ALL MEMORY IS LOCAL: MEASURING THE HORIZON OF PERSISTENT TEST-TIME TRAINING

Anonymous authors
Paper under double-blind review

ABSTRACT

Test-time training (TTT) lets a language model update a small set of fast weights as it reads. The drop-in methods considered here reset those weights at document boundaries, so nothing carries across documents. When the reset is removed, the model recalls symbols from earlier documents well after they have left the context, which looks like persistent memory. We ask how much of this signal is memory. Delayed recall has at least three other sources: information the model already had (pre-training contamination, in-context copying, synthetic vocabularies shared across sessions), properties of the evaluation (scoring order, length-normalized accuracy), and adaptation to the stream as a whole. We build a protocol, Streaming-TTT, that separates them: per-session obfuscated vocabularies, first-mention targets, fixed-lag re-scoring, and a control of sessions that were never streamed. On 180 code repositories and 200 prose sessions with Qwen3 models from 0.6B to 4B, persistence raises first-mention recall to 7.6 times the frozen model's immediately after a session, but the estimated session-specific component falls by about four fifths within five sessions; by lag 15 what remains cannot be distinguished from the gain on never-streamed sessions. A linear-interference model predicts that this decay depends on subsequent write mass rather than on the learning rate, and the measured curves collapse accordingly. Resetting at session boundaries keeps 60% of the immediate benefit and avoids the general-capability cost that unrestricted persistence incurs. In this setting, delayed recall alone did not establish persistent memory; the relevant quantity is what remains after adaptation is subtracted.

1 INTRODUCTION

Test-time training adapts a language model by updating fast weights, typically the MLP down-projections, as tokens arrive. It appears both in architectures trained for it (Titans, Behrouz et al., 2025b; LaCT, Zhang et al., 2025; TTT-E2E, Tandon et al., 2025) and in drop-in variants applied to pretrained models (In-Place TTT, Feng et al., 2026; TTT-NTP, Ouyang et al., 2026). The drop-in methods considered here reset the fast weights at document boundaries.

Real workloads are streams: a coding assistant reads one repository file after another; an enterprise assistant reads one organization's documents for months. If the reset is simply removed, the model does something that looks like remembering. It recalls identifiers defined in a file it read several files ago, long after that file has left the context window. This paper asks how much of that signal is memory, and how far it reaches.

The question is harder than it sounds, because delayed recall has other sources. Some of the information was never learned at test time at all: on unmodified public repositories, a frozen model already recalls 38.1% of cross-file identifiers, and every mention of a symbol after the first can be copied from context. Some of the signal comes from the evaluation rather than the model: scoring a chunk after it has been written, or using length-normalized accuracy, both move the numbers. And some of it is adaptation to the stream as a whole rather than to any particular session; on prose, this component accounts for 96% of the apparent likelihood gain. These sources did not all become visible at the same stage of the study, and our protocol was refined as they did. Its final form (§4) renames every session-defined symbol with a session-specific vocabulary, scores only first mentions,

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.

re-scores held-out documents at fixed lags after learning, and compares against sessions that were never streamed.

With these controls, the measurement gives a consistent picture across code and prose, with the same qualitative pattern across the model sizes we evaluate. Persistence raises first-mention recall to 7.6 times the frozen model's immediately after a session. The session-specific part of that gain falls by about four fifths within five sessions; by lag 15 the remaining excess ($0.9\,\mathrm{pt}$) lies within the interval of the gain on never-streamed repositories ($0.35\,\mathrm{pt}$, CI [-0.14, 0.91]), and by lag 40 the two coincide. The control therefore turns delayed recall into a measurement of session-specific retention. A linear-interference model (Prop. 1) predicts that the decay depends on how much is written afterwards and not on the learning rate; curves at three learning rates collapse when normalized (Fig. 3). Unrestricted persistence also costs 2–3 pt on HellaSwag and PIQA. Resetting at session boundaries keeps 60% of the immediate benefit with no cross-session exposure. Finally, at equal write budget, gating writes by cross-document sign consistency beats random placement by about one point at lag 0; the effect is larger on larger models and disappears by lag 15.

Our contributions are: (1) a protocol for measuring persistent TTT that separates prior knowledge, evaluation effects, and stream adaptation from session-specific memory, with unit tests and a never-streamed control; (2) measurements on code and prose streams at three model sizes showing that session-specific memory decays within a few sessions and that the long-lag residual is not distinguishable from adaptation; (3) evidence that sign-consistency gating improves short-lag retrieval at equal write budget, with a larger effect on larger models.

2 Related Work

TTT and fast weights. Fast-weight memories date to Schmidhuber (1992) and Ba et al. (2016); linear attention is their modern form (Katharopoulos et al., 2020; Schlag et al., 2021). Contemporary TTT instantiates them as inner-loop learners: TTT-Linear/MLP (Sun et al., 2024), LaCT (Zhang et al., 2025), TTT-E2E (Tandon et al., 2025), Titans (Behrouz et al., 2025b), HOPE (Behrouz et al., 2025a), DeltaNet variants (Yang et al., 2024; 2025), and drop-in retrofits In-Place TTT (Feng et al., 2026) and TTT-NTP (Ouyang et al., 2026); single-input adaptation (Hardt & Sun, 2024; Hübotter et al., 2025). All reset per sequence or document, or require (meta-)training; none measures cross-session retention. We adopt In-Place TTT's fast-weight location and chunk-wise schedule; we vary only whether fast weights persist across documents.

Continuous TTT inside episodes. Agentic TTT (Wang et al., 2026a) and TTT-Discover (Yuksekgonul et al., 2026) update continuously within one agent episode. Their stream is the model's own text; ours is an exogenous document stream across sessions.

