ creates a prediction opportunity: the previously observed successor B may be useful again. Efficient recurrent and state-space models must represent this association within a compressed state. Attention and representation-based retrieval can instead revisit individual records, but doing so introduces storage and search costs. Associative recall has consequently become an important lens for studying efficient language models (Arora et al., 2024a;b; Yang et al., 2025; Lahoti et al., 2026).

There are two complementary questions in this tradeoff. *Access* determines which observations survive and can be consulted; *weighting* determines how much probability those observations receive. A scorer operates over its available candidates, so retrieval coverage sets which successors can receive memory probability. Exact retention is especially useful when repeated keys recur with predictive successors; when a key has conflicting successors, the reader's weighting also matters. Separating access from weighting helps identify where a small memory contributes and where broader retrieval is useful.

We investigate this separation with PCAF-Exact, the exact-key form of the Parallel Causal Associative Field framework (Ahmed, 2026). Each key retains its latest K *completed* transitions. Occurrences of other keys do not evict these records, so retention depends on same-key recurrence rather than elapsed token distance. The default reader uses reciprocal occurrence-rank weights and a calibrated constant mixture coefficient. It has no trainable parameters, requires no hidden-state features in its records, and leaves the backbone unchanged when no key matches. Learned scoring, pruning, and contextual gating are evaluated as alternatives, not prerequisites.

Classical and neural cache models already use observed continuations through copying and interpolation (Kuhn & De Mori, 1990; Grave et al., 2017; Merity et al., 2017). Building on this line of work, we characterize a bounded exact-access interface and separate its retention, readout, and coverage. Three findings organize the study.

Exact access is useful with simple weighting. Independently trained capacity-matched controls isolate improvements beyond additional model parameters. A parameter-free cache attached to separately trained reader-free backbones recovers much of the improvement; in a co-trained ablation,

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.

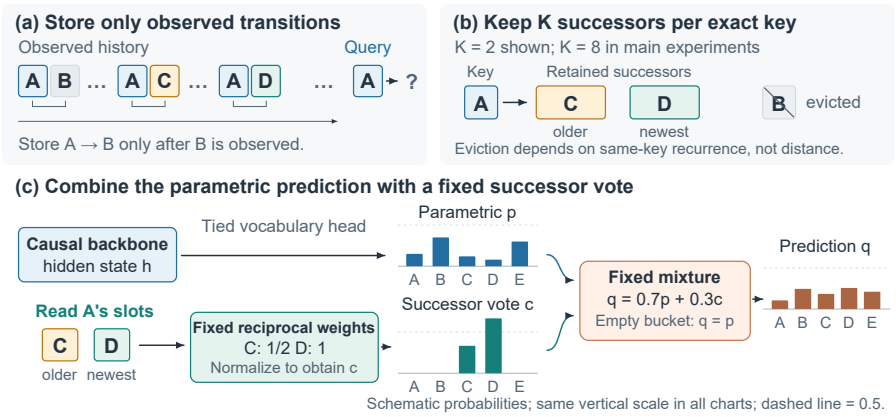


Figure 1: **Access first, weighting second.** A transition becomes readable only after its successor is observed. Each key retains its latest K completed records, independent of intervening occurrences of other keys. Reciprocal-rank voting then supplies sparse output evidence. The diagram uses $K = 2$ and an illustrative mixture weight of 0.3; experiments use $K = 8$ unless stated otherwise and report their own calibration choices.

fixed reciprocal-rank weighting and learned ranking have similar estimated NLL under the reported intervals (Section 4).

Retrieval coverage shapes the quality–cost tradeoff. At matched bytes, a continuous reader examines thousands of records and obtains better perplexity, while exact access has lower read cost. A target-level decomposition and an all-history same-key analysis show how missing target support and per-key retention each contribute to this tradeoff (Section 5).

Exact and contextual access can complement one another. Two separately specified hybrid studies test this directly. One obtains consistent test-book quality gains at matched declared payload; the other measures reader-path cost and the contribution of the exact branch. Longer-training and pretrained experiments show how these effects vary across model and training settings (Sections 6–7).

2 Bounded Exact-Successor Memory

2.1 Causal Records and a Parameter-Free Readout

At input position t, the observed prefix is $x_{1:t}$ and the target is $y_t = x_{t+1}$. A causal backbone produces h_t and a tied head gives

$$p_{\theta,t}(y) = \mathrm{softmax}(W\,\mathrm{RMSNorm}(h_t))_y\,. \tag{1}$$

A record at position i has key x_i and value $v_i = x_{i+1}$. It is committed only when v_i is observed. Let U_t contain the latest K positions satisfying $i < t$ and $x_i = x_t$. Thus every returned successor belongs to the observed prefix. Commit-before-query makes the transition (x_{t-1}, x_t) available without exposing the unobserved x_{t+1}.

For a nonempty inventory, let $r_t(i) = 1$ denote its newest record. The default vote and prediction are

$$c_t(y) = \frac{\sum_{i\in U_t} r_t(i)^{-1}\mathbf{1}[v_i = y]}{\sum_{i\in U_t} r_t(i)^{-1}}, \tag{2}$$

$$g_t = \lambda\mathbf{1}[U_t \neq \varnothing], \qquad q_t(y) = (1-g_t)p_{\theta,t}(y) + g_t c_t(y), \quad 0 \le \lambda < 1. \tag{3}$$

All retained records participate. When U_t is empty, $c_t = 0$ and $g_t = 0$. The policy uses no learned projections, scorer, gate, or pruning. The scalar λ is calibrated on the declared development data and then held fixed for evaluation. Algorithm 1 specifies the online interface.

Algorithm 1 One causal step of the fixed exact reader

Require: Observed token x_t, backbone distribution $p_{\theta,t}$, capacity K, coefficient λ
Maintain request-local deques $D[a]$ of at most K completed records
if $t > 1$ **then**
    Append $(t-1, x_t)$ to $D[x_{t-1}]$; evict oldest if $|D[x_{t-1}]| > K$
end if
Read $D[x_t]$ newest first to form U_t
if $U_t = \varnothing$ **then**
    return $p_{\theta,t}$
end if
Accumulate $1/r$ into the successor token at rank r and normalize
return $(1-\lambda)p_{\theta,t} + \lambda c_t$

2.2 Retention and the support boundary

Within a reset-free history, an eligible record survives exactly when fewer than K later completed occurrences share its key:

$$i \in U_t \quad \Longleftrightarrow \quad x_i = x_t,\ i < t,\ \left|\{j : i < j < t,\ x_j = x_i\}\right| < K. \tag{4}$$

For example, with $K = 2$, $A \to B$ survives any gap containing no A and one later transition $A \to C$. A second later transition $A \to D$ evicts it. The memory budget is therefore determined by recurrence of the key, rather than elapsed token distance.

For the actual target, write $p = p_{\theta,t}(y_t) > 0$, $c = c_t(y_t)$, and $g = g_t$. The NLL improvement over the backbone is

$$G_t = \log \frac{q_t(y_t)}{p_{\theta,t}(y_t)} = \log\left[1 + g\left(\frac{c}{p} - 1\right)\right]. \tag{5}$$

For $g > 0$, the sign is determined by $c - p$. If a nonempty inventory lacks the target, then $c = 0$ and the target probability is multiplied by $1 - g$. Reweighting a fixed inventory leaves its candidate support unchanged. This identity motivates measuring *target support*, not just matching-key coverage, and using calibrated mixing rather than unconditional copying. Target-defined groups are used for analysis and do not route inference-time queries.

2.3 Learned variants and execution

The learned variant retains the same inventory but adds normalized 16-dimensional query/key features, cosine scoring, recency, pruning, and a contextual gate. Its training recipe includes a zero-forward responsibility-completion correction. The correction restores the direct target-log-probability derivative; its component ablations are reported in Appendices A.4 and D. We treat this learned package as an alternative readout, while the central method uses the parameter-free rule above.

For fixed vocabulary V, batch size B, and K, dense decoding requires at most BVK records and $O(K)$ lookup-and-vote work per token. The token-only int64 record layout uses $16BVK$ bytes before counters and pending state; sparse allocation stores observed keys. Parallel evaluation builds a causal exact index by radix-sorting fixed-width token/position keys and gathering K predecessors, giving linear work for the pinned finite-width implementation. Allocated record payload, populated state, temporary GPU allocation, and process memory are reported separately (Appendix A).

3 Experimental design

We use WikiText-103, PG-19, and pinned RedPajama arXiv/GitHub corpora (Merity et al., 2017; Rae et al., 2020; Weber et al., 2024). The primary from-scratch comparisons train 40M and 125M models for 100M optimizer tokens and 350M models for 200M: approximately 2.5, 0.8, and 0.57 tokens per parameter. These are *early-training* comparisons. Each cell has three fixed training seeds. The backbone combines projected causal convolution and four Gated-Delta blocks; it is held fixed within reader-attribution comparisons. Table 7 records actual checkpoint dimensions and Appendix B gives data and optimization details.

Table 1: **Primary learned-reader attribution.** Three training seeds per cell. Δ_M is exact-minus-matched NLL; G is same-checkpoint reader-off-minus-exact NLL. Brackets are descriptive paired 95% seed intervals. Negative Δ_M and positive G favor exact memory.

Setting	Exact PPL	Matched PPL	Δ_M [interval]	G [interval]
40M Wiki / 8K	21.89	22.47	$-.026$ $[-.060, +.007]$	$+.030$ $[+.023, +.036]$
40M arXiv / 8K	6.10	6.49	$-.062$ $[-.110, -.014]$	$+.069$ $[+.053, +.084]$
40M code / 8K	2.94	3.00	$-.019$ $[-.095, +.057]$	$+.063$ $[+.039, +.087]$
40M PG / 64K	114.54	129.72	$-.124$ $[-.178, -.071]$	$+.124$ $[+.108, +.140]$
125M PG / 32K	55.29	57.38	$-.037$ $[-.043, -.031]$	$+.041$ $[+.037, +.044]$
350M PG / 8K	35.07	37.11	$-.057$ $[-.061, -.052]$	$+.050$ $[+.030, +.070]$

Three controls answer different questions. **Reader-off** disables the reader on the same checkpoint. **No-reader** independently trains the backbone and head. **Matched no-reader** independently trains a near-equal-parameter model with a position-wise adapter instead of the reader. Separately attaching fixed exact and continuous caches to these reader-free backbones controls dependence on reader-co-trained features. Cross-architecture Transformer and Mamba-3 comparisons contextualize complete systems; memory attribution comes from the within-backbone controls.

PPL is $\exp(\text{mean NLL})$. Primary training comparisons use descriptive paired Student-t intervals over three seed means. Same-target and hybrid studies average checkpoint contrasts within each document and resample documents, so intervals reflect document-level variation conditional on the fixed checkpoints. Cache choices are selected on the corresponding development partitions. Several follow-ups reuse previously evaluated corpora or documents, as detailed in their protocols.

The compact test-book hybrid and the projected, four-bit hybrid are different configurations. Their evaluation banks, payloads, and timing scopes are never combined into one quality–latency result. All systems measurements use one RTX 5090. A teacher-forced whole-window pipeline, an incremental decoder, and an isolated reader are different measured workloads. Appendix B.4 provides a study registry to make these distinctions explicit.

4 What does exact access add?

4.1 Predictive value beyond additional capacity

Exact memory improves on the capacity-matched control in every PG-19 seed at all three operating points (Table 1). At 40M/64K, PPL falls from 129.72 to 114.54; at 125M/32K and 350M/8K, it falls from 57.38 to 55.29 and from 37.11 to 35.07. The matched arXiv comparison also favors exact memory in every seed. WikiText and code show favorable mean differences, with wider intervals that include zero. Disabling the reader increases NLL in every primary cell. Together, these comparisons support a reader contribution within the declared training regimes; cross-architecture results are reported separately as whole-system comparisons.

4.2 How much learning does the reader need?

A parameter-free reciprocal-recency reader attached to three independently trained, capacity-matched reader-free 40M backbones reduces PG-19/64K PPL from 129.47 to 116.74. Separately co-trained learned readers reach 114.55 on the same report bank. The simple-minus-learned NLL difference is $+.01889$ with interval $[-.01697, +.05476]$ (Table 11). This complementary estimate uses separately reconstructed reader-free replicas and checkpoint reevaluation and is reported alongside the primary measurements.

A complementary co-training ablation keeps the learned gate and objective, but replaces learned ranking and pruning with reciprocal-rank weights. Its fixed-minus-learned NLL is $-.00195$ $[-.02593, +.02203]$. This experiment isolates scoring complexity, whereas the independently attached cache also removes learned gating and reader co-training. The intervals overlap zero; taken together, the studies show that a simple readout retains much of the measured benefit without the full learned package.

