Why a Finite-Capacity Neural Memory Does Not Help: A Controlled Negative Result on Latent-Slot and Recurrent Memory for Text Embedding

Kru Zeppesey

Jaume Project · ModelScope: LightNeko/Jaume

Abstract. We ask whether two common designs for neural memory — learnable latent slots (JM-S) and a recurrent state carried across input chunks (JM-R) — improve a 350M-parameter bidirectional text encoder beyond strong parameter-matched baselines. Under pre-registered criteria applied to independent holdouts and to synthetic tasks whose shortcuts we explicitly bound, both fail. JM-S trails a parameter-matched attention-pooling control by 0.0168 Spearman on SICK-R ($\approx 3\times$ seed standard deviation), and a controlled repair that successfully differentiates the slots makes performance *worse*. For JM-R, a state carried across chunks is 0.0785 F1 *below* using the chunk's own representation, consistently across 5 runs and 3 splits, while a wrong (permuted) state is 0.0321 above the true state's replacement — evidence that the state does carry paper-level information but is unsuitable as a per-chunk representation. Rather than report only the negative outcome, we contribute the diagnostic apparatus that makes it interpretable. We localize the failure to four separable stages of the state pathway: (i) the output never reads the carried state, so memory is invisible at inference; (ii) the additive update dilutes each new chunk, with relative state change decaying from 0.318 to 0.098 over 12 chunks while consecutive-state cosine approaches 0.999; (iii) the gating network is extensively trained (weights move 141%) yet degenerates to a near-constant $g \approx 0.52$ (SD $0.132 \to 0.006$); and (iv) forcibly sweeping the gate over $[0.1, 0.99]$ does not move accuracy outside chance ± 2SE, so the gate is not the constraint. We further report five methodological findings that invalidated earlier conclusions in our own pipeline — including a lexical-ceiling baseline that exposes four of six memory probes as solvable without memory, a filtering leak that underestimated a BM25 baseline by $2.2\times$, and evidence that full fine-tuning on 514 examples destroys a pretrained encoder's block-level signal at every learning rate tested. We release all code, configurations, and run artifacts.

1. Introduction

Motivation. Long-running agents need to retain information across inputs while operating under a fixed inference budget. A natural proposal is to give the encoder itself a small, learned memory: either a bank of latent slots that summarizes an input (JM-S), or a state that is updated and carried from one chunk to the next (JM-R). Both are cheap, differentiable, and fit inside a single forward pass, unlike retrieval over an external store.

This paper reports what happened when we built both and tested them honestly. They did not work. We consider that outcome worth reporting because (a) the designs are intuitive and frequently proposed, (b) the failure is structural rather than a matter of insufficient tuning, and (c) the diagnostic path we followed produced several general-purpose lessons about evaluating memory mechanisms.

Contributions.

1. A controlled comparison of latent-slot pooling against a *parameter-matched* attention-pooling baseline, showing that the slots add no value on short-sentence tasks and that the failure is not due to undifferentiated slots (§3).
2. A four-stage diagnosis of the recurrent state pathway that separates output-path, update-rule, and gating pathologies, each measured directly (§5).
3. Five methodological findings, each of which had previously produced a wrong conclusion in our own pipeline, including a lexically-solvable-probe audit and a filter-induced selection bias on a baseline (§6).
4. A reproducible artifact: code, configs, per-run logs, and the frozen pre-registration under which the headline comparisons were made.

What we do not claim. We do not claim that neural memory is impossible, nor that the broader research direction is refuted. We tested one state pathway (8 slots, mean pooling, gated update) with one objective on one base checkpoint. §9 enumerates what remains untested.

2. Setup

2.1 Model and training

The encoder is a bidirectional Transformer with GQA, RoPE, RMSNorm, and a top-2 MoE FFN, 16 layers, $d = 512$, 291.9M parameters, sharing the Loftey 151{,}646-token BPE vocabulary, following the sentence-embedding recipe of [reimers2019]. All memory arms are initialized from the same Stage-1 MLM checkpoint and trained under identical data, optimizer, step count, and seed set.

2.2 Memory mechanisms

JM-S (latent memory pooling). Given hidden states $H \in \mathbb{R}^{T \times d}$ and N learnable queries $S_0 \in \mathbb{R}^{N \times d}$, one cross-attention read produces slot states

$$R = \mathrm{softmax}\left(\frac{(S_0 W_q)(H W_k)^\top}{\sqrt{d_h}}\right)(H W_v), \qquad S_1 = S_0 + R, \tag{1}$$

and the embedding is obtained by mean-pooling the slots followed by projection and l_2 normalization:

$$e = \mathrm{Norm}(W_p\, \mathrm{mean}_n(S_1)), \qquad \|e\|_2 = 1. \tag{2}$$

JM-R (recurrent memory). For a document split into chunks $x_1, \cdots, x_T$, the state is carried forward,

$$S^{(t)} = S^{(t-1)} + R^{(t)}, \qquad C^{(t)} = (1 - g^{(t)})\, S^{(t-1)} + g^{(t)} S^{(t)}, \tag{3}$$

where the optional gate is a learned sigmoid

$$g^{(t)} = \sigma(w^{\top}[S^{(t-1)}; S^{(t)}] + b), \tag{4}$$

and the per-chunk output uses the *newly read* slots:

$$e^{(t)} = \mathrm{Norm}(W_p \,\mathrm{mean}_n(S^{(t)})). \tag{5}$$

Equation (5) is the code as written; §5 shows its consequence.