Context-to-weights persistence. Temp-LoRA (Wang et al., 2024), StreamAdapter (Muhtar et al., 2024), InfiniteICL (Cao et al., 2025), SCoL (Wang et al., 2026b), CaMeLS (Hu et al., 2023) and MAC (Tack et al., 2024) write context into parameters, several with meta-training, several noted to erode general capability. These works propose mechanisms; we contribute a measurement framework for the training-free case. Concurrently, Jeong (2026) compares trained latent memory banks attached to a frozen encoder–decoder under a cross-session forgetting curve; our object is the LM's own fast weights, training-free, with the lag protocol and controls of §4.

Continual test-time adaptation in vision. CoTTA (Wang et al., 2022), EATA (Niu et al., 2022), SAR (Niu et al., 2023) and Elastic TTT (Ma et al., 2026) stabilize non-reset adaptation for classification and reconstruction, the latter two with Fisher penalties. We evaluate non-reset adaptation as one member of our policy family; these works do not measure recall as a function of lag.

Memory theory. Palimpsest models (Parisi, 1986), cascade synapses (Fusi et al., 2005) and Benna & Fusi (2016) predict decay governed by subsequent write load, not time. Fig. 3 tests this prediction on an LLM's fast weights. Our Fisher gate is a test-time analogue of importance weighting in EWC (Kirkpatrick et al., 2017), SI (Zenke et al., 2017), and MAS (Aljundi et al., 2018); transfer metrics follow Lopez-Paz & Ranzato (2017).

3 PERSISTENT TEST-TIME TRAINING

3.1 SETUP

A pretrained causal LM processes an ordered stream of *sessions* (files of one repository; sections of one article), each a sequence of *documents*. Following In-Place TTT, the fast weights are the MLP down-projections, consistent with evidence that MLP layers store associations (Geva et al., 2021; Meng et al., 2022): $W_{\text{eff}} = W_{\text{base}} + G_{\text{slow}} + G_{\text{fast}}$, both G initialized to zero. Within a document, each chunk of $C{=}256$ tokens is first *scored* under the current W_{eff} (all metrics are prequential: each chunk is scored before it is written, so no chunk is ever scored after the model has learned from it) and only then *written*: $\delta = \text{clip}_{\pm m}(-\eta \nabla_W \mathcal{L}_{\text{NTP}}(\text{chunk}))$, $G_{\text{fast}} \mathrel{+}= \delta$. No optimizer state is kept; the method is training-free and drop-in.

3.2 WHAT COUNTS AS PERSISTENCE?

Three quantities are easy to conflate. *Within-session adaptation* is the gain on a session's held-out document scored immediately after the session (lag 0). *Stream adaptation* is what the model has absorbed about the stream as a whole (vocabulary shape, domain style); we use the gain the final fast-weight state produces on sessions that were never streamed as an estimate of this quantity, since that gain cannot be memory of any particular session. *Session-specific persistence* at lag L is what is left of a session's gain, L sessions after it was learned, once stream adaptation is accounted for. We estimate it as

$$P(L) \;=\; \big[R_{\text{streamed}}(L) - R_{\text{frozen}}\big] \;-\; \big[R_{\text{fresh}} - R^{\text{fresh}}_{\text{frozen}}\big],$$

where R is first-mention recall (or, on prose, log-likelihood), each excess is taken against the frozen model on the same documents, and the second bracket is measured on never-streamed sessions with the state at the end of the stream. The first bracket alone is what a delayed-recall experiment reports; $P(L)$ is the quantity our conclusions are about. Intervals for $P(L)$ combine a session-clustered bootstrap over streamed sessions with an independent one over never-streamed sessions.

3.3 A FAMILY OF PERSISTENCE POLICIES

At each document boundary $G_{\text{slow}} \leftarrow \gamma G_{\text{slow}} + \alpha \odot G_{\text{fast}}$, $G_{\text{fast}} \leftarrow 0$ unifies: **reset** ($\alpha{=}0$), **persist** ($\alpha{=}1, \gamma{=}1$), **decay** ($\gamma{<}1$), **consistency** ($\alpha = \Pi := |\text{EMA}_{\gamma_d}[\text{sign}\, G_{\text{fast}}]|$), **Fisher** ($\alpha = (1 + \hat{F}/\text{med}\,\hat{F})^{-1}$, with $\hat{F}$ a within-document moving average of squared gradients), **surprise** (a scalar α from the document's first-chunk loss relative to a running mean), and the control **random** (α = the consistency gate's values, randomly permuted). **Session** additionally zeroes G_{slow} at session boundaries. For *equal-budget* comparisons gates are normalized to mean one, so gated policies spend exactly persist's write mass and differ only in *where* they write.

3.4 A LINEAR MODEL OF WHAT PERSISTS

Model $G = \sum_k \Delta_k$ as a linear associative memory with write masses $m_k = \|\Delta_k\|_F$ and mutually isotropic sessions (a design property of the benchmark: per-session synthetic vocabularies).

Proposition 1 (interference decay)**.** *Under A1–A2 (App. A), probing session k at lag L with a unit vector v uniform on the sphere and independent of $\{\Delta_j\}_{j\neq k}$,*

$$\text{SNR}(k, L) \;:=\; \frac{\|\Delta_k v\|}{\sqrt{\mathbb{E}\,\big\|\sum_{j\neq k,\, j\leq k+L} \Delta_j v\big\|^2}} \;=\; \sqrt{d}\, c_k \sqrt{\rho(k, L)},$$

where $c_k := \|\Delta_k v\|/m_k \leq 1$ and $\rho(k, L) = m_k^2 / \sum_{j\neq k,\, j\leq k+L} m_j^2$.