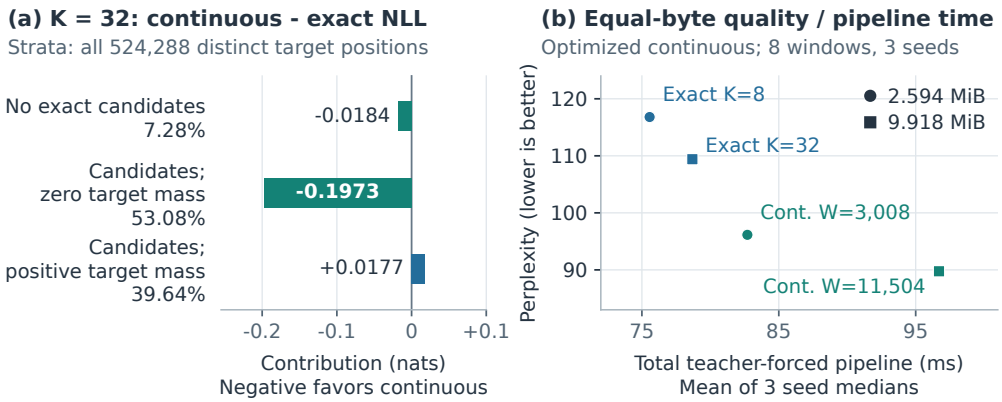


Figure 2: **Target support and the quality–cost tradeoff.** Left: contributions to continuous-minus-exact NLL for $K = 32$ at matched storage. Broader retrieval often supplies target tokens outside the exact inventory; groups are used to explain the aggregate difference. Right: perplexity against complete teacher-forced evaluation-pipeline time for the optimized comparisons in Table 2. Times are means of three seed medians.

Table 2: **Matched-storage quality and optimized evaluation cost.** The same three reader-free backbones and eight PG-19/64K windows are used in every row. Record payload excludes temporary tensors. Reader time includes index construction and mixture scoring; pipeline time additionally includes the backbone and vocabulary target probabilities. The reported workload is teacher-forced evaluation, distinct from autoregressive serving.

Reader	Records/query	PPL	Record MiB	Reader ms	Pipeline ms
Exact $K = 8$	≤ 8	116.80	2.594	3.83	75.56
Continuous $W = 3{,}008$	$\leq 3{,}008$	96.15	2.593	8.80	82.70
Exact $K = 32$	≤ 32	109.41	9.918	9.98	78.69
Continuous $W = 11{,}504$	$\leq 11{,}504$	89.76	9.918	22.64	96.70

The 24-model component cohort reinforces that distinction. Reducing K from eight to two increases NLL by .04694; increasing it to sixteen lowers NLL by .02526, with both intervals excluding zero. Removing pruning improves every seed's point estimate, while the aggregate interval includes zero. The ordinary and completed objectives show no stable separation across the two reported cohorts (Appendix D). In these experiments, access capacity is the more consistent design lever.

4.3 Explicit bindings isolate the access mechanism

On supervised atomic-token MQAR, the learned exact reader reaches 99.91% per-query accuracy at 8K after training at 1K. The retrieval-tuned Transformer fits the 1K condition (100.00%) and reaches 24.59% at 8K, distinguishing in-distribution fitting from extrapolation. A direct latest-successor rule reaches 100% on the separate unambiguous binding diagnostics. These tasks isolate exact binding retrieval and complement evaluations of natural-language reasoning and question answering. In the primary 32K multikey transfer study, one learned-reader seed reaches 9.77% complete-sequence accuracy even though the correct record remains present and selected. The reduced gate points to probability mixing, rather than record retention, as the relevant factor (Appendix J).

5 Coverage determines the quality–cost tradeoff

5.1 Matched bytes and matched fan-out answer different questions

At matched storage, continuous retrieval has lower PPL, while exact reading is about 2.3× faster and the full pipeline is 1.09–1.23× faster (Table 2). Peak allocation is 1,258 MiB in every row.

Table 3: **Hybrid quality under two separate protocols.** Δ is hybrid-minus-continuous NLL. Both studies use three fixed checkpoints at each scale; intervals resample documents after averaging within-document checkpoint contrasts. The compact study is payload matched and quality only. The projected study has a different reader, bank, and resource comparison.

Study	Scale	Hybrid PPL	Continuous PPL	Δ [interval]
Compact, 19 test books	40M	55.581	56.152	$-.01022$ $[-.01426, -.00703]$
95% intervals	350M	28.017	28.118	$-.00362$ $[-.00449, -.00281]$
Projected, 32 train-tail books	40M	53.258	53.477	$-.00410$ $[-.00588, -.00262]$
97.5% intervals	350M	28.932	28.952	$-.00072$ $[-.00146, +.00005]$

The continuous implementation was selected on development data using query batching and compiled reductions. The table reports the faster numerical-tolerance configuration; its maximum target-probability discrepancy is 4.95×10^{-6} and per-window NLL discrepancy is below 1.2×10^{-8}. Strict-tolerance results, timing ranges, and measurement boundaries are provided in Appendix E. The timings characterize these implementations on the study GPU.

Equal fan-out gives a different comparison. Restricting a continuous FIFO to eight or 32 recent *positions* gives PPL 129.51 or 126.24, versus 116.80 or 109.41 for exact $K = 8$ or 32. The exact store retains up to K occurrences *per key*, and therefore uses more total state than these short-window FIFOs. This comparison illustrates the value of per-key indexing under a small query budget; the matched-storage comparison above separately measures broader continuous retrieval.

5.2 Why broader search improves prediction

For the $K = 32$ comparison, 53.08% of the 524,288 distinct target positions have exact candidates but no correct successor. This group contributes $-.1973$ nats to the overall $-.1980$ continuous-minus-exact difference. In contrast, the 39.64% with positive exact target mass contribute $+.0177$ nats in favor of exact; the remaining 7.28% have no exact candidate (Figure 2). Continuous retrieval supplies target support on 81.32% of the candidate-present, target-absent positions. These groups explain the observed score difference; they do not evaluate an inference-time routing policy.

Eviction accounts for part of the coverage difference. On a separate eight-window bank, retaining *all* prior occurrences of the current key improves exact PPL from 64.59 at $K = 32$ to 62.46 and restores support on 14.68% of targets; continuous retrieval reaches 58.82. The all-history variant serves as an access-ceiling diagnostic: its growing history trades away the fixed-state property and can require quadratic total work for a repeatedly occurring key (Appendix E.3).

These findings motivate a complementary design: preserve an efficient exact lane for repeated transitions and use contextual retrieval to reach additional successors. The following studies measure the combined reader directly.

6 Exact access as a complement to contextual retrieval

Compact test-book study. A compact exact-plus-recent-state reader is compared with continuous-only retrieval under a declared 10.4 MB reader-array ceiling. Its exact component retains $K = 8$ successors. Nineteen eligible PG-19 test books each supply one 8,192-target window. These books were absent from the historical test manifests checked before the study; reader choices are frozen on development data, and every eligible report cell is included.

The hybrid improves on the payload-matched control on all 19 books at both scales (Table 3). Relative PPL reductions are approximately 1.02% at 40M and 0.36% at 350M. A secondary continuous control given exact-history summary features also has higher PPL, 55.760 and 28.037, although that control is not payload matched. This is evidence for the *combined reader* in the evaluated setting. The study reports logical array bytes, not realized device residency or latency (Appendix F).

Projected-reader study and branch intervention. A distinct hybrid pairs exact $K = 8$ with a 4,096-record contextual FIFO. Learned normalized keys have widths 128/256 at 40M/350M and are

Table 4: **Same-target 125M continuation.** Sixteen PG-19 documents and three checkpoints trained at 32K. All arms share weights and score identical targets across nested histories. The simple reader uses $K = 8$ and $\lambda = .025$. The contrast is simple-minus-reader-off NLL.

History	Reader-off PPL	Simple PPL	Learned PPL	Simple–off NLL
32K	45.341	44.497	44.542	$-.01879$
64K	45.339	44.528	44.728	$-.01805$
128K	45.338	44.525	44.697	$-.01808$

stored at four bits with per-record scales. A $21 \to 32 \to 3$ gate mixes backbone, exact, and contextual predictions. Global and token-specific trust offsets update after the corresponding target is observed; controls receive the same causal update. The 32 report books are held out from the run-specific training/development banks and come from the official PG-19 *training* split (Appendix F.2).

The estimated 40M contrast favors the hybrid; the 350M estimate is close to zero, with an interval spanning zero. Removing the exact branch's gate mass and renormalizing the remaining weights changes 40M PPL from 53.258 to 53.413. This intervention retains the original gates and measures the branch contribution under the fitted mixture. A separate CUDA-graph audit of this *projected* configuration measures reader times of 24.06 versus 25.19 microseconds per token at 40M and 25.28 versus 27.51 at 350M. Reader-state-plus-parameter bytes are 24.6% and 25.5% lower. These measurements cover reader state and computation; the backbone, prefill, temporary allocations, and graph pools are outside this timing scope.

The effect also depends on the hybrid design. An earlier consensus-gated configuration reaches 58.60 PPL versus 57.90 for continuous retrieval on its report bank; this comparison is documented in Appendix F.3. The three hybrid studies use distinct configurations and evaluation banks, so each comparison is interpreted within its own protocol.

7 Scope across context length and training regime

7.1 Same-target continuation through 128K

The 125M study evaluates the same final 256 targets in each of sixteen documents under nested 32K, 64K, and 128K histories. Simple exact improves 128K NLL by .01808 with a document-level 95% interval $[.00274, .03342]$ in gain orientation. Twelve of sixteen document means favor it. The gain remains similar across lengths; the context-interaction interval includes zero (Table 4). These results show stable operation beyond the training context over the tested range.

The retention rule explains this pattern. Prepending history leaves a bucket unchanged when its latest K records already lie in the shorter suffix. On these 4,096 distinct targets, 81.54% of inventories are identical at 32K and 128K. Longer history restores target support for 97 queries and loses it for none, although normalized vote mass can increase or decrease. These are post-report inventory diagnostics on the same sample (Appendix G).

At 128K, simple exact changes median decode time from 14.72 to 15.62 ms/token and prefill from 482.02 to 490.01 ms. Its additional populated persistent payload is 0.38 MiB, excluding Python containers and temporary allocation. All 54 specified 125M FP32 reference checks pass. A separate 40M same-target study finds 64K PPL 78.99 without a reader, 72.65 with simple exact, and 71.80 with the learned reader, with 144 passing terminal reference checks. Both studies use tolerance-based numerical parity criteria.

7.2 Longer-training quality and latency

Two 125M reader-free backbones receive 5B optimizer-token presentations sampled with replacement from a fixed 1B-token PG-19 pool. On the same validation bank, the exact-reader NLL advantage changes from .0311 at 0.1B presentations to .0025 at 5B. The protocol specified three seeds; results from the two completed runs are reported descriptively, without a seed-population interval.

Table 5: **Longer-trained 125M quality–latency grid.** Two reader-free backbones after 5B optimizer-token presentations from a fixed 1B-token pool. Eight reused test-document windows; teacher-forced 32K pipeline time. All nine configurations are retained in Table 32. State is not matched.

Reader	PPL	Pipeline ms
No reader	21.8058	85.78
Exact $K = 8$	21.7546	86.40
Exact $K = 32$	21.7328	86.37
Continuous $W = 32$	21.7965	86.41
Continuous $W = 512$	21.7144	93.18
Continuous $W = 1{,}761$	21.6776	111.13

At the test endpoint, exact $K = 32$ gives a 0.33% PPL reduction at approximately 0.7% additional measured pipeline time. The widest tested continuous reader reaches lower PPL at approximately 30% additional time (Table 5); it uses a development-selected fused sliding-window implementation. This grid places exact access at a low-incremental-cost point and wider continuous retrieval at the lower-PPL point. Results are measured on eight document windows and two completed backbones under the listed storage budgets (Appendix H).

7.3 Native-token prediction on a frozen pretrained model

On frozen Qwen3-1.7B, three small readouts are fitted without changing the backbone. Separate documents are used for fitting, calibration, and evaluation. The calibrated learned reader lowers PG-19 PPL from 19.689 to 19.583; simple exact reaches 19.592 and continuous retrieval reaches 19.420. On arXiv, the corresponding values are 4.932, 4.898, 4.904, and 4.883. These readouts yield small PPL changes relative to the frozen model, while continuous retrieval reaches the lowest PPL under its distinct access and storage budget (Appendix I). This experiment uses the model's native token probabilities. Deterministic prompt extraction and structured readouts are discussed separately from this neural probability-cache evaluation (Appendix L).

8 Related work

Successor voting and representation retrieval. Classical cache models adapt predictions to recent text (Kuhn & De Mori, 1990). Continuous neural caches weight observed successor tokens by hidden-state similarity (Grave et al., 2017), and pointer-sentinel models combine copying with parametric prediction (Merity et al., 2017). PCAF-Exact belongs to this family. Its distinguishing interface is a latest-K inventory indexed by exact token identity, with no record features needed by the fixed reader. This trades contextual matching for direct exact access. kNN-LM, Memorizing Transformers, RETRO, and product-key memory use representation neighborhoods, retrieved chunks, or trainable memory values (Khandelwal et al., 2020; Wu et al., 2022; Borgeaud et al., 2022; Lample et al., 2019). Their search procedures and indexing costs differ from a direct exact lookup, even when the number of returned neighbors is small.

Compressed models with additional memory. Zoology links associative recall to language-model quality and studies recall-targeted sparse attention (Arora et al., 2024a); Based, Gated DeltaNet, and Mamba-3 improve efficient sequence backbones (Arora et al., 2024b; Yang et al., 2025; Lahoti et al., 2026). Memory Caching retains recurrent-state checkpoints, while Neurocache, LoLA, and HOLA retain neural representations alongside efficient state (Behrouz et al., 2026; Safaya & Yuret, 2024; McDermott et al., 2026; Cui, 2026). Our study isolates the role of exact token-successor access when attached to a fixed backbone. The earlier PCAF preprint studies hash-bucket addressing and learned scoring (Ahmed, 2026); here we examine exact-key retention, simple readouts, and retrieval coverage.