2.3 Controls

Every comparison holds the encoder, data, optimizer, steps, and seeds fixed and varies only the pooling or state path. Table Table 1 defines the arms. The decision rule, pre-registered before any run:

$$\text{memory helps} \iff \Delta_{\texttt{reset}} > 0 \wedge \Delta_{\texttt{shuffled}} > 0 \wedge \mathrm{CI}_{95\%} \neq 0, \tag{6}$$

where $\Delta_{\texttt{reset}} = \mathrm{score}(\texttt{carried}) - \mathrm{score}(\texttt{reset})$ and likewise for $\texttt{shuffled}$, computed on paired examples and bootstrapped with the cluster unit set to the document.

Table 1. Control arms. `static`, `reset` and `concat` collapse to the same code path on non-recurrent arms (see §6), so only recurrent arms yield an informative $\texttt{carried} - \texttt{reset}$ contrast.

Arm	State	Purpose
`static/reset`	none	no-memory baseline
`carried`	true $S^{(t-1)}$	memory gain under test
`shuffled`	another example's state	rules out generic bias
`concat`	no state, both chunks	learnability ceiling
`prev_only`	last chunk only	one-step memory

3. JM-S: Latent Slots Do Not Beat Matched Pooling

3.1 Protocol

Five poolers share one encoder and one training run recipe: 66{,}799 NLI pairs, 1 epoch, 521 steps, AdamW at 3×10^{-4}, effective batch 128, maximum length 128, seeds $\{1, 2, 3\}$. Evaluation is on SICK-R (9{,}927 pairs) and SciFact retrieval

[thakur2021], both independent holdouts never used for checkpoint selection. The critical control is **B2mlp**: attention pooling followed by an MLP sized so its parameter count matches the slot module's (+1.05M, 292.92M total).