The denominator splits into sessions before k (independent of L) and sessions after; with near-uniform masses and K_k sessions before k, $\rho \approx 1/(K_k + L)$ and η cancels. The proposition conditions on A1–A2 and does not model the full transformer. Two falsifiable predictions follow: excess recall is a function of ρ alone, and lag curves are *invariant to* η; faster decay at large η isolates effects outside the model (clip saturation, base drift). We use the model as a measurement instrument, not as a theory of transformers. This model yields a testable prediction for the learning-rate sweep in §5.5.

Proposition 2 (what the consistency gate keeps)**.** *If a coordinate's write sign is* $+1$ *i.i.d. across documents w.p.* p*, then* $\mathbb{E}[\Pi] \geq |2p-1|$ *with equality as* $\gamma_d \to 1$*.*

The gate therefore passes systematically repeated directions and suppresses one-shot ones, concentrating write mass along whatever recurs (a session's identifiers on code, a stream's slot statistics on prose).

Remark 1 (session)**.** *Zeroing* G_{slow} *at session boundaries makes cross-session signal and interference identically zero while within-session updates equal persist's; session is the* $L{=}0$ *endpoint of the persistence–risk trade-off.*

Proofs and a boundedness result are in App. A.

4 The Streaming-TTT Protocol

Delayed recall can come from three sources other than session-specific memory. The protocol removes or measures each; implementation details are in App. B–D.

Information the model already has. On unmodified public repositories a frozen model recalls 38.1% of cross-file identifiers; after renaming, 1.7%. We therefore rename every session-defined symbol to a synthetic name drawn from a session-specific random source (a per-session substitution cipher on prose, so that character statistics are constant across sessions while strings are disjoint). The random source must be independent per session: a shared one produces a vocabulary the model learns across the stream, which appears as a long-lag memory that survives every other control (App. E, Fig. 6). Within a document, only the *first mention* of a symbol introduced in an earlier document is scored; later mentions are predictable from context and are recorded separately.

Properties of the evaluation. Each chunk is scored before it is written (§3.1), so no chunk is scored after the model has learned from it; these invariants are enforced by unit tests (App. D). Each session's held-out document, never streamed, is re-scored at lags $\{0, 5, 15, 40\}$ sessions after learning, lag 0 before any boundary transition, so every session contributes to every lag it reaches. General capability is reported under both raw and length-normalized accuracy, because the two can move in opposite directions (§5.6).

Adaptation to the stream. The final fast-weight state is scored on sessions that were never streamed; its excess over the frozen model on those sessions estimates stream-level adaptation and is subtracted from streamed-session excess to give $P(L)$ (§3.2). Gated policies are compared at equal write budget (gates normalized to mean one) and against a random-placement control with the same gate histogram, so that differences between them reflect where they write and not how much.

Statistics. Targets within a session are correlated, so all intervals are session-clustered bootstrap 95% CIs and policy comparisons are paired on shared held-out documents. Three hypotheses were fixed in an internal protocol document before the main 1.7B experiments (dated copy in the supplement): H1 persist $>$ reset at lag 0; H2 persist $>$ frozen at lag 40; H3 consistency and random are equivalent at equal budget (TOST, $\pm 2\,\text{pt}$). Seeds resample stream order and vocabularies. On prose the frozen recall is 0.3%, so the likelihood readout is primary there; the rule (frozen recall $< 1\%$) depends only on the frozen model.

5 Results

5.1 Persistent TTT produces a strong short-lag effect

At lag 0, persist reaches 7.1% first-mention recall against 0.9% for the frozen model (paired excess $6.2\,\text{pt}$, CI [5.0, 7.5]; H1), a $7.6\times$ ratio; the likelihood gap is $3510 \to 398$ PPL (Fig. 1). Per-document reset is indistinguishable from frozen (Table 1). Whatever persistence provides, it is provided across documents, not within them.

5.2 The apparent memory tail

The excess falls to $1.6\,\text{pt}$ at lag 5, $0.9\,\text{pt}$ at lag 15, and $0.67\,\text{pt}$ at lag 40 (CI [0.29, 1.05]), still above frozen; H2 as stated is confirmed. Read on its own, this curve suggests a memory that decays quickly

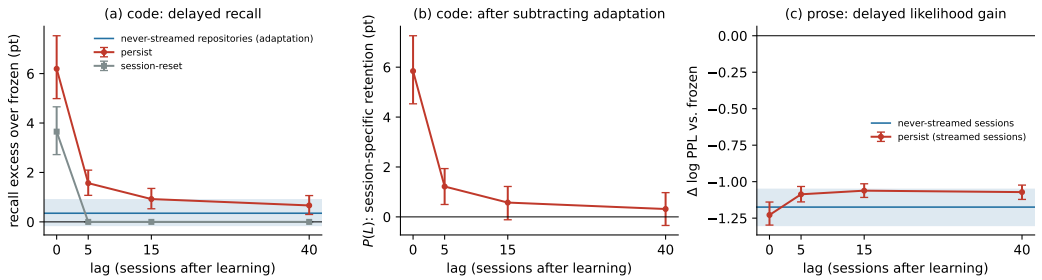


Figure 1: Delayed recall against the never-streamed control (Qwen3-1.7B; session-clustered 95% intervals). (a) Code: first-mention recall excess over the frozen model for persistent TTT and the session-reset control at lags 0–40 (3 seeds); the band is the excess of the final persist state on repositories never streamed. (b) The same data after subtracting the band, i.e. $P(L)$ of §3.2; intervals combine both bootstraps. (c) Prose (3 seeds): likelihood gain on streamed sessions and on sessions never streamed.

and then persists at a low level for at least forty sessions. That reading is the one a delayed-recall experiment invites, and it is the one we test next.