9 DISCUSSION AND CONCLUSION

The results support a focused role for bounded exact-successor memory. Exact access preserves useful transition evidence with a small reader, and much of its measured benefit remains when learned scoring and gating are removed. The target-level analysis shows that broader contextual retrieval often reaches targets outside the exact inventory; the compact hybrid demonstrates that both access paths can improve a shared prediction at a fixed payload.

The controlled early-training comparisons show the clearest language-model gains, while longer-trained and frozen-pretrained settings show more modest changes. The compact study demonstrates quality gains for its fixed checkpoints; the projected reader provides a separate measurement of reader-path cost. Exact token identity is useful for repeated transitions, while contextual retrieval reaches a broader set of matches. Together, the experiments characterize exact access as an efficient complement whose contribution varies with the prediction regime and retrieval budget.

ETHICS STATEMENT

The experiments use public corpora, existing question–answer pairs, and synthetic retrieval tasks, with no new human-subject data collection. Exact recall can reproduce sensitive material already present in a prompt. Deployment should therefore enforce request isolation, access controls, retention limits, and output safeguards. A finite synthetic conflict test is not a guarantee of safe behavior on arbitrary natural text. Efficiency improvements may reduce per-request resource use while increasing aggregate demand.

REPRODUCIBILITY STATEMENT

Algorithm 1 and Appendix A specify the fixed exact reader and its causal retention rule. Appendix B, including its study registry, identifies architectures, controls, data partitions, and statistical units. The remaining appendices give seed-level results, calibration choices, numerical tolerances, and timing boundaries for the central studies. The manuscript source includes an evidence map linking revised tables to the corresponding experimental summaries. The compact and projected hybrid implementations have separate identities and accounting; one configuration's cost measurements must not be used for the other. Reproducing the training and GPU timing studies requires the corresponding original checkpoints, data manifests, and experiment-specific code in addition to the manuscript sources.

The compact 19-book hybrid's scoring code, frozen protocol, calibration artifacts, data manifest, and results are supplied in the accompanying source package under `experiment_code/iclr27_exact/`.

AI USE STATEMENT

AI tools assisted with coding and debugging, experiment and result checks, literature and citation checks, figure preparation, and manuscript editing and proofreading. The authors made the research and experimental decisions, verified the reported evidence, and are responsible for the final paper.

REFERENCES

Muhammad Ahmed. Parallel causal associative fields: Gated sparse memory for long-context language modeling. *arXiv preprint arXiv:2606.10435*, 2026. URL https://arxiv.org/abs/2606.10435.

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*, 2024a. URL https://arxiv.org/abs/2312.04927.

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 *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 1763–1840. PMLR, 2024b. URL https://proceedings.mlr.press/v235/arora24a.html.

Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Song, Han Zhao, Yuxiao Gao, Zhi Xu, et al. LongBench: A bilingual, multitask benchmark for long context understanding. In *Proceedings of the Association for Computational Linguistics*, 2024. URL https://arxiv.org/abs/2308.14508.

Ali Behrouz, Zeman Li, Yuan Deng, Peilin Zhong, Meisam Razaviyayn, and Vahab Mirrokni. Memory caching: RNNs with growing memory. *arXiv preprint arXiv:2602.24281*, 2026. URL https://arxiv.org/abs/2602.24281.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George van den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, et al. Improving

language models by retrieving from trillions of tokens. In *International Conference on Machine Learning*, 2022. URL https://arxiv.org/abs/2112.04426.

Wanyun Cui. A hippocampus for linear attention: An exact memory for what the recurrent state forgets. *arXiv preprint arXiv:2607.02303*, 2026. URL https://arxiv.org/abs/2607.02303.

Edouard Grave, Armand Joulin, and Nicolas Usunier. Improving neural language models with a continuous cache. *International Conference on Learning Representations*, 2017. URL https://arxiv.org/abs/1612.04426.

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*, 2024. URL https://openreview.net/forum?id=kIoBbc76Sy.

Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. Generalization through memorization: Nearest neighbor language models. In *International Conference on Learning Representations*, 2020. URL https://arxiv.org/abs/1911.00172.

Roland Kuhn and Renato De Mori. A cache-based natural language model for speech recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(6):570–583, 1990.

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

Guillaume Lample, Alexandre Sablayrolles, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. Large memory layers with product keys. In *Advances in Neural Information Processing Systems*, 2019. URL https://arxiv.org/abs/1907.05242.

Luke McDermott, Robert W. Jr. Heath, and Rahul Parhi. LoLA: Low-rank linear attention with sparse caching. *Transactions on Machine Learning Research*, 2026. URL https://openreview.net/forum?id=3KhDA252y3.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. In *International Conference on Learning Representations*, 2017. URL https://arxiv.org/abs/1609.07843.

Jack W. Rae, Anna Potapenko, Siddhant M. Jayakumar, and Timothy P. Lillicrap. Compressive transformers for long-range sequence modelling. In *International Conference on Learning Representations*, 2020. URL https://arxiv.org/abs/1911.05507.

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

Ali Safaya and Deniz Yuret. Neurocache: Efficient vector retrieval for long-range language modeling. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 870–883, 2024. doi: 10.18653/v1/2024.naacl-long.50. URL https://aclanthology.org/2024.naacl-long.50/.

Maurice Weber, Daniel Fu, Quentin Anthony, Yonatan Oren, Shane Adams, Anton Alexandrov, Xiaozhong Lyu, Huu Nguyen, Xiaozhe Yao, Virginia Adams, Ben Athiwaratkun, Rahul Chalamala, Kezhen Chen, Max Ryabinin, Tri Dao, Percy Liang, Christopher Ré, Irina Rish, and Ce Zhang. RedPajama: An open dataset for training large language models. *arXiv preprint arXiv:2411.12372*, 2024. URL https://arxiv.org/abs/2411.12372.

Yuhuai Wu, Markus N. Rabe, DeLesley Hutchins, and Christian Szegedy. Memorizing transformers. In *International Conference on Learning Representations*, 2022. URL https://arxiv.org/abs/2203.08913.

Songlin Yang, Jan Kautz, and Ali Hatamizadeh. Gated delta networks: Improving Mamba2 with delta rule. In *International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2412.06464.

A CAUSAL SEMANTICS, RETENTION, AND IMPLEMENTATION SCOPE

This appendix makes the fixed-reader contract explicit. The properties below follow directly from the storage and mixture rules; they are not claims of a new general theory of caching. Indices are one-based unless stated otherwise. A reset clears the applicable request or document history. Every query uses only completed records from that history.

A.1 CAUSALITY AND RECURRENCE-BASED EVICTION

Proposition 1 (Causal prediction)**.** *At input position t, every successor returned by Algorithm 1 is a function of $x_{1:t}$. Changing tokens strictly after t cannot change the exact inventory or fixed-reader prediction at t, provided the backbone is causal.*

Proof. Before querying x_t, the algorithm may append the newly completed record $(t-1, x_t)$ to the bucket for x_{t-1}. Every record previously present has an earlier source position. Thus each stored value is x_{i+1} with $i < t$, so $i + 1 \leq t$. Lookup uses x_t and previously committed keys; weights use only occurrence rank. The gate uses the nonempty indicator and a frozen scalar. The backbone distribution is causal by assumption. None of these operations accesses x_{t+1} or a later token. □

Proposition 2 (Latest-K retention)**.** *In a reset-free history with exact key identity preserved, a completed occurrence i remains in its key bucket at query t precisely when fewer than K later completed occurrences have that key.*

Proof. Insertion appends one record to its own key bucket. Only an insertion into that same bucket can increase the number of records newer than i. The algorithm removes the oldest record exactly when the capacity would exceed K. Hence i survives the first $K-1$ later same-key insertions and is removed by the Kth. Insertions under other keys leave the bucket unchanged. The query at t has no observed successor yet, so it does not count as another completed occurrence. □

For $K = 2$, consider $A, B, Z_1, \ldots, Z_m, A, C, A$, where none of the Z_j equals A and $B, C \neq A$. The final A retrieves both B and C regardless of m. Continuing with D, A replaces the old B with D. This is a retention guarantee, not a guarantee that any retained successor receives sufficient probability after weighting or gating.

Corollary 1 (Nested histories)**.** *For a fixed query and nested history suffixes, extending the suffix can add older records to an underfilled exact bucket but cannot remove a newer record already retained. If the latest K completed occurrences lie in the shorter suffix, its inventory is unchanged.*

The statement concerns inventory membership. Adding older records can change the normalization in Equation 2, including reducing the target vote. Target support can increase without a monotone improvement in perplexity. A lossy hash bucket, a reset inside the suffix, or a separate global key cap changes the assumptions. In particular, whole-key eviction under an additional global cap must be treated separately from per-key recurrence-based eviction.

A.2 PREDICTIVE GAIN AND MISSING SUPPORT

Dividing Equation 3 by $p > 0$ gives Equation 5. Since logarithm is strictly increasing, G_t is positive, zero, or negative according to $c - p$ for $g > 0$. With $g = 0$, it is zero. For $c = 0$ and $g < 1$, the penalty is $\log(1-g)$. The case $g = 1, c = 0$ has unbounded NLL before a numerical floor; the fixed reader therefore restricts $\lambda < 1$.

The mathematical identities concern ordinary probabilities away from implementation floors. Reported numerical checks are tolerance based. Matching-key coverage, target-valued-record recall, and actual probability gain are different quantities. The empirical analysis keeps their denominators separate rather than treating every repeated key as a successful recall.

A.3 STATE AND PARALLEL INDEX CONSTRUCTION

With fixed vocabulary V, batch size B, and capacity K, a dense fixed-reader implementation stores a successor ID and source position in each of BVK slots. A feature-bearing learned implementation additionally stores a d_k-dimensional key. With int64 IDs/positions and two-byte features,

$$M_{\text{fixed}} = 16BVK, \qquad M_{\text{learned}} = BVK(16 + 2d_k) \quad \text{bytes.} \tag{6}$$

One int64 count per bucket adds $8BV$ bytes. Pending records, backbone state, parameters, transient buffers, allocator overhead, and process residency are additional. For $d_k = 16$, the learned record payload is $48BVK$ bytes. These formulas do not apply to older routed PCAF prototypes that also retain router-state vectors.

Table 6: Calculated dense record capacity, MiB, at $V = 50{,}000$ and $d_k = 16$. Counts and pending state are excluded; these are layouts, not measured process memory.

Batch	Layout	$K = 2$	$K = 4$	$K = 8$	$K = 16$
1	Fixed token/rank	1.53	3.05	6.10	12.21
1	Learned feature	4.58	9.16	18.31	36.62
8	Fixed token/rank	12.21	24.41	48.83	97.66
8	Learned feature	36.62	73.24	146.48	292.97

Sparse allocation replaces dense vocabulary allocation with records for observed keys, plus a key dictionary and its metadata. The reported sparse stateful path counts retained tensor/int64 payload but excludes Python-container overhead. Occupancy can grow with a prompt while remaining below the fixed vocabulary-by-K capacity.

The parallel path encodes nonnegative token and position ranges in injective signed 64-bit keys, radix-sorts the length-BT index, and gathers fixed-K equal-key predecessors. The finite-width radix implementation uses a fixed number of passes, so sequence-dependent work and intermediate storage are linear in BT under its supported range. An arbitrary comparison-sort implementation would not inherit that statement. Counts and positions are assumed not to overflow between resets.

The original tests include future-token/value perturbations, adversarial duplicate keys, empty pools, boundary queries, ring wraparound, long rare-key gaps, and reset. A separate dense learned-feature store microbenchmark verifies fixed allocation at 1K and 8K streams, but excludes projections, scoring, gating, the backbone, and vocabulary head. It establishes a populated store, not optimized integrated serving.

A.4 THE OPTIONAL LEARNED READOUT

Query and record keys are bias-free projections of causal states, normalized to unit length, with $d_k = 16$ in the primary from-scratch models. For a candidate $i < t$, the score is

$$s_t(i) = \tau \langle \widehat{q}_t, \widehat{k}_i \rangle + \rho(i, t), \qquad \rho(i, t) = \lambda_{\text{rec}}\, i / \max(t, 1), \tag{7}$$

where this score uses the implementation's zero-based positions. The inverse temperature is $\tau = \exp(\text{clip}(a, \log .25, \log 64))$, initialized at eight. The recency coefficient is $\lambda_{\text{rec}} = \min(\text{softplus}(b), 2)$, initialized at .02. The query-relative position denominator does not depend on unused future sequence capacity.

Pruning sorts valid scores using detached values and always retains the best record. Additional candidates remain while the cumulative softmax mass before the candidate is below .90 and the score is within four of the best. Weights are renormalized over retained candidates. Selection is discrete, whereas retained scores remain differentiable.

The confidence vector contains maximum weight, entropy normalized by $\log \max(n, 2)$, the hyperbolic tangent of the top-two score gap, retained-candidate fraction, effective count divided by n, an agreement field, and route coverage. Effective count is $1/\sum_i w_i^2$. The exact-only agreement field is zero; empty pools have all-zero confidence. Separate linear projections of a LayerNorm of h_t and the confidence vector feed a width-$\max(d/2, 16)$ SiLU layer and scalar sigmoid. The gate is

initialized near .05 and multiplied by the nonempty indicator. The exact-only configuration has no learned product route or cross-batch memory.

For nonempty pools, the auxiliary gate target is positive when $\mathrm{sg}[c_t(y_t)] > \mathrm{sg}[p_{\theta,t}(y_t)]$ and negative otherwise. The loss equally weights the average positive and negative log losses, with denominators clamped to at least one and log arguments clipped at 10^{-6}. Its weight is .05. The labels are detached; the loss can nevertheless update shared hidden features through the gate network.