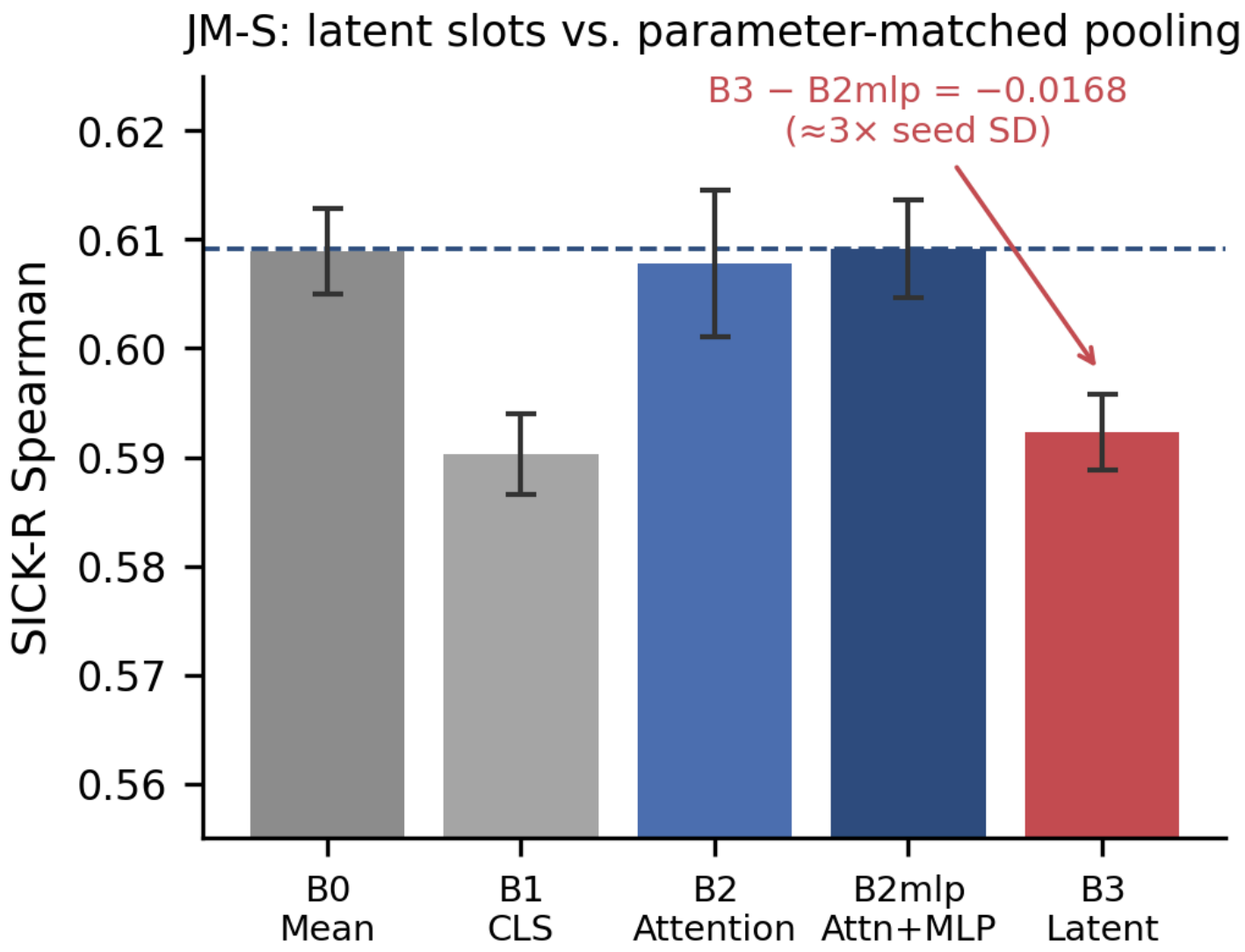


Figure 1. JM-S on SICK-R (mean ±SD over 3 seeds). The parameter-matched attention control B2mlp (0.6091) exceeds the latent-slot arm B3 (0.5923) by 0.0168, roughly three seed standard deviations, in the direction opposite to the success criterion.

Table 2. JM-S, SICK-R Spearman (3 seeds). The slot arm is last despite an equal parameter budget; B2mlp is the fair comparator.

ID	Pooling	SICK-R	Params
B0	mean	0.6089 ±0.0039	291.87M
B1	CLS	0.5903 ±0.0037	291.87M
B2	attention	0.6078 ±0.0067	291.87M
B2mlp	attention + MLP	0.6091 ±0.0045	292.92M
B3	**latent slots**	0.5923 ±0.0035	292.92M

3.2 Result

Table Table 2 and Figure Figure 1 give the outcome: $B3 - B2mlp = -0.0168$ with a pooled seed SD of 0.0057 ($\approx 3\sigma$). SciFact Recall@10 agrees in direction (B0 0.0737, B2mlp 0.0678, B3 0.0619). Two independent holdouts therefore point the same way.

3.3 The obvious repair does not work

Diagnostics showed the slots were not differentiated: pairwise slot-state cosine 0.8716 and attention-JS divergence 0.0437 (near-uniform, i.e. all slots read the same tokens). We therefore implemented the competitive assignment mechanism of Slot Attention [locatello2020] together with a diversity loss, producing six repair arms (C1–C6).