5.3 The never-streamed control changes the interpretation

The final persist state, scored on repositories that were never streamed, exceeds the frozen model by $0.35\,\mathrm{pt}$ (CI [-0.14, 0.91]). This gain cannot be memory of any session; it is adaptation to the stream. It is of the same size as the lag-40 residual, and each interval contains the other's point estimate. On prose the same comparison is sharper: the lag-0 likelihood gain of -1.23 log-PPL is 96% matched by never-streamed sessions (-1.17, CI [-1.30, -1.05]). We therefore do not treat the long-lag result as evidence for session-specific memory. H2 holds in the form in which it was stated; what changed is what it measures. Fig. 1 shows the never-streamed level as a band; the persist curve enters it between lag 5 and lag 15.

5.4 What remains: session-specific retention

Subtracting stream adaptation (§3.2) gives $P(0) = 5.8\,\mathrm{pt}$, $P(5) = 1.2\,\mathrm{pt}$, $P(15) = 0.6\,\mathrm{pt}$ (CI [-0.1, 1.2]), and $P(40) = 0.3\,\mathrm{pt}$ (CI [-0.4, 1.0]) on code; the last two intervals include zero. On prose, $P(L)$ includes zero at every lag. Session-specific retention is therefore real and large immediately after a session, falls by roughly four fifths within five sessions, and is not distinguishable from zero beyond fifteen. We do not conclude that it is absent; we conclude that our measurement cannot separate it from adaptation past that point. Lag-0 and lag-5 excess follow the same pattern at 0.6B and 4B (Fig. 2a).

5.5 Write load, not learning rate, predicts the decay

Prop. 1 was formulated before the learning-rate sweep and predicts that lag curves at different η coincide once normalized by their lag-0 excess. At 0.6B, $\eta \in \{2, 3, 4\} \times 10^{-2}$ give normalized lag-5 retention of 0.27, 0.25, 0.21 (Fig. 3). The slight ordering is in the direction one would expect from clip saturation; we did not test this directly. This write-mass dependence is consistent with the palimpsest picture of Parisi (1986).

5.6 Persistence has a capability cost

We use $\eta = 3 \times 10^{-2}$, just below the capability cliff observed at 0.6B (between 3 and 4×10^{-2} the mean HellaSwag/ARC-Easy delta drops from $+0.9$ to $-6.9\,\mathrm{pt}$; 500 items per task). Even below that point, persistent policies at 1.7B change by $-3.0\,\mathrm{pt}$ on HellaSwag (Zellers et al., 2019) and $-2.7\,\mathrm{pt}$ on PIQA (Bisk et al., 2020) under raw accuracy, and the same under length normalization (Fig. 4a). ARC-Easy (Clark et al., 2018) moves $+3.7\,\mathrm{pt}$ under length normalization but only $+0.1\,\mathrm{pt}$ in raw

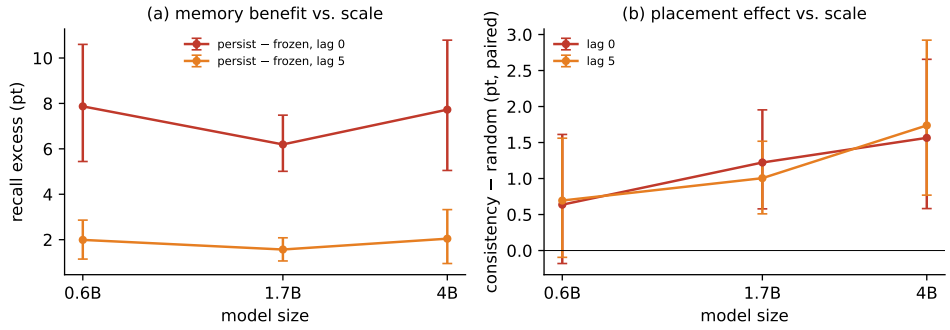


Figure 2: Scale dependence (0.6B: 1 seed; 1.7B: 3 seeds; 4B: 1 seed; paired, clustered CIs). (a) Lag-0 excess is similar across scales. (b) Consistency − random at equal budget grows with scale: 0.64, 1.22, 1.56 pt at lag 0.

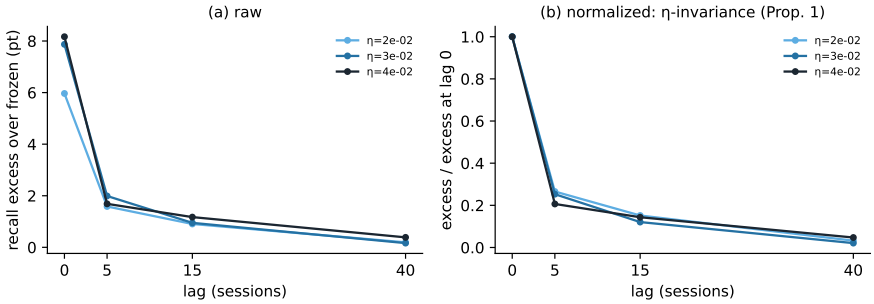


Figure 3: Learning-rate invariance (0.6B, persist − frozen). (a) Raw excess differs by η. (b) After normalizing by lag-0 excess the curves collapse, as Prop. 1 predicts.

accuracy: the adaptation reshapes per-token likelihood in a way the normalized rule rewards without changing which answers are correct. We report both rules. At 4B the HellaSwag cost is comparable (Fig. 4b; single seed).