A.5 Responsibility completion: exact identity, empirical scope

For the target mixture, define $r = (1-g)p/q$. Ordinary mixture likelihood has

$$\frac{\partial(-\log q)}{\partial \log p} = -r. \tag{8}$$

The learned-reader recipe adds

$$L_{\mathrm{complete}} = -(1-\mathrm{sg}[r])\big(\log p - \mathrm{sg}[\log p]\big). \tag{9}$$

The term has zero forward value and derivative $-(1-r)$ along the direct $\log p$ input. Treating g and c as separate inputs, the combined coefficient is therefore -1. This deliberately specifies a surrogate gradient. It is not an equality of all backbone parameter gradients, because the scorer and gate can depend on shared features. Nor does it establish a generalization benefit. Appendix D preserves both objective cohorts and their differing mean orderings.

B Training configurations, data, and statistical units

B.1 Backbone and primary optimization

Table 7: Actual checkpoint inventory. Delta blocks replace convolution blocks within the total stack; positions are one-based. All tiers use a tied 20,000-token vocabulary.

Tier	Width	Blocks	Conv.	Delta	Delta positions	FFN conv./delta
40M	448	12	8	4	3, 6, 9, 12	1024 / 1016
125M	768	16	12	4	4, 8, 12, 16	1408 / 1400
350M	1024	24	20	4	6, 12, 18, 24	2360 / 2352

The convolution blocks use projected causal depthwise convolution, kernel size five, and dilation $2^{(\ell-1) \bmod 9}$ at layer ℓ. Blocks have RMS pre-normalization and SwiGLU feed-forward sublayers. Gated-Delta blocks have short-convolution size four, value expansion two, and no predecessor-write heads. Their head counts are 6/12/16 and key-head widths 56/48/48 across the three tiers. The reader is applied after the full backbone.

Table 8: Primary learned-reader optimization. Token budgets count optimizer presentations. All use BF16, tied embeddings, fused AdamW, cosine decay, gradient clipping at one, and exact cut-cross-entropy.

Setting	Params (M)	Context	Tokens (M)	Steps	Peak LR	WD
WikiText	40.118	8,192	100.008	12,208	.001	.10
arXiv / code	40.118	8,192	100.008	12,208	.001	.10
PG-19	40.118	65,536	100.008	1,526	.0006	.01
PG-19	124.405	32,768	100.008	3,052	.0006	.01
PG-19	346.435	8,192	200.008	24,415	.0004	.10

At 40M, learned exact and matched no-reader contain 40,118,068 and 40,117,842 parameters. At 125M the counts are 124,405,380 and 124,406,530. The 40M and 350M readers contribute 117,603 and 563,715 parameters; matched adapters contribute 117,377 and 563,201. The matched adapter is position-wise and cannot retrieve across positions. A dense-linear MAC proxy excludes normalization,

sorting, indexing, softmax, and scatter; measured runtime is the resource comparison rather than the proxy alone.

Primary model seeds are 20260921/22/23 for WikiText, 20260937/41/42 for 40M PG-19, 20263121/22/23 for 125M PG-19, and 20260936/43/44 for 350M PG-19. Train/evaluation sampling seeds are 31001/41001. WikiText scores twelve fixed 8K batches; PG-19 scores eight 64K, eight 32K, or twelve 8K batches at the corresponding tiers. PG-19 windows respect document boundaries; standard contiguous WikiText windows do not.

B.2 Corpus identity and split scope

The recorded dataset revision prefixes are `b08601e04326c79d` for WikiText-103 and `c021754c8e01c5b1` for PG-19. The RedPajama revision is recorded in full in the original release manifest (prefix `398f92572e94`, suffix `9e7de4`). A seeded content-hash rule assigns unique RedPajama documents to a 95/5 train/evaluation split. Deduplication and empty-document filtering leave 12,471/692 arXiv documents and 318,122/16,540 code documents. Both domains use model seeds 20263851/52/53 and 32 fixed evaluation batches.

The separate fixed-reader factorial uses contiguous validation blocks and can cross PG-19 document boundaries. It therefore cannot replace the document-contained primary evaluation. Its WikiText bank contains four calibration and twenty report blocks; its PG-19 bank contains two calibration and eight report blocks. The PG-19 report includes three boundary-containing blocks, with no within-block reset. These are comparisons on previously evaluated validation corpora, not fresh global confirmation.

B.3 Statistics and what the intervals cover

For primary from-scratch comparisons, losses are first averaged within each training seed. A paired contrast is formed within seed, and the descriptive two-sided Student-t interval uses two degrees of freedom. Geometric-mean PPL exponentiates the mean of the seed NLLs. The three-seed intervals describe this small replicated sample; correlated tokens do not create additional training replicates.

For the same-target and hybrid studies, let $\ell_{d,s,m}$ denote mean NLL for document d, fixed checkpoint s, and method m. The document contrast is

$$\delta_d = \frac{1}{3}\sum_{s=1}^{3}\left(\ell_{d,s,\text{candidate}} - \ell_{d,s,\text{reference}}\right). \tag{10}$$

The document bootstrap resamples d while carrying the methods and checkpoints for each sampled document together. Its interval describes document variation conditional on those checkpoints, not variability over newly trained models. The projected-hybrid study uses 97.5% intervals for two scale-wise comparisons; the compact study uses descriptive 95% intervals. Their confidence levels and banks are not pooled.

The longer-training study has only two completed seeds after a planned three-seed design and reports results descriptively. Repeated evaluation at its four milestones uses the same validation documents. An interval containing zero is interpreted as unresolved ordering, not equivalence. Timing ranges across seed medians are not confidence intervals, and single-device results do not establish a hardware-independent complexity advantage.

B.4 Study registry

C Complete primary contrasts and independent-backbone controls

For arXiv, exact-minus-matched NLL is $-.06203$ $[-.10969, -.01438]$. For code it is $-.01872$ $[-.09490, +.05746]$: two pairs favor exact and one favors matched. Same-checkpoint gains are $+.06863$ $[+.05308, +.08418]$ and $+.06309$ $[+.03890, +.08729]$. Cross-domain averages are not a primary endpoint.

Table 9: Evidence families and their comparison units. Each row retains its own bank, checkpoints, and measurement boundary.

Study	What is held fixed / replicated	Supported measurement
Primary attribution	Three training seeds per setting; matched backbone family, data, schedule	Learned-reader quality versus independent capacity-matched controls
Independent fixed cache	Three reader-free replicas; eight document-contained 64K windows	Attachment without reader co-training
Co-trained fixed rank	Three new matched seed pairs; learned gate retained	Scoring/pruning replacement, not removal of all learning
Storage-matched cache audit	Same three reader-free backbones and eight report windows	Reader and teacher-forced pipeline time; explicit record bytes
All-history exact diagnostic	Three fixed backbones; eight separate 64K windows	Effect of removing per-key eviction; not fixed-state streaming
Compact hybrid	Six fixed checkpoints; 19 PG-19 test books	Payload-matched quality only
Projected hybrid	Six fixed checkpoints; 32 bank-held-out PG-19 train-tail books	Quality; separate incremental reader-only cost
125M continuation	Three fixed checkpoints; 16 test documents; nested histories	Same-target quality and stateful execution through 128K
Longer training	Two completed backbones; fixed 1B-token pool and reused report documents	Descriptive 5B-presentation trajectory and pipeline grid
Native-token pretrained	One frozen Qwen3-1.7B; three readout fits; 24 report documents/domain	Token-probability quality; no serving-speed result
Prompt selector	Frozen pretrained models; separate deterministic rules and task banks	Explicit-format extraction and abstention, not neural-cache quality

Table 10: Per-seed primary PPL. Parentheses give the learned checkpoint's reader-off PPL. A dash means the plain no-reader arm is not reported for that domain; the matched control remains available.

Setting	Seed	Learned exact (off)	No reader	Matched
Wiki / 8K	20260921	22.05 (22.70)	22.51	22.43
	20260922	21.95 (22.68)	22.54	22.41
	20260923	21.66 (22.26)	22.61	22.59
arXiv / 8K	20263851	6.226 (6.715)	–	6.500
	20263852	6.011 (6.404)	–	6.521
	20263853	6.057 (6.476)	–	6.442
Code / 8K	20263851	2.983 (3.206)	–	3.121
	20263852	2.871 (3.026)	–	2.947
	20263853	2.965 (3.164)	–	2.921
PG / 40M, 64K	20260937	115.18 (129.44)	127.23	127.59
	20260941	114.33 (129.57)	131.59	129.66
	20260942	114.11 (129.92)	131.67	131.93
PG / 125M, 32K	20263121	55.15 (57.44)	–	57.26
	20263122	55.34 (57.56)	–	57.28
	20263123	55.39 (57.75)	–	57.59
PG / 350M, 8K	20260936	35.02 (36.70)	37.13	37.11
	20260943	34.96 (37.09)	37.97	37.02
	20260944	35.23 (36.83)	37.55	37.21

C.1 Independent reader-free replicas with a parameter-free cache

The three 40M PG-19/64K matched controls are independently reconstructed using the same backbone initialization consumption, data, token budget, and schedule: 1,526 steps, 100,007,936 optimizer tokens, batch one, learning rate .0006, 76 warmup steps, cosine decay to one-tenth peak, and weight decay .01. These reader-free models use ordinary cross-entropy. A fixed reciprocal-rank cache and coefficient .025 introduce no new trainable parameters or auxiliary loss.

The report uses the original document-contained bank with start offsets 572371, 375554, 441048, 798948, 165763, 491022, 617460, and 476313, disjoint from constant-gate development spans. No report outcome selects a policy, seed, or checkpoint. Learned checkpoints are re-evaluated, not retrained. Replica terminal NLL differs from the original matched terminal values by $-.00363, +.00611, -.00801$, within the stated .01 tolerance; these are not bit-identical replicas.

Table 11: Independent-backbone cache control on the document-contained PG-19/64K bank. The final row exponentiates mean seed NLL. The learned system was trained separately.

Seed	Reader off	Fixed cache	Learned exact	Fixed–learned NLL
20260937	127.08	115.40	115.13	+.00237
20260941	130.46	117.33	114.40	+.02523
20260942	130.91	117.50	114.13	+.02908
Geometric mean	129.47	116.74	114.55	+.01889

The descriptive paired interval is $[-.01697, +.05476]$. All three point differences favor the learned reader, but the ordering is not resolved by the small paired sample. The result establishes a strong parameter-free attachment rather than an equivalence claim.

C.2 Continuous cache on the same independently trained features

The continuous reader pairs a prior hidden state with its observed successor and performs strictly causal retrieval from a recent FIFO. The development grid covers raw-dot temperature settings $\{0, .001, .003, .01, .03, .1, .3, 1\}$, with zero denoting the hard-neighbor limit, and cosine settings $\{1, 4, 16, 64\}$. Both readers calibrate their mixture weight on $\{0, .01, .025, .05, .1, .2, .35, .5, .75\}$. The continuous reader therefore receives its own score calibration. All three continuous selections choose cosine inverse temperature 16 and weight .35.

Two document-contained development windows per seed are disjoint from each other and the eight report windows. The common 7,680,000-byte record ceiling admits 8,495 BF16 vectors of width 448 plus int64 successors. The fixed token-only exact cache uses less than this ceiling. This is a common ceiling, not equality of actual payload. No backbone is updated by either attached cache.

Table 12: Same reader-free backbone comparison. Seed headings abbreviate 20260937/41/42. The co-trained row has different backbone weights.

Reader	NLL 0937	NLL 0941	NLL 0942	PPL
None	4.84485	4.87105	4.87454	129.47
Exact, locked .025	4.74844	4.76495	4.76646	116.74
Exact, calibrated	4.75164	4.76452	4.76534	116.80
Continuous, calibrated	4.50881	4.51918	4.51162	91.21
Co-trained learned PCAF	Separate backbone/readout training			114.55

Continuous-minus-calibrated-exact NLL is $-.24730$ $[-.26147, -.23312]$. This quality advantage is obtained by reading up to 8,495 records rather than eight. The storage-matched and equal-fan-out studies below use different capacity constraints to expose that distinction.

D Readout and training-component ablations

D.1 Fixed-feature factorial

The factorial preserves the three WikiText/8K and three PG-19/64K exact-trained checkpoints and changes only the readout. Its contiguous validation-bank protocol is distinct from the document-contained primary bank (Appendix B). Constant gates minimize calibration NLL over $\{0, .01, .025, .05, .10, .20, .35, .50, .75\}$, with ties favoring smaller values. All six checkpoints select reciprocal-rank weighting with gate .025 among the simple policies. Latest-successor selects .01 and uniform selects .025.

Table 13: Fixed-feature factorial. Δ is condition-minus-original NLL; positive favors the original learned/pruned readout. Each corpus has three checkpoints. Full intervals accompany the original factorial; this compact table retains all point estimates.