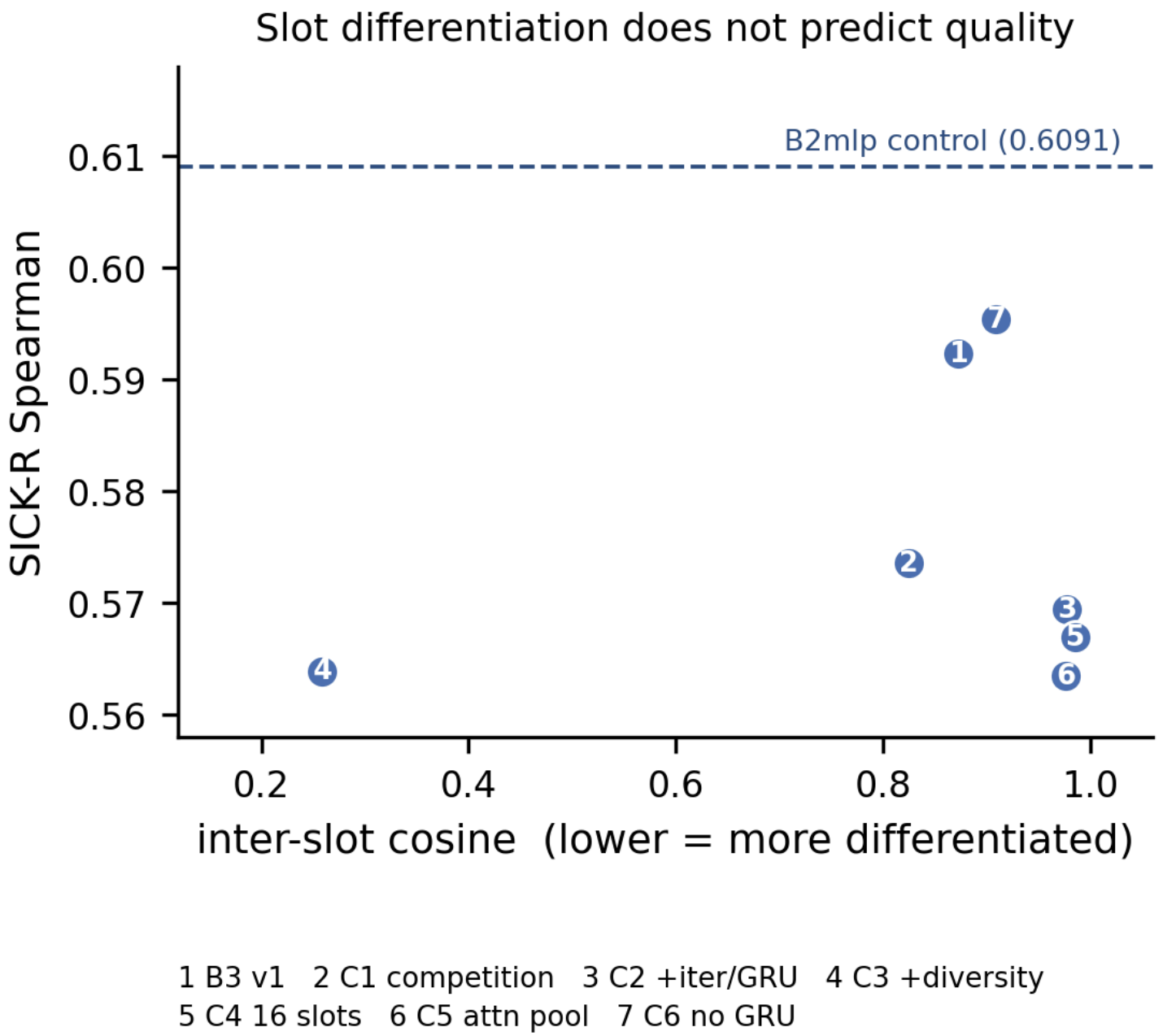


Figure 2. Slot differentiation versus quality. C3 (4) achieves the most differentiated slots (cosine 0.259) yet the second-worst score; the best- differentiated arms are not the best-performing ones. Differentiation is therefore not the binding constraint.

The repairs succeed at their stated goal and fail at the real one (Figure Figure 2). C1 raises attention JS from 0.044 to 0.089 and lowers slot cosine from 0.872 to 0.825, i.e. the diagnosed symptom is fixed, and the score drops to 0.5736. C3 drives slot cosine to **0.259** — by every structural measure the healthiest slot bank we produced — and scores 0.5639, second worst. Multi-iteration refinement makes slots collapse faster (C2 reaches cosine 0.977, and 0.998 before any training). Removing the GRU gives the best repair (C6, 0.5954), still below B2mlp.

We read this as falsifying the ``slots failed because they did not differentiate'' hypothesis. On short sentences (the SICK-R median is well under 128 tokens) attention pooling already covers every token, so additional slot capacity has little to absorb. The original motivation for slots — lossy compression of long inputs — is neither confirmed nor refuted by this task.

4. JM-R: A Recurrent State That Cannot Help

4.1 The one positive result, and why it is weak

On a synthetic two-chunk task we constructed so that neither lexical overlap nor single-chunk information can solve it (lexical ceiling 0.055–0.080 versus chance 0.156; an oracle parser confirms 100% derivability), the carried state does help:

Table 3. Synthetic cross-chunk task, mean ±SD over 3 seeds (chance 0.167). The fair `concat{}` baseline removes the state's apparent advantage entirely.

Arm	Accuracy
`reset`	0.161 ±0.003
`shuffled`	0.162 ±0.018
`concat{}` (unfair: embeddings summed)	0.461 ±0.006
`concat{}` (fair: joint encoding)	0.491 ±0.007
`carried`	0.492 ±0.012