5.7 Session-level reset as an operating point

Session keeps 60% (45–72% across seeds) of persist's lag-0 excess in recall and 48% in log-likelihood, has no cross-session exposure (Remark 1), and returns the model to its base state at stream end; within a session it carries the same exposure as persist. The paired session–persist difference at lag 0 is -2.5 pt (CI [-3.4, -1.8]; Fig. 5b), of which stream adaptation explains 0.35 pt; the remainder is not explained by our current controls. For the setting studied here, session reset keeps most of the short-lag benefit while avoiding the cost above and the cliff. Persistence adds two to three fast-weight-sized buffers and reduces throughput to $\approx 0.4\times$ frozen inference; session needs no gate state.

5.8 Selective writing improves short-lag recall, not the horizon

At equal write budget, consistency gating beats random placement by 1.22 pt at lag 0 (CI [0.59, 1.91]) and 1.00 pt at lag 5, and not beyond; random is indistinguishable from persist throughout. The difference is significantly nonzero (CI excludes zero) but lies within the prespecified ± 2 pt equivalence margin, so H3 as stated is not rejected at 1.7B. It is absent at 0.6B and larger at 4B (Fig. 2b; single seeds). On never-streamed repositories consistency and random tie, so on code the gate improves session-specific retention rather than adaptation; on prose it improves adaptation instead (App. F), as Prop. 2 would predict for a gate that keeps whatever recurs. Decay, Fisher, and surprise gates are in App. F.

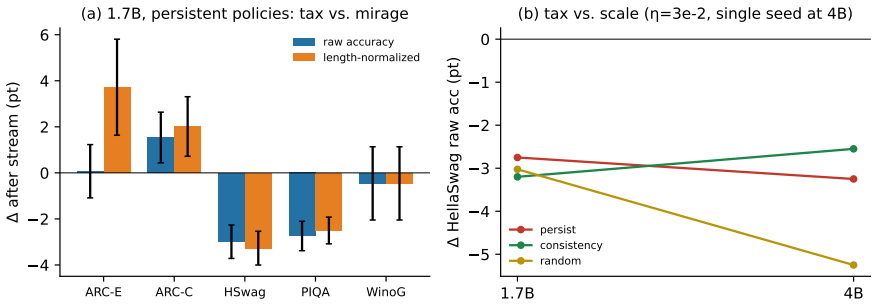


Figure 4: General-capability cost (a) and its scale dependence (b). At 1.7B, HellaSwag and PIQA losses appear under both scoring rules; the ARC-Easy gain under length normalization is a scoring artifact (raw accuracy is flat).

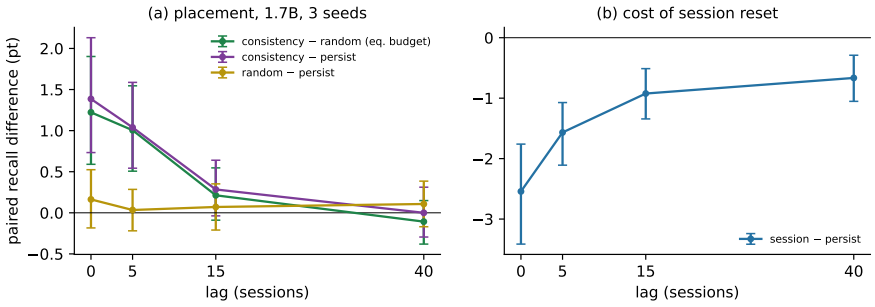


Figure 5: Placement and the cost of reset (1.7B, 3 seeds, paired). (a) Consistency beats random at lags 0–5; random is indistinguishable from persist. (b) Session's deficit relative to persist shrinks from -2.5 to -0.7 pt.

6 Discussion

What the experiments establish. Under the controls of §4, removing the document-boundary reset produces a large, session-specific gain immediately after a session ($P(0) = 5.8$ pt) that falls by about four fifths within five sessions and is still detectable at lag 5 ($P(5) = 1.2$ pt, CI [0.5, 2.0]). The decay is predicted by subsequent write mass rather than learning rate, as Prop. 1 anticipated before the sweep was run. Unrestricted persistence costs 2–3 pt on two commonsense benchmarks under both scoring rules, and one length-normalized benchmark rises without any change in raw accuracy. At equal write budget, gating by sign consistency shifts a small amount of recall into short lags on code and into adaptation on prose.

What they do not establish. They do not show that session-specific memory is absent beyond fifteen sessions; they show that our measurement cannot separate it from stream adaptation there ($P(15)$ and $P(40)$ include zero). They do not show that delayed recall is uninformative in general, only that in this setting it did not identify persistence without the never-streamed control. The linear model reproduces the decay's dependence on write mass under isotropy that the benchmark itself induces; natural streams are not isotropic, and we have not tested the collapse on them. The residual persist–session gap at lag 0 (2.5 pt) is not explained by our current controls; faster within-session learning from a warm-started state is one candidate we have not isolated.

Practical implication. For training-free fast-weight TTT of the kind studied here, resetting at session boundaries keeps most of the short-lag benefit, returns the model to its base state at stream end, and avoids the learning-rate cliff. Persisting across sessions buys a further 2.5 pt at lag 0 and a residual that we cannot attribute, at a measurable capability cost. Which side of that trade is right depends on the deployment; we would not persist across sessions by default.