Weights	Gate	Wiki PPL	Wiki Δ	PG PPL	PG Δ
Reader off	Zero	24.84	.03563	109.23	.12244
Learned / pruned	Original	23.97	.00000	96.65	.00000
Learned / pruned	Constant	24.22	.01030	97.34	.00717
Learned / all eight	Original	23.92	$-.00212$	95.58	$-.01108$
Learned / all eight	Constant	24.19	.00931	96.59	$-.00053$
Latest	Original	30.52	.24167	108.38	.11463
Latest	Constant	24.62	.02696	105.26	.08539
Uniform	Original	24.81	.03453	96.99	.00360
Uniform	Constant	24.37	.01682	99.19	.02603
Reciprocal rank	Original	24.89	.03786	97.10	.00470
Reciprocal rank	Constant	24.38	.01695	99.17	.02582

On fixed learned features, replacing learned weights with a simple policy can worsen quality, especially when retaining the original gate. This does not contradict successful independent attachment: the feature training and gate interventions differ. Removing pruning improves NLL in all six checkpoint pairs. Simple weighting, contextual gating, and pruning should therefore not be treated as one inseparable mechanism.

D.2 Co-trained fixed rank versus learned ranking and pruning

Three new fixed-rank models use the same inventory, successor values, learned confidence gate, completion objective, budget, initialization consumption, and model seeds as learned references. A position-wise adapter replaces learned score projections, giving 40,118,068 parameters per model. Every valid record participates with score $-\log r$. Two unused development windows and eight nonoverlapping, document-contained 64K report windows are kept separate from selection.

Table 14: Co-trained weighting comparison. All terminal checkpoints are included. The contrast is fixed-minus-learned NLL.

Seed	Learned NLL	Fixed NLL	Δ
20266411	4.358437	4.357068	$-.001368$
20266412	4.357564	4.364967	$+.007403$
20266413	4.369261	4.357384	$-.011877$
Mean contrast	$-.001948$ $[-.025927, +.022032]$		

Within the fixed-rank checkpoints, the mixture improves the parametric path by $.0867, .0881, .0862$ nats. The comparison identifies the incremental weighting/pruning package under a retained learned gate; it is not the fully parameter-free attachment of Table 11.

D.3 Two objective cohorts and a 24-model component cohort

The first objective cohort contains six new 40M PG-19/64K models, paired within seed on architecture, samples, schedule, and initialization. Both conditions use the same gate auxiliary. Its completed-minus-ordinary distinction changes only the responsibility correction.

Its mixture-NLL interval is $[-.00728, +.02228]$; the parametric-only contrast is $+.01705$ $[-.00450, +.03859]$. The coefficients verify the derivative identity, while the quality intervals include zero.

The later cohort retains 24 checkpoints: eight conditions at seeds 20266411–20266413, all using the 40M/64K recipe, 1,526 steps, 100,007,936 tokens, train-sampling seed 32001, and eight fixed report windows from seed 42001. Initial parameter fingerprints match across conditions within seed. The

Table 15: First completion cohort. Δ is ordinary-minus-completed NLL. Multiplier is the mean direct target-parametric coefficient magnitude.

Seed	Complete PPL	Ordinary PPL	Δ	Complete mult.	Ordinary mult.
20263801	86.32	87.49	$+.01346$	1.000	.896
20263802	87.39	87.53	$+.00156$	1.000	.896
20263803	87.62	88.28	$+.00749$	1.000	.895
Geometric mean	87.11	87.76	$+.00750$	1.000	.895

reference uses $K = 8$ and completion; other arms change one component. No report checkpoint or condition is selected after outcomes.

Table 16: Training-time components. Seed columns abbreviate 20266411–413. Δ is variant-minus-reference NLL, with descriptive three-seed 95% intervals.

Condition	411 PPL	412 PPL	413 PPL	Mean PPL	Δ [interval]
Reference $K = 8$	87.37	87.06	88.17	87.53	Reference
Ordinary objective	86.63	87.31	88.22	87.38	$-.00172$ $[-.01663, +.01319]$
$K = 2$	91.33	92.05	91.84	91.74	$+.04694$ $[+.02750, +.06638]$
$K = 4$	89.46	90.49	89.57	89.84	$+.02600$ $[-.00279, +.05479]$
$K = 16$	85.76	84.68	85.61	85.35	$-.02526$ $[-.03992, -.01061]$
No pruning	86.26	86.62	86.65	86.51	$-.01174$ $[-.02713, +.00365]$
Zero recency	87.25	87.78	88.20	87.74	$+.00238$ $[-.01035, +.01512]$
Fixed temperature	87.02	87.35	88.16	87.51	$-.00027$ $[-.00941, +.00887]$

The ordinary objective's mean ordering reverses between cohorts, and both aggregate intervals include zero. Common-reference token slices also show heterogeneous effects: completion improves the parametric predictor in middle nonzero-cache-evidence bins, while target-absent positions move in the opposite direction. Those slices are fixed from a separate reference checkpoint, not recomputed using each arm's outcomes. The restored coefficient must therefore be separated from a claim of reliable aggregate quality improvement.

E MATCHED-STORAGE COST AND COVERAGE

E.1 THE ACTUAL-OPERATOR OPTIMIZATION AUDIT

The audit fixes three independently trained, reader-free 40M backbones, hidden width 448, vocabulary size 20,000, and eight document-contained PG-19/64K report windows per seed. It is an engineering sensitivity on previously exposed data, not a new quality confirmation. The exact operator includes causal record resolution, latest-K selection, reciprocal-rank voting, successor aggregation, and final mixture NLL. The continuous operator includes BF16-to-FP32 normalization, FP32 cosine scores and softmax, strict FIFO causality, successor aggregation, and mixture NLL. TF32 is disabled. The frozen gates are .20 for exact and .35 for continuous, with continuous inverse temperature 16.

An exact compact layout uses two int64 $V \times K$ arrays and one int64 counter per key, or $8V(2K+1)$ bytes. Continuous records use 448 BF16 coordinates and an int64 successor, or $904W$ bytes. Thus $K = 8$ pairs with $W = 3{,}008$ at 2,720,000 versus 2,719,232 bytes, and $K = 32$ pairs with $W = 11{,}504$ at 10,400,000 versus 10,399,616 bytes. These arrays account for a declared record payload, not an integrated streaming allocator or the full evaluation working set.

Development timing compares query chunks of 256, 512, 1,024, and 2,048 with eager or compiled mask/softmax/reduction. Two pre-existing, report-disjoint development windows of seed 20260937 supply three warmups and five repetitions per candidate. The selected implementation is frozen before report execution. The strict audit requires maximum target-probability and mixture-NLL errors at most 10^{-6} and selects compiled reduction with chunks of 256. A separately declared, post-primary engineering sensitivity relaxes only target-probability tolerance to 10^{-5} and selects chunks of 2,048. The main text compares against this faster secondary implementation rather than making the continuous baseline obey an unnecessarily restrictive GEMM shape.

Table 17: Secondary numerical-tolerance audit. Times average three seed-level medians. Peak is reader-path allocated CUDA memory, including resident features and temporaries; all complete-pipeline peaks are 1,257.56 MiB.

Reader	Payload MiB	PPL	Read ms	Pipeline ms	Peak MiB
Exact $K = 8$	2.5940	116.80	3.83	75.56	295.16
Continuous 3,008, original	2.5933	96.15	84.99	157.68	468.31
Continuous 3,008, selected	2.5933	96.15	8.80	82.70	475.37
Exact $K = 32$	9.9182	109.41	9.98	78.69	878.56
Continuous 11,504, original	9.9178	89.76	96.53	159.20	468.31
Continuous 11,504, selected	9.9178	89.76	22.64	96.70	674.81

The strict audit has maximum probability/NLL errors $4.18 \times 10^{-7}/1.84 \times 10^{-9}$; the secondary sensitivity has $4.95 \times 10^{-6}/1.15 \times 10^{-8}$. All 24 seed–window quality values per condition reproduce previously saved NLL within 4.17×10^{-8}. Under the strict protocol, exact-8/continuous-3,008 reader times are 3.30/50.56 ms and pipeline times 75.29/126.77 ms; the corresponding exact-32/continuous-11,504 times are 8.59/58.00 and 78.56/121.01 ms. These larger gaps do not replace the faster-comparator results.

On one report window per seed, each condition receives three rotating-order warmups, ten synchronized reader repetitions, and five synchronized complete-pipeline repetitions. Pipeline timing includes backbone encoding and exact cut-cross-entropy target probabilities. Loading and host transfer are excluded; cached features are released before pipeline timing. Peak allocation is measured separately. The protocol evaluates teacher-forced windows, not autoregressive generation.

Both fixed-K and fixed-W readers have bounded per-query work. These measurements establish neither a hardware lower bound nor an asymptotically widening latency advantage. They show an implementation-specific exchange: exact access has lower measured read cost, whereas continuous retrieval has lower perplexity.

Table 18: Ranges across the three seed-median times in the secondary audit. These are ranges, not confidence intervals.

Reader	Reader range (ms)	Pipeline range (ms)
Exact $K = 8$	2.97–5.34	75.31–75.74
Selected continuous $W = 3{,}008$	8.59–9.13	82.50–82.82
Exact $K = 32$	8.10–13.50	78.41–78.91
Selected continuous $W = 11{,}504$	22.46–22.76	96.58–96.85

E.2 Equal fan-out is a different comparison

At $K = W \in \{8, 32\}$, both policies consult at most the same number of records per query. They do not have equal storage or temporal coverage. Exact reserves up to K records per vocabulary key, while the continuous comparator retains only the most recent W positions globally. Each policy selects its gate on separate development windows; continuous also selects its score and temperature. Every seed favors exact at both fan-outs. This identifies the utility of indexing under a very small read budget, not superiority over a representation index with the same retained history. The small-window continuous prototype is launch dominated; its timing is not used for the main efficiency claim.

Table 19: Equal-query-fan-out quality comparison on the same 40M/64K backbones. The NLL contrast is continuous minus exact, with descriptive three-seed 95% intervals.

Fan-out	Exact PPL	Cont. PPL	NLL contrast	Exact MiB	Cont. MiB
≤ 8	116.80	129.51	$+.1033$ $[+.0537, +.1529]$	2.594	.007
≤ 32	109.41	126.24	$+.1431$ $[+.0425, +.2437]$	9.918	.028

E.3 Where the storage-matched quality gap occurs

The $K = 32$ diagnostic partitions the same 524,288 distinct target positions by the exact inventory's target support. The three backbones share these targets, so the corpus fractions are counted once, not as three independent token samples. Continuous retrieval supplies positive target mass to 226,304 of the 278,288 positions with exact candidates but zero exact target mass (81.32%). The contributions sum to the overall gap up to displayed rounding and have consistent signs across seeds. Gates differ between the policies, so this is not a causal intervention isolating access alone. Its implication is narrower: reweighting a fixed exact inventory cannot assign positive mass to an absent successor.

Table 20: Target-defined decomposition of continuous-minus-exact NLL at matched record payload. Negative contribution favors continuous. Groups are explanatory and cannot be used as an inference-time selector.

Exact evidence	Targets (%)	Exact NLL	Cont. NLL	Contribution
No candidate	7.28	4.162	3.909	$-.0184$
Candidates, zero target mass	53.08	6.443	6.072	$-.1973$
Candidates, positive target mass	39.64	2.452	2.497	$+.0177$
All	100.00	4.695	4.497	$-.1980$

E.4 Removing the per-key cap does not remove the exact-key restriction

A separate bank contains eight additional, document-contained 64K windows on the same three reader-free backbones. Exact $K = 8$, $K = 32$, and all-prior-same-key access use the same reciprocal-rank vote and fixed gate .2. The continuous reader retains $W = 11{,}504$, cosine temperature 16, and gate .35. The all-history arm changes retention, not the key definition. All-history exact improves over $K = 32$ in all seeds, with mean NLL difference $-.03351$ $[-.03378, -.03325]$. Continuous-minus-all-exact remains $-.05997$ $[-.06109, -.05885]$. Uncapped retention restores target support

Table 21: All-history exact-access ablation on a separate window bank. These PPLs should not be compared numerically with the preceding bank.

Reader	PPL
Exact $K = 8$	67.13
Exact $K = 32$	64.59
Exact, all prior same-key records	62.46
Continuous $W = 11{,}504$	58.82

on 76,982 positions (14.68%) where $K = 32$ has candidates but no target mass. A separately development-selected all-history gate gives PPL 63.80; the fixed-gate arm remains the prespecified primary comparison. The GPU all-history target probabilities agree with a direct CPU implementation within 2.98×10^{-7}.

The most frequent key has 3,688–6,156 eligible prior occurrences in these windows. Unlike bounded streaming exact memory, this diagnostic stores $O(T)$ events and can perform $O(T^2)$ total read work if a key repeatedly occurs. It tests a coverage ceiling, not a proposed constant-state implementation.

E.5 Value interventions and positive versus negative reads

Frozen-checkpoint interventions hold the read weights and gate fixed when reassigning successor values. On WikiText/8K, successor permutation increases NLL by .00576 [.00501, .00650]; on PG-19/64K, the increase is .01329 [.01320, .01337]. Candidate-order reversal changes NLL by exactly zero because scoring and aggregation are permutation invariant under the specified sorting/pruning rule. Reversing recency has smaller increases, .00028 on WikiText and .00004 on PG-19.

Table 22: Descriptive decomposition of the learned reader's gain. Fractions use all targets; conditional gain is $\log(q/p)$. The remaining positions have empty inventories and zero gate.

Corpus	Helpful (%)	Gain if helpful	Other covered (%)	Gain if other	Net
WikiText-103	19.42	+.392	59.41	−.079	+.029
PG-19	18.98	+1.112	73.74	−.118	+.124

A majority of covered predictions can be individually worsened while the mean gain remains positive. In the PG-19 conflict diagnostic, single-successor inventories have larger average gain than inventories with multiple successor values (.213 versus .129 nats). These overlapping, outcome-defined summaries motivate calibration; they do not establish a target-free oracle gate.