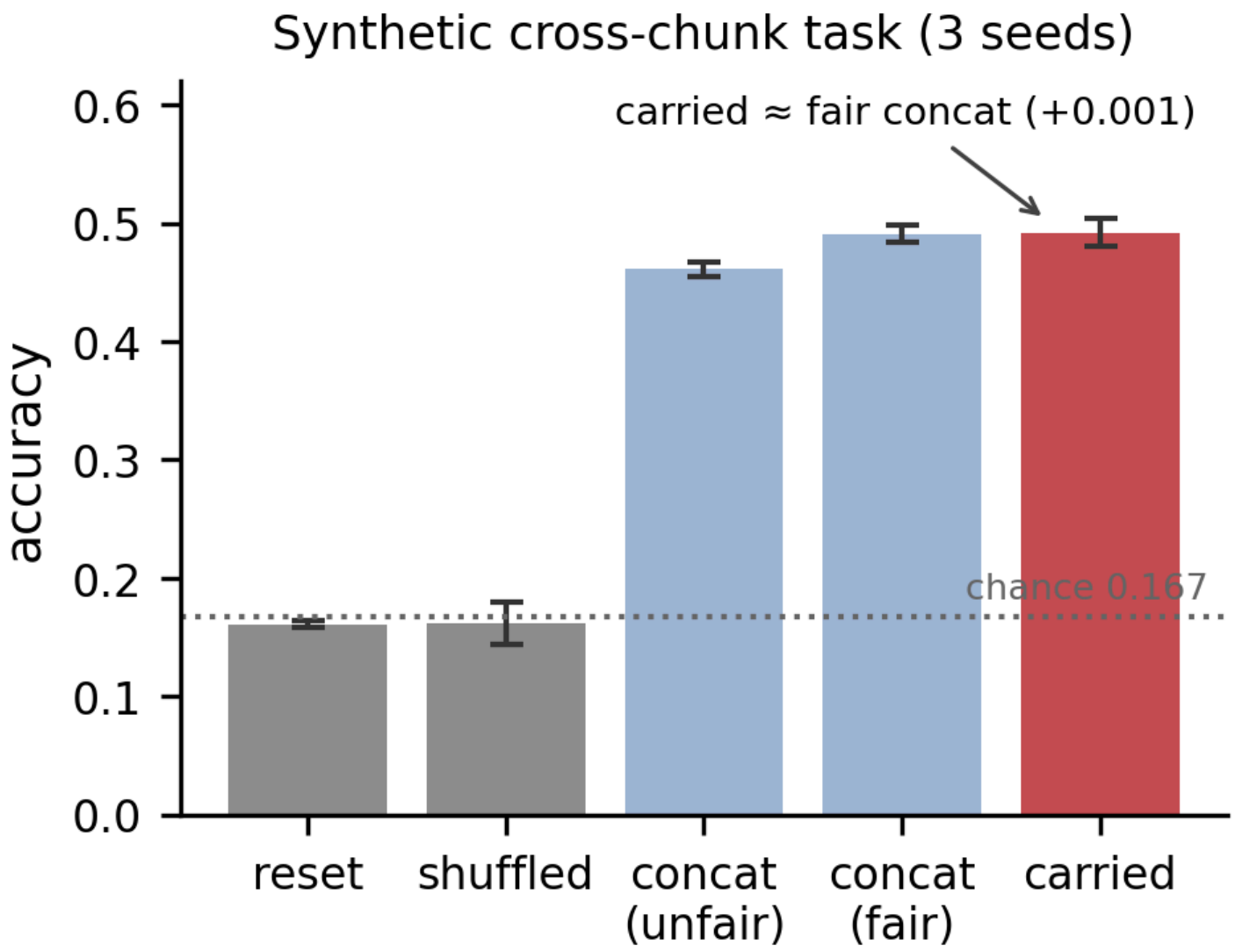


Figure 3. Synthetic task. `carried{}` clears the no-memory controls by a wide margin, but matches the fair `concat{}` baseline to within 0.001, so the state is a compression of information that joint encoding already exposes.

`carried{}` beats `reset{}` by 0.331 (3/3 seeds, CI excludes 0) and `shuffled{}` behaves exactly like `reset{}`, so the gain comes from correct content rather than a generic bias. But the fair `concat{}` baseline — both chunks encoded together, which is what an agent with sufficient context budget would do — scores 0.491, statistically indistinguishable from `carried{}` at 0.492 (paired differences + 0.002, + 0.010, − 0.009). Our earlier claim that state transfer beat concatenation was an artifact of an unfair baseline in which two independent embeddings were summed without a second forward pass.

4.2 Real text: below the pre-registered bar

On SciFact (three-stage abstract→claim classification, 228 test claims, document-component splits), `carried{}` attains the highest mean Macro-F1 (Table Table 4), but the pre-registered criterion (Eq. (6)) is not met: `carried{}` exceeds `shuffled{}` in only 2/3 seeds, and the accuracy bootstrap CI contains zero for 2 of 3 seeds.

Table 4. SciFact transfer, Macro-F1 (constant-predictor baseline 0.196). The mean ordering favors `carried{}`, but the `shuffled{}` contrast fails the pre-registered sign test in one seed.

Condition	Macro-F1
`carried`	0.3151 ± 0.0408
`concat`	0.2953 ± 0.0259
`shuffled`	0.2866 ± 0.0193
`reset`	0.2553 ± 0.0227

4.3 Nine pre-registered runs

We fixed three paper-level splits and three seeds per split (9 runs) with a frozen protocol specifying success criteria S1–S4 and forbidding any adjustment. The outcome is uniformly negative (Figure Figure 4): $\texttt{carried} - \texttt{static} = -0.0210$, 95% CI $[-0.0414, +0.0170]$, and **9/9 runs negative**. We report this despite the CI containing zero because the sign consistency is itself informative.