Table 1: Main results (Qwen3-1.7B, 180 repositories, 3 seeds). Recall in % with session-clustered 95% CI; first-token PPL. †Equal write budget. 124 held-out documents per seed, 2045 first-mention targets.

	recall (%) at lag				PPL at lag			
policy	0	5	15	40	0	5	15	40
frozen	0.9 [0.7,1.2]	1.0 [0.7,1.2]	1.0 [0.7,1.3]	1.0 [0.7,1.3]	3510	3437	3451	3556
persist	7.1 [6.0,8.4]	2.5 [2.0,3.0]	1.9 [1.6,2.3]	1.7 [1.3,2.2]	398	678	825	934
session-reset	4.6 [3.7,5.5]	1.0 [0.7,1.2]	1.0 [0.7,1.3]	1.0 [0.7,1.3]	1229	3437	3451	3556
consistency†	8.5 [7.2,10.0]	3.6 [2.9,4.3]	2.2 [1.8,2.6]	1.7 [1.3,2.2]	358	578	747	873
random†	7.3 [6.1,8.6]	2.6 [2.1,3.0]	2.0 [1.6,2.4]	1.8 [1.4,2.2]	401	669	823	925

Limitations. The fresh-session control has 65 repositories (43 with qualifying held-out files) and 28 prose sessions, and its intervals are correspondingly wide. Three seeds at 1.7B were the largest replication we could afford on a single A100; 0.6B and 4B are single-seed, so the size dependence in Fig. 2 is a trend, not a scaling result. Our learning rate sits just below the capability cliff on code and was calibrated separately for prose; other streams will need their own calibration. Decay, Fisher, and surprise gates were compared only on the pre-fix stream (App. F). Learned gates, replay, and TTT architectures trained for persistence are outside our scope. Persistent adaptation on user streams raises retention and privacy questions; session reset is also the conservative default there.

7 Conclusion

For the training-free, fast-weight TTT setting studied here, memory is local. Persistence helps within a session but decays with subsequent write load; what survives beyond 15 sessions cannot be distinguished from adaptation to the stream. Resetting at session boundaries captures most of the benefit without the benchmark cost or privacy exposure. The question for future work is how much write load a memory can survive before it is overwritten.

AI use statement

We used a generative AI assistant (Claude, Anthropic) extensively as a research and engineering assistant. Its contributions included proposing and refining candidate hypotheses, suggesting experimental controls and protocol details, assisting with the implementation of the synthetic-stream and obfuscation pipeline, developing and checking drafts of the linear-interference model and its propositions, writing and editing code, generating figures from raw results, suggesting experimental parameters, drafting the manuscript, and identifying related work.

The authors determined the research question, central scientific argument, experimental objectives, and final interpretation of the results. In particular, the identification of session-specific persistence from the never-streamed control, the definition of the persistence estimand, and the conclusions drawn from the experimental controls were determined by the authors based on the empirical evidence. AI-generated hypotheses, model formulations, proofs, and interpretations were treated as proposals and were independently reviewed and revised by the authors.

All experiments were run by the authors. Numerical results reported in the paper were regenerated from raw experimental logs using scripts inspected by the authors. The authors take responsibility for the final claims, analyses, code, figures, and manuscript.

Reproducibility statement

Experimental code, stream construction scripts, obfuscation pipelines, and unit tests are provided as supplementary material. Hyperparameters are listed in App. C; all base checkpoints are public (Qwen3-0.6B/1.7B/4B-Base). Bootstrap seeds and stream orderings are included in the supplementary material. Evaluation uses the public lm-eval-harness with fixed few-shot seeds.

Ethics statement

Persistent adaptation raises retention and privacy questions; we recommend session-reset as the privacy-conservative default. The code stream uses only MIT-licensed repositories; the repository list with licenses is included in the supplement. No human subjects were involved.

References

Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. Memory aware synapses: Learning what (not) to forget. In *ECCV*, 2018.

Jimmy Ba, Geoffrey Hinton, Volodymyr Mnih, Joel Z. Leibo, and Catalin Ionescu. Using fast weights to attend to the recent past. In *NeurIPS*, 2016.

Ali Behrouz, Meisam Razaviyayn, Peilin Zhong, and Vahab Mirrokni. Nested learning: The illusion of deep learning architectures. *arXiv:2512.24695*, 2025a.

Ali Behrouz, Peilin Zhong, and Vahab Mirrokni. Titans: Learning to memorize at test time. *arXiv:2501.00663*, 2025b.

Marcus K. Benna and Stefano Fusi. Computational principles of synaptic memory consolidation. *Nature Neuroscience*, 19, 2016.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. Piqa: Reasoning about physical commonsense in natural language. In *AAAI*, 2020.

Bowen Cao, Deng Cai, and Wai Lam. Infiniteicl: Breaking the limit of context window size via long short-term memory transformation. *arXiv:2504.01707*, 2025.

Peter Clark et al. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv:1803.05457*, 2018.

Xiao Feng, Ziyao Luo, Chengye Hua, Zhiyuan Zhang, Xu He, Silin Huang, and Deng Cai. In-place test-time training. In *ICLR*, 2026. arXiv:2604.06169.

Stefano Fusi, Patrick J. Drew, and L. F. Abbott. Cascade models of synaptically stored memories. *Neuron*, 45(4), 2005.

Leo Gao et al. A framework for few-shot language model evaluation. Zenodo, 2023.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. Transformer feed-forward layers are key-value memories. In *EMNLP*, 2021.

Moritz Hardt and Yu Sun. Test-time training on nearest neighbors for large language models. In *ICLR*, 2024.

Nathan Hu, Eric Mitchell, Christopher D. Manning, and Chelsea Finn. Meta-learning online adaptation of language models. In *EMNLP*, 2023.

Jonas Hübotter, Sascha Bongni, Ido Hakimi, and Andreas Krause. Efficiently learning at test-time: Active fine-tuning of llms. In *ICLR*, 2025.

Hong Jeong. Trained persistent memory for frozen encoder–decoder llms: Six architectural methods. *arXiv:2603.16413*, 2026.

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

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. *PNAS*, 114(13), 2017.

David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. In *NeurIPS*, 2017.