F THREE DISTINCT EXACT-PLUS-CONTEXT EXPERIMENTS

F.1 COMPACT READER ON 19 PG-19 TEST BOOKS

The compact study evaluates all 19 eligible books in a frozen PG-19 test inventory that were absent from the historical test manifests checked before the study. This does not establish that those books were globally unseen. One SHA-256-derived 8,193-token window per book supplies 8,192 predictions. There are three fixed checkpoints at each of 40M and 350M. Reader weights and temperatures are frozen on development data before report scoring, with no report-cell exclusion or retuning.

The compact configuration is identified as `pcaf_compact_hybrid_confirmation_v1`; its scoring implementation and frozen artifacts are included in the source package under `experiment_code/`. For each query, the exact branch reads the latest $K = 8$ observed successors of the current token ID. Its ring is indexed directly by the 20,000 token IDs and stores a $20{,}000 \times 8$ `uint16` successor array, 20,000 `uint8` head indices, 20,000 `uint8` counts, and one `int32` pending token. The ring therefore uses 360,004 array bytes. On observing a new token, the implementation completes the preceding token's successor record, advances that key's ring, and then reads the current key's successors newest first. If $c_t(y)$ is the normalized reciprocal-rank vote for successor y, the exact branch assigns weight proportional to $1/j$ to the jth most recent retained occurrence.

The contextual branch uses the backbone's existing hidden state directly: there is no learned contextual projection or projection-fitting stage in this compact experiment. A record stores the hidden vector h_i as BF16 and the observed successor x_{i+1} as int64, for $2d_{\text{model}} + 8$ bytes per record. At query position t, the reader considers only prior positions $i \in [\max(0, t - W), t - 1]$. It casts the stored BF16 vectors to FP32, L2-normalizes them, computes cosine similarities with inverse temperature 16, applies a softmax over the eligible prior positions, and sums the weights whose stored successor equals the target. The contextual widths are set separately for each frozen checkpoint from the declared 10,399,616-byte record-array budget B: $W_{\text{hybrid}} = \lfloor (B - 360{,}004)/(2d_{\text{model}} + 8) \rfloor$ and $W_{\text{continuous}} = \lfloor B/(2d_{\text{model}} + 8) \rfloor$. The model widths are 448 at 40M and 1,024 at 350M.

The hybrid gate is a five-feature linear softmax with 15 FP64 parameters. Its prefix-only feature vector is

$$f_t = \left[1, \frac{n_t}{8}, \max_y c_t(y), \frac{-\sum_y c_t(y) \log c_t(y)}{\log 8}, \frac{|\{y : c_t(y) > 0\}|}{8}\right],$$

where n_t is the number of available exact records; the four non-bias features are zero when the exact bucket is empty. With fitted matrix $A \in \mathbb{R}^{5\times 3}$, $\pi_t = \text{softmax}(f_t A)$ mixes the backbone, exact, and contextual target probabilities:

$$q_t(y) = \pi_{t,0} p_t(y) + \pi_{t,1} e_t(y) + \pi_{t,2} r_t(y).$$

When the exact bucket is empty, its branch distribution is set to the contextual distribution. When no prior contextual record exists, both reader branches equal the backbone distribution. The gate sees no current-target value; target probabilities enter only the calibration loss.

For each of the six frozen checkpoints, calibration uses the first two declared calibration documents, with 8,192 targets per document. The contextual inverse-temperature grid is $\{4, 8, 16, 32\}$; all six hybrid fits select 16. Gate weights are fitted jointly on the two calibration documents by L-BFGS (maximum 120 iterations, learning rate 0.7, gradient tolerance 10^{-9}, strong-Wolfe line search), minimizing mean calibration NLL plus 0.001 times the mean squared non-bias coefficients. Slopes start at zero and the bias logits start at $[0, -5, -3]$. The payload-matched continuous control uses inverse temperature 16 and a fixed gate of 0.10 at 40M and 0.05 at 350M. The fitted weights and selections are frozen before report scoring. The exact ring and contextual records are reset at each document boundary.

The declared budget below counts record arrays, not gate parameters. Gate parameters add 120 bytes for the hybrid and 8 bytes for the fixed-gate continuous control. The matched comparison accounts for capacities and record-array payloads as follows. The record-array payloads differ by less than 0.007%. Including gates, the hybrid/continuous totals are 10,399,044/10,399,624 bytes at 40M and 10,399,572/10,399,256 bytes at 350M; thus the 40M continuous total is 8 bytes above the record-array budget. A second continuous comparator receives exact-history confidence summaries and is not

Table 23: Compact-reader capacities and record-array payloads. Gate parameters are itemized separately in the text. Array payloads exclude Python objects, temporary allocations, model weights, and process residency.

Scale	W_{hybrid}	$W_{\text{cont.}}$	Hybrid (B)	Cont. (B)	Diff. (B)
40M	11,105	11,504	10,398,924	10,399,616	692
350M	4,883	5,058	10,399,452	10,399,248	204

payload matched. This second comparator tests a broader confidence interface, not equal realized memory. The compact study does not include latency or device-residency measurements.

For each document and scale, seed-level NLL contrasts are averaged across the three fixed checkpoints. Ten thousand paired document-bootstrap draws produce the descriptive intervals. These intervals condition on the existing checkpoints and do not quantify variation from retraining or across the complete PG-19 test corpus. All 19 document means favor the hybrid against the primary payload-

Table 24: Complete compact-hybrid contrasts on the 19-book sample. Δ is hybrid minus comparator NLL.

Scale	Comparator	Hybrid PPL	Control PPL	Δ [95% interval]
40M	Payload matched	55.581	56.152	$-.01022$ $[-.01426, -.00703]$
	Wide confidence	55.581	55.760	$-.00322$ $[-.00421, -.00231]$
350M	Payload matched	28.017	28.118	$-.00362$ $[-.00449, -.00281]$
	Wide confidence	28.017	28.037	$-.00074$ $[-.00103, -.00044]$

matched comparator at both scales. Against wide confidence, 18/19 documents favor it at 40M and 16/19 at 350M. Reader-free PPL is 65.368 and 29.381, respectively. The result supports the combined compact reader at these operating points; it does not transfer the timing of another hybrid to this configuration.

F.2 Projected contextual reader with a learned gate

This separate configuration retains latest-$K = 8$ exact successors with reciprocal-rank weights and the most recent $W = 4{,}096$ contextual records. A trained projection produces unit-normalized keys of width $d_c = 128$ at 40M and $d_c = 256$ at 350M. Retrieval uses signed four-bit keys with a per-record FP32 scale. Cosine scores have inverse temperature 16 and are normalized over strictly prior records. Each branch returns a successor distribution; an empty branch falls back to the backbone distribution.

A $21 \to 32 \to 3$ MLP produces mixture logits from five exact-prefix summaries and sixteen projected-state coordinates. The continuous controls use the same feature interface with two output logits. The mixture combines backbone, exact, and contextual probabilities. After prediction t has been scored and y_t observed, define

$$r_{t,j} = \frac{g_{t,j} p_{t,j}(y_t)}{\sum_k g_{t,k} p_{t,k}(y_t)}, \qquad u_t = r_t - g_t. \tag{11}$$

The global offset and the offset for the current token key update as

$$b_{t+1} = .999 b_t + .05 u_t, \qquad b_{x_t,t+1} = .999 b_{x_t,t} + .05 u_t. \tag{12}$$

The next query adds its applicable global and key-specific offsets to frozen gate logits before softmax. Offsets reset at document boundaries. The current target is not used to choose its own prediction, and the continuous controls use the same post-observation update.

The projection is fit on 32 distinct development books for 512 updates with objective .8 times backbone NLL plus .2 times contextual NLL. Gate fitting uses the same development partition, 300 Adam updates at learning rate .003, and .001 displacement regularization. The width rule is $d_c = 64\lceil d_{\text{model}}/256 \rceil$. Projection, gate, and update settings are fixed before report documents are prepared. This specification refers to the projected configuration, not the 15-parameter compact gate above.

Report bank and estimand. The first 32 eligible PG-19 train-tail documents from raw index 28,000 onward each supply a deterministically offset 32,769-token window. Two short documents are excluded using a pre-scoring length rule. Raw identities and sliding 64-token blocks are checked against declared training, development, calibration, and prior confirmation banks. The bank is held out from those sources but remains in the official training split.

The primary endpoint averages the first 8,192 target NLLs in each window. The primary continuous comparator uses full-width contextual records and the same global/key-specific trust updates. All six fixed checkpoints are evaluated on all windows. After averaging three checkpoint contrasts per document, 20,000 paired document-bootstrap draws with seed 20260925 yield 97.5% intervals, giving a two-scale Bonferroni family-wise level of 95%. The intervals are conditional on fixed checkpoints. The checkpoint-mean contrasts are $-.00484, -.00390, -.00357$ at 40M and $-.00095, -.00069, -.00051$ at 350M. The shared sign does not turn a document-conditional interval into a training-seed interval. Over all 32,768 predictions, hybrid/control PPL is 50.384/50.718 at 40M and 27.562/27.608 at 350M; these longer-window endpoints are descriptive.

Table 25: Projected-reader primary endpoint. Δ is hybrid minus strongest continuous NLL.

Scale	Reader-free PPL	Hybrid PPL	Control PPL	Δ [97.5% interval]
40M	60.852	53.258	53.477	$-.00410\ [-.00588, -.00262]$
350M	30.210	28.932	28.952	$-.00072\ [-.00146, +.00005]$

Table 26: Projected-reader secondary ablations. Exact-off removes the candidate's exact gate mass and renormalizes its remaining weights without refitting.

Scale	Reader	Primary 8K PPL	Full 32K PPL
40M	Hybrid	53.258	50.384
	Matched-feature continuous	53.486	50.726
	Hybrid, exact mass removed	53.413	50.675
350M	Hybrid	28.932	27.562
	Matched-feature continuous	28.955	27.611
	Hybrid, exact mass removed	28.948	27.594

Reader-only cost. Two preselected windows (indices 0 and 16), a 4,096-token prefix, 256 sequential predictions, all six checkpoints, two warmups, and eleven rotated repetitions supply the cost audit. The CUDA-graph path includes input staging and trust updates but excludes backbone inference and prefill. All 36 reader-output parity checks pass. The reported reader-time reductions are 4.52%/8.13%, and the state-plus-parameter reductions are 24.6%/25.5%. These results characterize this reader path, not a complete serving system or the compact configuration's runtime.

Table 27: Projected-reader cost. Persistent bytes include reader state and parameters, but not backbone state, temporary allocations, or graph memory pools.

Scale	Reader	Microseconds/token	Persistent reader bytes
40M	Hybrid	24.06	1,519,520
	Strongest continuous	25.19	2,014,844
350M	Hybrid	25.28	2,600,928
	Strongest continuous	27.51	3,489,692

F.3 Earlier consensus-gated hybrid

An earlier configuration divides a 7,680,000-byte ceiling into 2,560,000 bytes for eight token-only exact records per vocabulary entry and the remainder for 5,663 continuous records. It interpolates the exact reciprocal-rank vote with a continuous/base mixture. Development compares continuous weights $\{.1, .2, .35, .5\}$, additional exact weights $\{0, .01, .025, .05, .1, .2\}$, and target-free activation

conditions: any candidate; at least two unanimous candidates; or at least two candidates with one value carrying at least .75 of vote mass. The selected hybrid uses continuous weight .35, exact weight .2, and the .75 majority condition.

The fixed choice is evaluated on six additional, mutually nonoverlapping, document-contained 64K validation windows, using sampling seed 2026091117 and the same three reader-free backbones. These are separate windows in a previously exposed corpus. Reader-free PPL is 65.89, continuous PPL is 57.90, and hybrid PPL is 58.60. Hybrid-minus-continuous NLL is $+.01212$ with the same sign in every seed. A larger, non-payload-matched exact $K = 64$ cosine reader gives PPL 64.16. This comparison illustrates how hybrid quality depends on the allocation and readout used for the two access paths.

G SAME-TARGET CONTINUATION AND STATEFUL EXECUTION

G.1 40M: INDEPENDENT MATCHED BACKBONES AND TWO NESTED HISTORIES

Eight document-contained windows provide a 64K history followed by 256 targets each. The 8K condition uses the suffix of that same history. Every method and history scores the same 2,048 distinct targets, or 6,144 model–target evaluations across three checkpoint pairs (seeds 20260937/41/42). The learned reader uses its co-trained checkpoint. Reader-free and simple exact share a separately trained matched checkpoint; simple exact uses $K = 8$, reciprocal-rank weights, and the development-selected coefficient .025. All arms reset at document boundaries. The simple-minus-matched

Table 28: 40M same-target quality. PPL exponentiates mean NLL across the 24 seed–window cells. Candidate and gate values average scored targets.

History	Reader	PPL	Top-1 (%)	Candidates	Mean gate
8K	Matched no-reader	79.00	26.55	0	0
	Simple exact	73.80	26.66	6.19	.0225
	Learned	73.41	26.92	6.19	.1257
64K	Matched no-reader	78.99	26.56	0	0
	Simple exact	72.65	26.71	7.21	.0244
	Learned	71.80	26.94	7.21	.1422

contrasts are $-.0680$ $[-.1119, -.0241]$ at 8K and $-.0837$ $[-.1343, -.0330]$ at 64K, using eight paired window means as interval units. All eight favor simple exact at both lengths. Learned-minus-simple is $-.0054$ $[-.0355, +.0247]$ and $-.0118$ $[-.0501, +.0264]$, with five of eight windows favoring learned in each condition. Their context interaction is $-.00645$ $[-.02108, +.00817]$. None of these intervals establishes equivalence between readouts.