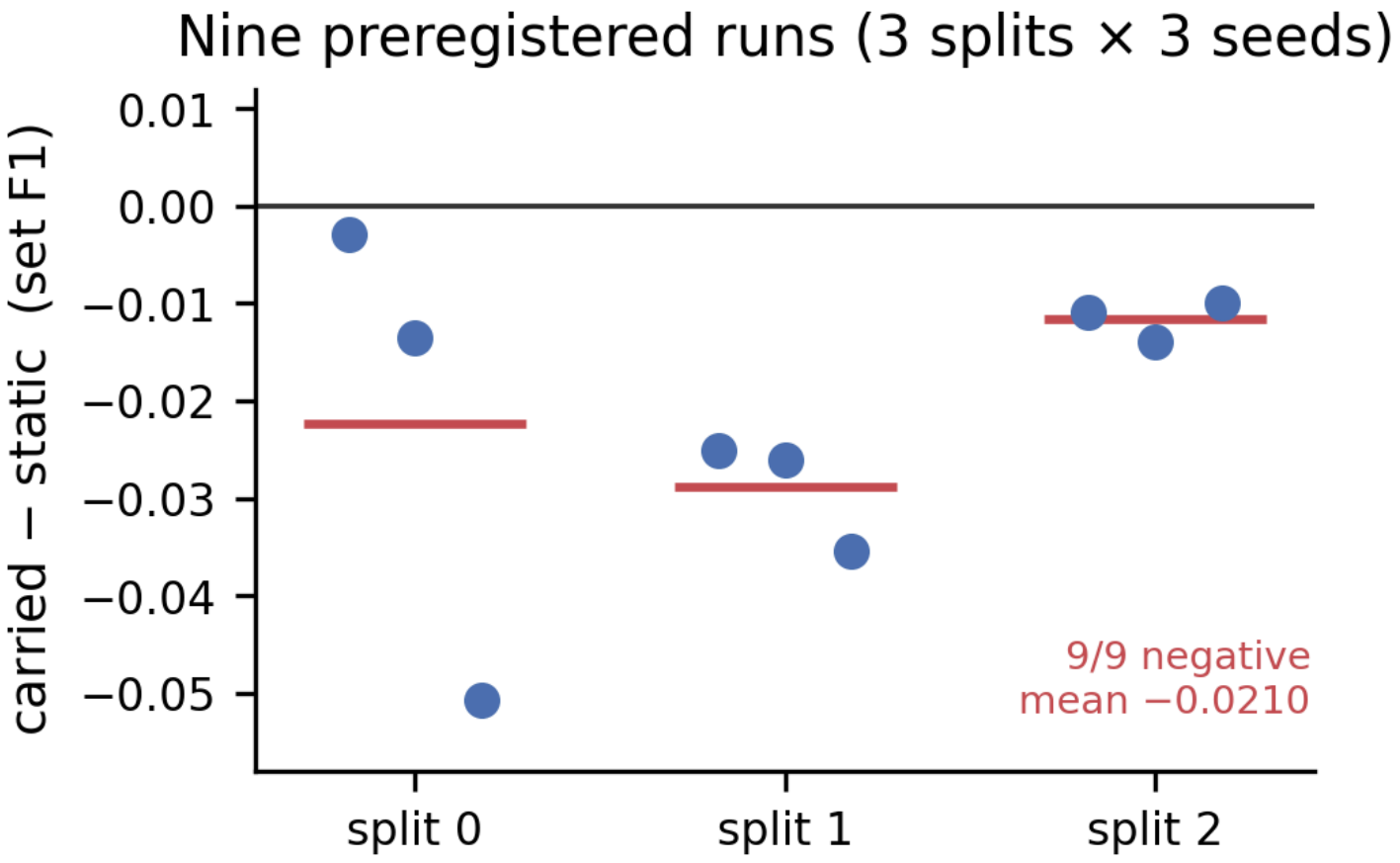


Figure 4. Nine pre-registered runs. Every paired difference is negative; the red bar is the per-split mean. Because training never left chance on this task (§9), these runs *cannot* adjudicate the mechanism — we report them as a bounded, protocol-respecting negative.

4.4 Long documents: the state collapses

On Qasper [dasigi2021] (12 chunks, 93% of test items have evidence at distance $\geqslant 8$), the state pathway collapses outright: loss stays at $\ln 12 = 2.479$, block representations become nearly identical (inter-block cosine 0.990, SD 0.0036), and similarity rises monotonically with position rather than reflecting content. State norm grows from 37.2 to 205.3 within 12 chunks.

5. Diagnosing the State Pathway

We localized the failure by measuring each stage in isolation (Figure Figure 5).