Ziqiao Ma, Xueyang Yu, Haoyu Zhen, Yuncong Yang, Joyce Chai, and Chuang Gan. Fast spatial memory with elastic test-time training. *arXiv:2604.07350*, 2026.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in gpt. In *NeurIPS*, 2022.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. In *ICLR*, 2017.

Dilxat Muhtar, Yelong Shen, Yaming Yang, Xiaodong Liu, Yadong Lu, Jianfeng Liu, Yuefeng Zhan, Hao Sun, Weiwei Deng, Feng Sun, Xueliang Zhang, Jianfeng Gao, Weizhu Chen, and Qi Zhang. Streamadapter: Efficient test time adaptation from contextual streams. *arXiv:2411.09289*, 2024.

Shuaicheng Niu, Jiaxiang Wu, Yifan Zhang, Yaofo Chen, Shijian Zheng, Peilin Zhao, and Mingkui Tan. Efficient test-time model adaptation without forgetting. In *ICML*, 2022.

Shuaicheng Niu, Jiaxiang Wu, Yifan Zhang, Zhiquan Wen, Yaofo Chen, Peilin Zhao, and Mingkui Tan. Towards stable test-time adaptation in dynamic wild world. In *ICLR*, 2023.

Xuan Ouyang, Zefan Cai, and Junjie Hu. Test-time training with next-token prediction. *arXiv:2606.21803*, 2026.

Giorgio Parisi. A memory which forgets. *Journal of Physics A*, 19(10), 1986.

Qwen Team. Qwen3 technical report. *arXiv:2505.09388*, 2025.

Gobinda Saha, Isha Garg, and Kaushik Roy. Gradient projection memory for continual learning. In *ICLR*, 2021.

Imanol Schlag, Kazuki Irie, and Jürgen Schmidhuber. Linear transformers are secretly fast weight programmers. In *ICML*, 2021.

Jürgen Schmidhuber. Learning to control fast-weight memories: An alternative to dynamic recurrent networks. *Neural Computation*, 4(1), 1992.

Yu Sun, Xinhao Li, Karan Dalal, Jiarui Xu, Arjun Vikram, Genghan Zhang, Yann Dubois, Xinlei Chen, Xiaolong Wang, Sanmi Koyejo, Tatsunori Hashimoto, and Carlos Guestrin. Learning to (learn at test time): Rnns with expressive hidden states. *arXiv:2407.04620*, 2024.

Jihoon Tack, Jaehyung Kim, Eric Mitchell, Jinwoo Shin, Yee Whye Teh, and Jonathan Richard Schwarz. Online adaptation of language models with a memory of amortized contexts. In *NeurIPS*, 2024.

Arnuv Tandon, Karan Dalal, Xinhao Li, Daniel Koceja, Marcel Rød, Sam Buchanan, Xiaolong Wang, Jure Leskovec, Sanmi Koyejo, Tatsunori Hashimoto, Carlos Guestrin, Jed McCaleb, Yejin Choi, and Yu Sun. End-to-end test-time training for long context. *arXiv:2512.23675*, 2025.

Qin Wang, Olga Fink, Luc Van Gool, and Dengxin Dai. Continual test-time domain adaptation. In *CVPR*, 2022.

Yan Wang, Dan Ma, and Deng Cai. With greater text comes greater necessity: Inference-time training helps long text generation. *arXiv:2401.11504*, 2024.

Yanbo Wang, Jinhua Hao, Yuze Shi, Kun Yuan, and Ming Sun. No time like the present: Agentic test-time training. *arXiv:2607.03441*, 2026a.

Zekun Wang, Anant Gupta, Zihan Dong, and Christopher J. MacLellan. Self-consolidating language models: Continual knowledge incorporation from context. *arXiv:2605.07076*, 2026b.

Songlin Yang, Bailin Wang, Yu Zhang, Yikang Shen, and Yoon Kim. Parallelizing linear transformers with the delta rule over sequence length. In *NeurIPS*, 2024.

Songlin Yang, Jan Kautz, and Ali Hatamizadeh. Gated delta networks: Improving mamba2 with delta rule. In *ICLR*, 2025.

Mert Yuksekgonul, Daniel Koceja, Xinhao Li, Federico Bianchi, Jed McCaleb, Xiaolong Wang, Jan Kautz, Yejin Choi, James Zou, Carlos Guestrin, and Yu Sun. Learning to discover at test time. *arXiv:2601.16175*, 2026.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence? In *ACL*, 2019.

Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In *ICML*, 2017.

Tianyuan Zhang, Sai Bi, Yicong Hong, Kai Zhang, Fujun Luan, Songlin Yang, Kalyan Sunkavalli, William T. Freeman, and Hao Tan. Test-time training done right. *arXiv:2505.23884*, 2025.

A PROOFS

Setup. $G \in \mathbb{R}^{d_{\text{out}} \times d}$; sessions write $\Delta_1, \Delta_2, \ldots$; $m_k = \|\Delta_k\|_F$. A1 (independence): $\{\Delta_j\}$ independent with $\mathbb{E}\Delta_j = 0$ over vocabulary randomization. A2 (isotropy): a probe v of session k is a unit vector uniform on the sphere, independent of $\{\Delta_j\}_{j \neq k}$ (it may depend on Δ_k).

Proof of Prop. 1. For $j \neq k$, v is independent of Δ_j by A2, so $\mathbb{E}\|\Delta_j v\|^2 = \operatorname{tr}(\mathbb{E}[\Delta_j^\top \Delta_j]\,\mathbb{E}[vv^\top]) = m_j^2/d$; cross terms vanish by A1. Interference power at lag L is $\sum_{j \neq k,\, j \leq k+L} m_j^2/d$; with $c_k := \|\Delta_k v\|/m_k \leq 1$ (Cauchy–Schwarz), $\mathrm{SNR} = \sqrt{d}\, c_k m_k / \sqrt{\sum_{j \neq k,\, j \leq k+L} m_j^2}$. □ Recall accuracy is a monotone function of SNR under a fixed-margin readout; the collapse test uses only monotonicity. If $m_j = \eta t_j \kappa(\eta)$, ρ is η-free whenever κ is.