All 144 arm–history–seed–window terminal checks pass against an independent FP32 full-prefix reference. The maximum full-distribution difference is .003055; independent simple-cache target log probability agrees exactly. A BF16 companion study scores 64 targets per cell, not the full 256-target quality bank, and has aggregate PPL drift at most .0206 against FP32. Two predetermined windows per seed are timed with one warmup and seven synchronized repetitions. Paired timing

Table 29: 40M BF16 stateful companion measurements. Timing averages six cell medians. The PPL column uses the restricted 64-target diagnostic slices, not the preceding 256-target bank.

History	Reader	Decode ms/token	State MiB	Diagnostic PPL
8K	Matched no-reader	12.78	3.679	71.895
	Simple exact	11.78	3.734	67.623
	Learned	15.86	3.951	67.636
64K	Matched no-reader	12.97	3.679	71.917
	Simple exact	11.96	3.926	67.661
	Learned	17.18	4.911	67.641

ratios vary across runs, so reader-time point estimates are reported descriptively. State counts per-request backbone-plus-reader tensor/int64 payload, excluding weights, temporary allocations, Python containers, and process residency.

G.2 125M: SHARED CHECKPOINT WEIGHTS THROUGH 128K

Sixteen held-out PG-19 test documents each supply 256 final targets under nested 32K/64K/128K histories. Three checkpoints trained at 32K (seeds 20263121/22/23) give 4,096 distinct target positions and 12,288 model–target evaluations per method–history condition. Reader-off, simple exact, and learned exact use identical checkpoint weights. Simple exact substitutes the $K = 8$ reciprocal-rank rule with coefficient .025; learned exact retains its scorer, pruning, and gate. All 432 arm–history–seed–document cells complete. Evaluation uses FP32 parameters with BF16 autocast on one RTX 5090. At 32K, 64K, and 128K, the simple-minus-off NLL contrasts are $-.01879, -.01805, -.01808$, with document-level Student-t intervals $[-.03445, -.00314]$,

Table 30: Complete 125M continuation and stateful results. Quality aggregates 48 seed–document cells; times are medians over report cells. State is mean persistent backbone-plus-reader payload.

History	Reader	PPL	Top-1 %	Decode ms/tok	Prefill ms	State MiB
32K	Off	45.341	29.712	15.02	143.55	11.158
	Simple	44.497	29.793	15.12	143.49	11.311
	Learned	44.542	29.899	17.78	160.15	11.926
64K	Off	45.339	29.696	14.76	262.36	11.158
	Simple	44.528	29.785	16.15	264.69	11.402
	Learned	44.728	29.891	18.32	289.57	12.380
128K	Off	45.338	29.728	14.72	482.02	11.158
	Simple	44.525	29.801	15.62	490.01	11.534
	Learned	44.697	29.972	18.95	524.10	13.038

$[-.03351, -.00258]$, and $[-.03342, -.00274]$. Paired document-bootstrap sensitivity intervals are $[-.03426, -.00674]$, $[-.03340, -.00594]$, and $[-.03338, -.00607]$. Thirteen, thirteen, and twelve of sixteen document means favor simple exact. The bootstrap resamples the same observations; it is not a second confirmation.

Learned-minus-simple NLL is $+.00100$ $[-.01196, +.01395]$ at 32K, $+.00447$ $[-.00749, +.01644]$ at 64K, and $+.00386$ $[-.00815, +.01588]$ at 128K. The simple-reader 128K-minus-32K interaction is $+.00071$ $[-.00053, +.00195]$, with bootstrap interval $[-.00043, +.00179]$. The differences do not resolve an increasing benefit from extending history.

Two predetermined documents per seed, arm, and history supply 54 FP32 terminal checks. Every greedy token matches the independent reference, with maximum full-distribution log-probability difference .001609 under the specified .02 tolerance. These are finite-tolerance numerical checks. At 128K, simple exact's additional persistent payload is $11.534 - 11.158 = .376$ MiB. Python objects, weights, temporaries, and process memory remain outside this quantity.

G.3 An inventory audit of the context plateau

The audit constructs occurrence lists directly on CPU from the existing sixteen-document bank, without loading models, tuning a policy, or generating new quality measurements. All 48 document–history cells reproduce saved key coverage, retained hits, opportunities, and final retained-record counts. Counts below use 4,096 distinct targets, not targets multiplied by the checkpoint count. At

Table 31: Inventory changes when extending to 128K on identical targets.

Change at 128K	Relative to 32K	Relative to 64K
Identical candidate pool	3,340 (81.54%)	3,629 (88.60%)
Changed candidate pool	756 (18.46%)	467 (11.40%)
Target support restored	97 (2.37%)	45 (1.10%)
Target support lost	0	0
Target vote increased	136	76
Target vote decreased	117	67

32K, 3,172 queries (77.44%) already have eight records. Their pools cannot change when older history is prepended. Other unchanged pools have no additional same-key occurrence in the added prefix. Target support grows from 1,136 positions at 32K to 1,233 at 128K, but most inventories remain identical. Added records can dilute or increase a normalized target vote even when support is already present. Target labels enter only this descriptive audit, never candidate construction or prediction.

H LONGER-TRAINING CHARACTERIZATION

Two independently trained 124,406,530-parameter reader-free hybrid backbones each receive 5,000,003,584 optimizer-token presentations, sampled with replacement from a fixed 1B-token PG-19 pool. A single warmup–cosine schedule uses 32K training windows and a 20,000-token vocabulary. The intended protocol specified three seeds, but execution ended after two once interim outcomes became available. Both completed runs are retained descriptively; no third seed is imputed, replaced, or represented as completed.

Two separate validation documents calibrate each constant gate over $\{0, .001, .0025, .005, .01, .025, .05, .1, .2, .35\}$. The endpoint quality bank comprises centered 32,768-target windows from the first eight eligible complete PG-19 test documents. Those documents had been used in an earlier reader-adaptation analysis. The endpoint is consequently a reused-document characterization, not fresh held-out confirmation. Every one of nine reader configurations is reported on both completed seeds.

Table 32: Full 5B-presentation endpoint grid. PPL exponentiates mean NLL across equal-sized document windows and seeds. Pipeline time is teacher-forced evaluation of one 32K window.

Reader	PPL	Pipeline ms
No reader	21.8058	85.78
Exact $K = 8$	21.7546	86.40
Exact $K = 16$	21.7363	86.41
Exact $K = 32$	21.7328	86.37
Continuous $W = 8$	21.8058	86.09
Continuous $W = 32$	21.7965	86.41
Continuous $W = 128$	21.7689	87.59
Continuous $W = 512$	21.7144	93.18
Continuous $W = 1{,}761$	21.6776	111.13

Timing and numerical convention. One RTX 5090 evaluates batch-one windows using FP32 backbone parameters and BF16 autocast. The pipeline includes encoding, exact cross-entropy target probabilities, causal candidate construction, reading, mixture scoring, and scalar reduction. Sorting the exact token–position keys occurs inside timing. Each of the first two report documents per seed receives two warmups and five synchronized repetitions; the reported time averages four document-level medians.

Continuous retrieval uses BF16 hidden vectors, normalized FP32 similarity, and inverse temperature 16. Development timing selects among query chunks 128/512/1,024, eager or compiled reduction, and a fused sliding-window kernel. The fused implementation is selected for all five window sizes. Its target-probability error is at most 1.85×10^{-6} under a 10^{-5} sensitivity tolerance, and calibrated document NLL differences remain below 10^{-6}. No report target determines an implementation choice.

Storage is not matched. A dense int32 successor layout with one int32 counter per vocabulary key uses approximately 1.30 MiB at $K = 16$ and 2.52 MiB at $K = 32$, while continuous $W = 32$ uses about .047 MiB in BF16 keys and int64 values. These logical layouts exclude Python objects and transients. The sub-millisecond differences among exact configurations do not distinguish their runtime statistically. The meaningful observation is the small retained exact-memory gain at the low-latency end of the measured grid, not a universal Pareto frontier.

H.1 VALIDATION TRAJECTORY OVER OPTIMIZER-TOKEN PRESENTATIONS

The same eight validation documents are scored at each milestone, with two other documents reserved for gate calibration. These fixed-bank, two-seed results are shown without a seed-population interval. Mean and seed-level NLLs are retained, preserving the originally reported rounding. The exact-reader advantage becomes smaller as training advances, and continuous retrieval has lower mean NLL at every milestone. This trajectory complements the early-training comparisons; it is descriptive and does not isolate training duration from other study differences.

Table 33: Longer-training validation trajectory. Parentheses give the two seed values. This bank is distinct from the endpoint test windows above.

Presentations	Reader off	Exact $K = 8$	Continuous	Exact–off
0.100B	3.9251	3.8941	3.8588	−.0311
	(3.9292,3.9210)	(3.8984,3.8897)	(3.8633,3.8543)	
1.000B	3.2327	3.2291	3.2228	−.0036
	(3.2315,3.2339)	(3.2279,3.2303)	(3.2222,3.2234)	
2.500B	3.1016	3.0987	3.0953	−.0029
	(3.0989,3.1042)	(3.0960,3.1014)	(3.0933,3.0973)	
5.000B	3.0425	3.0400	3.0381	−.0025
	(3.0394,3.0456)	(3.0368,3.0431)	(3.0353,3.0410)	

I FROZEN PRETRAINED NATIVE-TOKEN PREDICTION

This study uses token-level probability mixtures on frozen Qwen3-1.7B, not prompt extraction. Backbone and vocabulary-head weights remain unchanged. Each of three seeds fits a 262,339-parameter reader with $K = 8$ exact-token candidates, rank-64 query/key projections, and a confidence gate. The gate is a $4 \to 32 \to 1$ SiLU MLP followed by sigmoid. Features summarize candidate concentration, normalized-entropy complement, candidate coverage, and nearest-record proximity; an empty inventory forces zero gate.

Fitting, calibration, and reporting. The fit bank contains 64 document-contained 8K windows: 32 PG-19 training books and 32 RedPajama arXiv training documents. One native-token window is taken per document and all 8,191 next-token targets are scored. There are 524,288 unique input tokens; a fixed 256-step schedule visits each window four times, or 2,097,152 input-token exposures per seed. Frozen hidden states and backbone target log probabilities are cached and shared across fits. The reader minimizes mixture NLL with AdamW, learning rate .001, weight decay .01, and gradient-norm cap 1, using one window per update.

Eight other documents per domain calibrate controls. Here the simple reader uses reciprocal-*distance* scores, with coefficient $\rho \in \{0, .5, 1, 2\}$ and constant gate in $\{0, .01, .03, .05, .1, .2, .4\}$. It has the same exact $K = 8$ inventory, but is not the occurrence-rank policy of the from-scratch experiments. The random-scoring control freezes rank-64 query/key projections and score parameters, fitting only its four-feature gate on calibration data for 200 Adam updates at learning rate .001 and minibatch size 8,192. The continuous comparator reads 256 recent full-width BF16 hidden states, with cosine inverse temperature in $\{4, 8, 16, 32\}$ and the same gate grid. This is not a parameter- or payload-matched continuous comparison.

Each report bank contains 24 further documents per domain, or 196,584 targets. Fit, calibration, and report document hashes differ. PG-19 uses training, validation, and test books for these roles; arXiv uses content-separated source partitions. Intervals resample documents, not tokens or pretrained checkpoints. The three fits share a single pretrained backbone.

I.1 FIRST REPORT AND CALIBRATION-ONLY FOLLOW-UP

Table 34: First native-token report bank, with fit seeds 20263921–23. Base, simple, and continuous are common references; fitted readouts vary by seed.

Domain	Seed	Base	Simple	Random+gate	Learned	Continuous
PG-19	21	20.1474	20.0577	20.0604	20.0679	19.8422
	22	20.1474	20.0577	20.0595	20.0673	19.8422
	23	20.1474	20.0577	20.0596	20.0680	19.8422
arXiv	21	4.4445	4.4125	4.4127	4.4056	4.3929
	22	4.4445	4.4125	4.4124	4.4058	4.3929
	23	4.4445	4.4125	4.4129	4.4055	4.3929

On this bank, learned readouts improve over base but are slightly worse than simple exact on PG-19. Only 7–8 of 24 PG-19 documents favor learned over simple, whereas every arXiv document does. Exact candidates include the target on 19.6% of PG-19 and 48.9% of arXiv positions. Continuous retrieval has the lowest PPL in both domains.

The follow-up keeps projections, candidate distributions, pretrained weights, and readout checkpoints fixed. It changes only $g'_t = \alpha_d g_t$, where one domain-specific scalar is shared across seeds. Selection minimizes pooled NLL on the eight reserved calibration documents over $\alpha_d \in \{0, .25, .5, .75, 1, 1.25, 1.5, 2\}$. The selected multipliers are .5 for PG-19 and .75 for arXiv. The choice is frozen before a third, content-disjoint report bank of 24 documents per domain is generated. Simple and continuous controls reuse their earlier calibration settings. The calibrated-minus-simple

Table 35: Calibration-only follow-up on the third report bank. The main text uses these values, not those of the first bank.