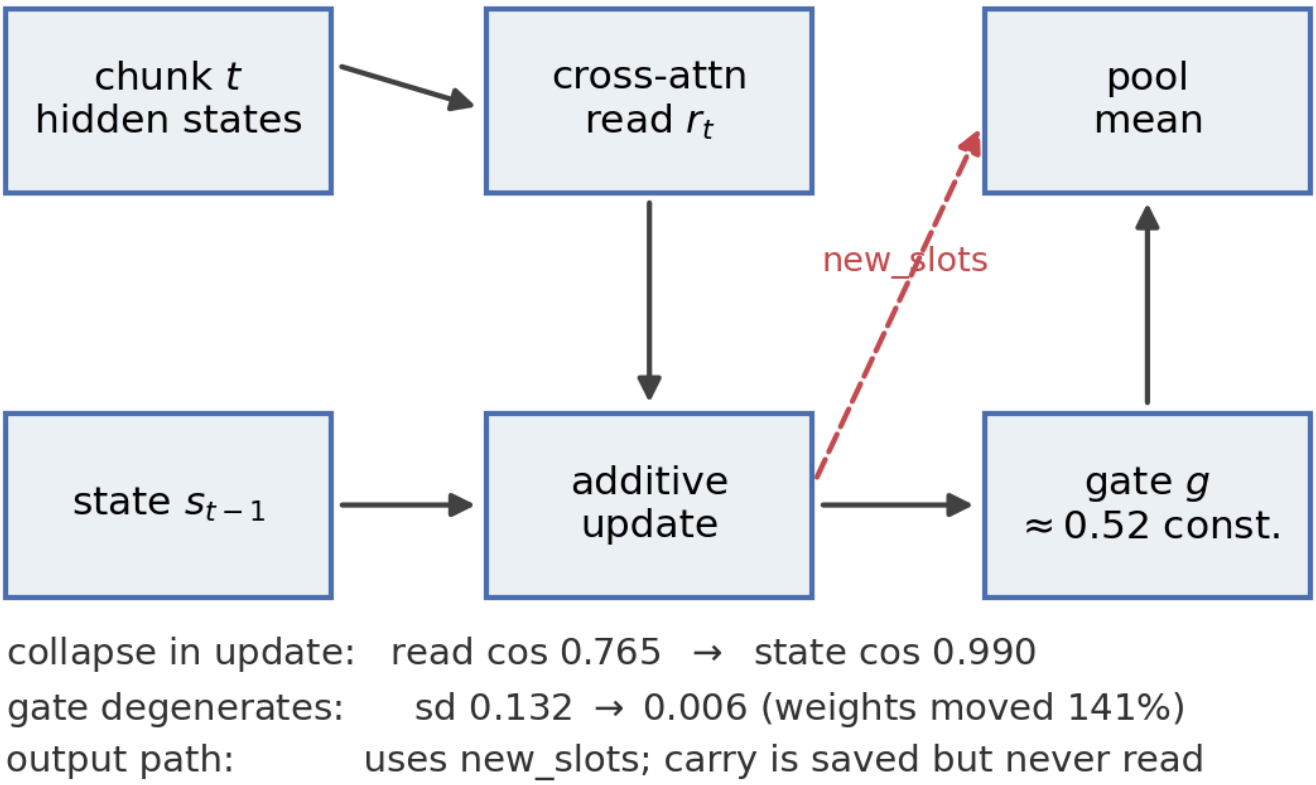


Figure 5. The audited pathway. The dashed red edge is the one used for the output; the carried state (via the gate) is stored but never read. Collapse occurs in the additive update, and the gate converges to a constant.

(i) The output never reads state. Expanding Eqs. (3)–(5) for a single read iteration,

$$e^{(t)} = \mathrm{Norm}(W_p[\mathrm{mean}(S^{(t-1)}) + \mathrm{mean}(R^{(t)})]), \tag{7}$$

so the previous state enters only as a *task-independent additive term* $\mathrm{mean}(S^{(t-1)})$, while all chunk-specific information comes from $R^{(t)}$, which depends on chunk t alone. We verified this empirically: $\mathrm{mean}(S^{(t)})$ and the model's output are *bit-*

identical ($\max|\Delta| = 0$), and linear probes recover value information equally well from either ($R^2 = 0.3847$ both), ruling out ``pooling discards slot structure'' (best single slot $R^2 = 0.4149$, flattened slots 0.3342, inter-slot cosine 0.935).

(ii) The update dilutes new chunks. Cross-attention read-outs *are* discriminative — counterfactually swapping two chunks changes $\|R\|$ by 0.0960 — but the additive update attenuates it 6.2×, to $\|\Delta S\| = 0.0155$. The relative change decays monotonically with chunk index (Figure Figure 6), from 0.318 at chunk 1 to 0.098 at chunk 11, while consecutive-state cosine rises to 0.9992. Gradients are not the problem (backbone gradient norm 1298); large-batch contrastive training would additionally require gradient caching [gao2021].