Proof of Prop. 2. $D_t = \gamma_d D_{t-1} + (1-\gamma_d) s_t$, s_t i.i.d. ± 1, $P(+1) = p$. Stationarity gives $\mathbb{E}D = 2p-1$, $\mathrm{Var}D = \frac{1-\gamma_d}{1+\gamma_d}(1-(2p-1)^2)$; $\mathbb{E}|D| \geq |\mathbb{E}D|$ (Jensen) with equality as $\mathrm{Var} \to 0$. □

Boundedness. For gates with $\alpha \leq 1$: element-wise clipping by m over N parameters gives $\|\delta\|_F \leq m\sqrt{N}$, hence $\|G\|_F \leq m\sqrt{N}\,T$ after T writes and $\leq m\sqrt{N}\,T_s/(1-\gamma)$ under boundary decay γ. The equal-budget cap of $4\times$ scales the bound accordingly.

Remark 1. Immediate from the definition: $G_{\text{slow}} = 0$ at the start of every session, so for any probe of session k at lag ≥ 1 both signal and interference vanish; within a session the update sequence equals persist's.

B STREAMS

Code. 180 MIT-licensed Python repositories created after 2025-06 (30–1500 stars), files in import-dependency order, at most 8192 tokens per file; module- and class-level definitions (≥ 3 characters, non-dunder) renamed to random lowercase strings with a per-repository RNG. Held-out: the last file with ≥ 5 first-mention targets (124/180 qualify). 65 further repositories (43 with qualifying held-out files) are never streamed. **Prose.** WikiText-103 (Merity et al., 2017) articles ($\geq$ 8k characters) split into $\approx$ 1200-token documents (3–12 per session); recurring capitalized phrases renamed with a per-session substitution cipher; 203 stream sessions, 28 fresh. Tokenization: Qwen3 tokenizer; targets scored at the position predicting the first sub-token.

C HYPERPARAMETERS

Chunk 256, context 1024 within document, $\eta = m = 3 \times 10^{-2}$ (code; calibrated to 10^{-2} on prose to match the tax), $\gamma_d = 0.9$, gate normalization capped at $4\times$, all layers' down-projections, bf16. Models: Qwen3-0.6B/1.7B/4B-Base (Qwen Team, 2025). General capability: lm-eval-harness (Gao et al., 2023), 2000 items per task, fixed few-shot seeds, raw and length-normalized accuracy both recorded.

D LEAK-FREEDOM TESTS

Unit tests assert: (1) eval-only passes never mutate parameters; (2) each recorded chunk loss equals a frozen model holding the pre-write state; (3) reset equals processing documents in isolation; (4) persist $\equiv$ decay(γ=1); (5) the random gate's histogram equals the consistency gate's exactly. Held-out documents are scored before any boundary transition at lag 0.

E SIX ARTIFACTS

(1) Pre-training contamination (38.1%→1.7%). (2) In-context copying (repeat mentions ≈65% vs. first mentions ≈1% for frozen). (3) Shared synthetic vocabulary across sessions (Fig. 6). (4) Boundary ordering: scoring lag 0 after a session reset reads frozen. (5) Length-normalized scoring (Fig. 4a). (6) Stream-level slot adaptation on prose (Fig. 1c).

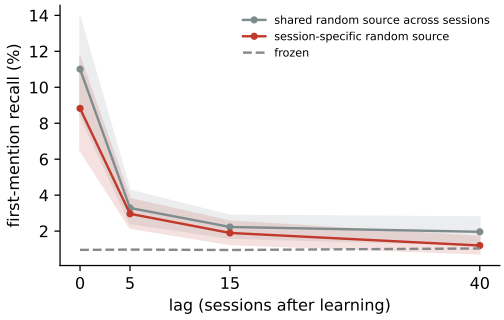


Figure 6: A shared random source for synthetic names across sessions (0.6B, pre-fix stream) inflates recall at every lag and roughly doubles it at lag 40; a session-specific source removes the effect. Frozen model dashed.

F ADDITIONAL POLICIES

Decay, Fisher, and surprise gates, and an activation-subspace projection (GPM-style, k=64; Saha et al., 2021), were evaluated at 0.6B on 50- and 200-repository streams *before* the per-session-vocabulary fix (App. E, item 3). Only their lag-0 ordering is therefore interpretable: all three gates were dominated by persist, and their apparent "retention" tracked how little they wrote; the projection halved lag-0 plasticity. Its apparent gain at lags 15–40 (≈ 0.5 pt) lies within the shared-vocabulary inflation and is not claimed. The 6×10^{-2} learning-rate point comes from the same pre-fix stream. Tables are in the supplement, labeled accordingly. On prose, consistency lowers log-PPL by 0.10 relative to random at lag 0 (CI [-0.14, -0.07]) and by a similar margin on never-streamed sessions (-0.17), so its advantage there is adaptation rather than session memory.

G HYPOTHESES FIXED IN ADVANCE

H1 confirmed. H2 confirmed as stated; interpretation revised by the fresh-session control (§5.3). H3: equivalence within ± 2 pt holds at 0.6B and 1.7B; at 4B the single-seed intervals (upper bounds 2.6–2.9 pt) are too wide to establish it. A nonzero difference of about 1–1.7 pt at lags 0–5 is detected at 1.7B and 4B, and not at 0.6B. All other comparisons are exploratory.