Domain	Seed	Base	Simple	Learned	Calibrated	Continuous
PG-19	21	19.6887	19.5915	19.6015	19.5833	19.4199
	22	19.6887	19.5915	19.6008	19.5834	19.4199
	23	19.6887	19.5915	19.6013	19.5837	19.4199
arXiv	21	4.9316	4.9043	4.8982	4.8981	4.8826
	22	4.9316	4.9043	4.8980	4.8980	4.8826
	23	4.9316	4.9043	4.8980	4.8981	4.8826

NLL contrasts are $-.000420, -.000415, -.000399$ on PG-19 and $-.001270, -.001280, -.001277$ on arXiv. Their document-bootstrap upper bounds are at most $-.000286$ and $-.001075$, respectively. These are numerically small readout differences on a single frozen pretrained model, not evidence of large general-purpose gains.

I.2 Denser simple-reader calibration

A post-report robustness analysis calibrates simple exact more finely using only the previously reserved calibration documents. Scores are $-\rho \log(1 + d)$ for distance d, with $\rho \in \{0, .125, \ldots, 4\}$. For each ρ, deterministic derivative bisection minimizes the convex one-dimensional NLL over constant gate $g \in [0, .4]$. The selected (ρ, g) is $(.375, .0068734)$ for PG-19 and $(.25, .0083889)$ for arXiv.

On the same third report bank, tuned simple PPL is 19.59015 and 4.90454. Tuned-simple-minus-calibrated-learned NLL is $+.000329$–$+.000349$ on PG-19 and $+.001314$–$+.001325$ on arXiv; the six document-bootstrap intervals are above zero. The calibrated learned readout retains a small advantage over this denser simple control, while continuous retrieval remains best in PPL. No pretrained-reader latency or serving-memory result is inferred from this quality study.

J Controlled explicit bindings and baseline scope

Controlled systems contain approximately 1.499M parameters with two layers, width 128, vocabulary size 8,192, and tied embeddings. MQAR contains 64 unique bindings and scores every query; training length is 1,024 and evaluation lengths are 1,024, 4,096, and 8,192. Overwrite presents four successive distinct values for each key and asks for the latest completed binding; training length is 4,096. Query-region answers are not supplied as observed successors for later queries.

PCAF and matched reader-free models use 22,500 constant-learning-rate updates at $10^{-2.5}$; the Mamba-3 control uses its specified fused SISO implementation under the same schedule. The separately tuned Llama-style Transformer is a from-scratch full-attention model with RMSNorm, SwiGLU, RoPE, tied embeddings, and Flash-only SDPA. Its development grid compares peak rates $.0003, .001, 10^{-2.5}, .01$, using 22,500 updates, 2,250 warmup updates, and 65,536 optimizer tokens per step. Selection maximizes worst-seed terminal in-distribution accuracy before other tie breaks. These are matched evaluation budgets, not identical hyperparameter-search histories. The tuned Transformer reaches approximately 100% in every 1K seed and partially extrapolates in length. On overwrite, the best worst/mean scores in the development grid are $.830078/.878906$ percent, below

Table 36: MQAR per-query accuracy (%), mean ± sample SD over three seeds. The retrieval-tuned Transformer is a successful in-distribution positive control.

System	1K	4K	8K
PCAF-Exact	$99.97 \pm .04$	$99.96 \pm .04$	$99.91 \pm .09$
Matched no-reader	$.04 \pm .05$	$.07 \pm .05$	$.07 \pm .06$
Mamba-3	12.88 ± 20.87	12.69 ± 20.65	11.97 ± 19.53
Transformer-Llama	$100.00 \pm .01$	48.71 ± 8.85	24.59 ± 4.51

the prespecified 90% worst-seed fitting criterion. We report overwrite as a development result; the main comparison uses the tuned 1K positive control.

J.1 Supervised structural transfer and gate behavior

Single needle, multikey, and variable tracking train at 4K and evaluate at 4K/8K/16K/32K. Each primary cell uses 256 sequences per seed, or 768 total, with seeds 20263211/12/13. Single/multikey require a complete correct answer; variable tracking generates the five-token chain autoregressively. The predefined 32K criterion requires every seed to exceed 90%. Multikey accuracy varies across

Table 37: Primary supervised PCAF transfer accuracy (%). Parentheses give the three seed values at 32K. These are atomic-token adaptations, not zero-shot official RULER scores.

Task	4K	8K	16K	32K	32K seed values
Single needle	100.00	100.00	100.00	100.00	(100,100,100)
Multikey	99.87	99.87	99.48	69.66	(100,9.77,99.22)
Variable tracking	100.00	98.70	99.87	99.87	(100,100,99.61)

seeds, and the prespecified every-seed criterion is not met. At 32K, every query has one valid, selected exact candidate in each seed, while mean gates are 1.000, .269, and .849. The corresponding per-query accuracies are 100.00, 75.10, and 99.90%. In the lowest-performing seed, the candidate remains available while the mixture assigns it a smaller gate. This diagnostic points to probability mixing as an important factor alongside record retention.

A separate same-feature cache diagnostic uses 128 sequences per seed (384 per cell), rather than the 256-sequence primary bank. On that diagnostic bank, the learned reader reaches 71.35% multikey sequence accuracy at 32K, whereas directly returning the latest exact successor reaches 100%. The two learned-reader percentages reflect different banks and must not be combined. Latest-successor lookup also reaches 100% on all five explicit-binding generators in that study.

Continuous baselines receive a development grid over dot/cosine scores, inverse temperatures, recency, and interpolation. Their 11,915-record ceiling covers the complete 8K history. At 8K, same-feature continuous retrieval reaches 70.96% MQAR and 1.02% overwrite accuracy, compared with 99.88% and 99.95% for the learned exact reader on that diagnostic bank. The exact rule is aligned with these generators' discrete binding semantics. These results do not establish semantic retrieval, natural-language reasoning, or superiority to every trained attention architecture.

K Whole-system language-model context

Cross-architecture comparisons contextualize the complete implementation but do not isolate the output reader. They differ in architecture, tuning, compilation, and checkpointing. The capacity-matched same-backbone comparisons remain the attribution evidence. The Llama-style model uses uncompiled non-reentrant layer checkpointing. PCAF and Transformer-LM use compiled/fused paths, while Mamba-3 uses its uncompiled fused implementation. Equal training tokens do not imply equal search budgets. The PCAF–Mamba quality interval includes zero at this operating point. The PCAF/Transformer-LM PPL ratio is .9319 with interval $[.8677, 1.0007]$; the PCAF/Mamba ratio is .9146 $[.9044, .9249]$. The primary 350M matched-control model has 346.435M parameters and PPL 35.07. Its quality is not paired with the throughput of the different 347.065M system.

Table 38: Separate 125M/32K whole-system comparison. Throughput and allocation characterize the respective training implementations, not a common serving benchmark.

System	PPL	Thousands of tokens/s	Peak MiB
PCAF-Exact	55.29	135.03	12,366
Matched no-reader	57.38	135.49	12,373
Transformer-LM	79.04	52.26	12,527
Transformer-Llama	92.83	34.04	4,683
Mamba-3	60.22	20.64	23,371

Table 39: Separate 350M/8K whole-system comparison. The PCAF checkpoint here differs from the primary matched-control checkpoint.

System	Parameters (M)	PPL	Thousands of tokens/s	Peak GiB
PCAF-Exact	347.065	35.18	53.00	9.44
Transformer-LM	347.706	37.75	41.78	9.66
Transformer-Llama	347.686	46.87	27.47	5.81
Mamba-3	347.967	38.46	34.66	15.70

L SEPARATE DETERMINISTIC PROMPT-SELECTOR APPLICATIONS

The pretrained prompt-selector experiments are included to delimit a related application of successor storage, not as evidence for the probability-mixture reader. The query in these experiments identifies a key somewhere in a prompt rather than placing the key at its final token. A deterministic string-extraction rule recognizes explicit mapping syntax, selects a source record, and copies its value. It computes no neural token-level mixture probability. Structured variable-tracking and word-count readouts similarly implement separately specified symbolic operations; their gains must not be assigned to learned cache scoring.

L.1 BOUNDED LOCALIZATION AND ITS INTERFACE

The localized Llama-3.2-1B-Instruct selector takes the final nonempty line of the final user turn as its query region. History and query are tokenized separately. It retains a 40,960-token history, at most 8,192 token buckets, and the latest eight occurrences per bucket. Candidate comparisons examine at most 24 preceding tokens. A qualifying match has length at least three, contains a history anchor occurring at most twice, and leaves at least two query tokens after the match. Accepted source text begins with the declared mapping delimiter `is:` or =, with the specified optional short connector.

Accepted source positions must lie in one cluster with successive gaps at most 24. Selection prioritizes agreement, rarity, rare-anchor count, source recency, and query position. Output stops at the first period or newline after a nonempty value, with a 64-token cap. The reverse-predecessor control emits the same number of tokens preceding the chosen source. Activation selects a copied response; abstention selects the base response. The 8K prompt baseline uses the first 128 and last 8,064 prompt tokens, not a continuously rolling KV cache.

A declared compact int32 layout uses

$$4(40{,}960 + 8{,}192 \cdot 8 + 8{,}192 + 128{,}256) = 971{,}776 \text{ bytes.}$$

This is .92676 MiB of history, positions, bucket IDs, and vocabulary counts. The evaluated Python path additionally retains strings, token lists, counters, dictionaries, and candidates. Scanning and tokenization cost is outside the trained exact reader's fixed-width lookup bound.

L.2 OFFICIAL-GENERATOR RETRIEVAL CONFIRMATION

The frozen model and localized rule are evaluated on 250 new prompts for each of 13 official RULER task configurations (Hsieh et al., 2024) at 16K and 32K, or 6,500 prompts. Eight needle tasks define the task-balanced retrieval endpoint. References are withheld until prediction completion; 10,000 paired within-task bootstrap draws carry all systems together. The selector-minus-full gain is 21.825

Table 40: Task-balanced needle-retrieval accuracy (%) for the separate deterministic selector. These are not neural probability-cache results.

Context	Full base	Full+selector	Predecessor	8K base	8K+selector
16K	70.08	91.90	22.79	35.45	82.43
32K	70.79	87.11	23.60	19.71	71.25

points $[20.325, 23.300]$ at 16K and 16.325 $[14.925, 17.725]$ at 32K. Against the predecessor control, the gains are 69.113 $[68.175, 70.013]$ and 63.513 $[62.512, 64.550]$. Window-plus-selector minus window gains are 46.975 $[45.250, 48.700]$ and 51.537 $[49.950, 53.150]$. All six directional criteria are met.

The five non-retrieval families have zero activation and identical base responses; this is abstention, not demonstrated improvement on those tasks. Multiquery has a .20-point decline at each length. Every activated single-reference output equals the parsed complete reference value (1,382 at 16K and 1,270 at 32K). The rule's structural alignment with explicit mapping prompts is essential to interpreting these results.

L.3 CONFLICT, GRAMMAR, AND NATURAL-PROSE BOUNDARIES

An exposed-prompt stress study finds that the original spatial-cluster rule can accept conflicting values: 216/384 adjacent-conflict and 10/384 distant-conflict prompts activate, while a comparison lexical parser abstains on all 768. The original selector also frequently abstains on duplicated identical mappings and unmodeled delimiters. A separate reject-only guard consequently checks full literal-key equality and retained-value consistency. It cannot create a new activation or modify an accepted output.

In a new finite reader-only bank of 792 conditions from 72 base cases with correlated variants, the guard preserves 144 correct safe activations and rejects all 493 original activations among 576 conflicting, mismatched, or ineligible conditions. These correlated grammar tests do not justify a universal safety guarantee or an independent-binomial error bound. Retrospective replay on the original official prompts retains all 2,654 activations without changing scores. A further prospective 832-prompt official bank retains all 346 active proposals, checking compatibility rather than conflict prevalence in natural documents.

Table 41: Natural-question scope test: 256 SQuAD1.1 questions per condition (Rajpurkar et al., 2016). Original and guarded selectors activate on zero prompts; every system has the same prediction. Scores are base-model performance.

Condition	EM (%)	F1 (%)	Original active	Guarded active
Original paragraph	47.27	65.47	0/256	0/256
With approximately 16K distractors	8.59	22.28	0/256	0/256

Questions are selected by a fixed hash ordering with at most one per paragraph. The long condition inserts the unchanged paragraph among other-article distractors. Both selectors preserve fallback responses and correct no base errors. We report this public-development subset as a subset characterization rather than a hidden-test SQuAD or LongBench (Bai et al., 2024) score; pretraining contamination is not independently evaluated. Its zero activation identifies the explicit-mapping interface's coverage boundary. The neural probability cache is evaluated separately from this QA bank.

L.4 SEPARATE REQUEST COST, INCLUDING INFERENCE BYPASS

An outcome-blind hash rule selects sixteen prompts per task from the completed 32K confirmation, or 208 prompts across thirteen tasks. Synchronized request timers include CPU localization and tokenization, host transfer, prefill, decode, or direct copying, but exclude model loading. Three rotating method orders compare full context, the 8K prompt, and the 8K prompt plus selector.

Across the eight retrieval-task medians, full context takes 1.442 seconds, the 8K prompt .921, and 8K plus selector .582. On 81 of 128 retrieval prompts, activation bypasses model inference: the combined path's median is .045 seconds, against 1.082 for full and .490 for the window. On the 47 abstentions, window-plus-selector is slower in median (1.506 versus 1.428 seconds). Five non-retrieval families always abstain. These single-repeat request measurements concern the original deterministic selector; the optional guard's additional CPU cost is measured separately. Neither these speedups nor the selector's retrieval accuracy should be attributed to the neural exact-memory mixture.