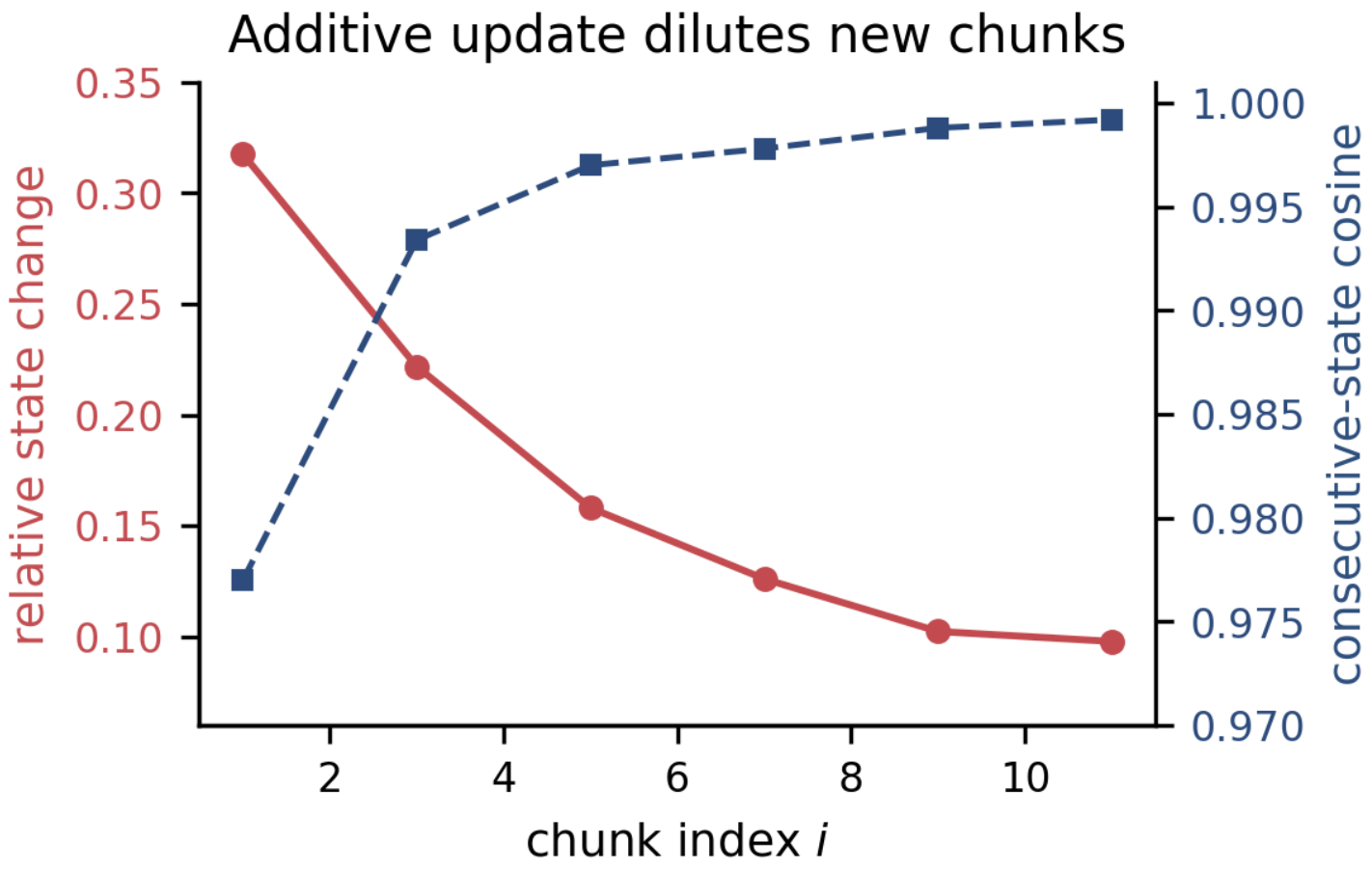


Figure 6. Additive update dilutes each new chunk. Relative state change (left axis) decays monotonically; consecutive-state cosine (right axis) approaches 1. Collapse originates in the update, not the read-out, whose inter-block cosine is 0.765.

(iii) The gate degenerates, and it is trained to. The gate (Eq. (4)) starts with SD 0.1316 under random inputs and ends after one epoch with SD 0.0063 — a 21× contraction to a near-constant $g \approx 0.52$. Its weights nevertheless move by 141% relative to initialization, so this is not an untrained module: the optimizer actively converges on a solution that ignores its input.

(iv) The gate is not the constraint. If the gate were the bottleneck, forcing a better value should help. Freezing the checkpoint and sweeping g over $\{0.10, 0.30, 0.50, 0.70, 0.90, 0.99\}$ yields accuracies 0.082, 0.115, 0.066, 0.098, 0.090, 0.074 — no monotonic trend, and every value inside chance ± 2SE (Figure Figure 7). Whatever is mixed, the state holds nothing useful for the task.

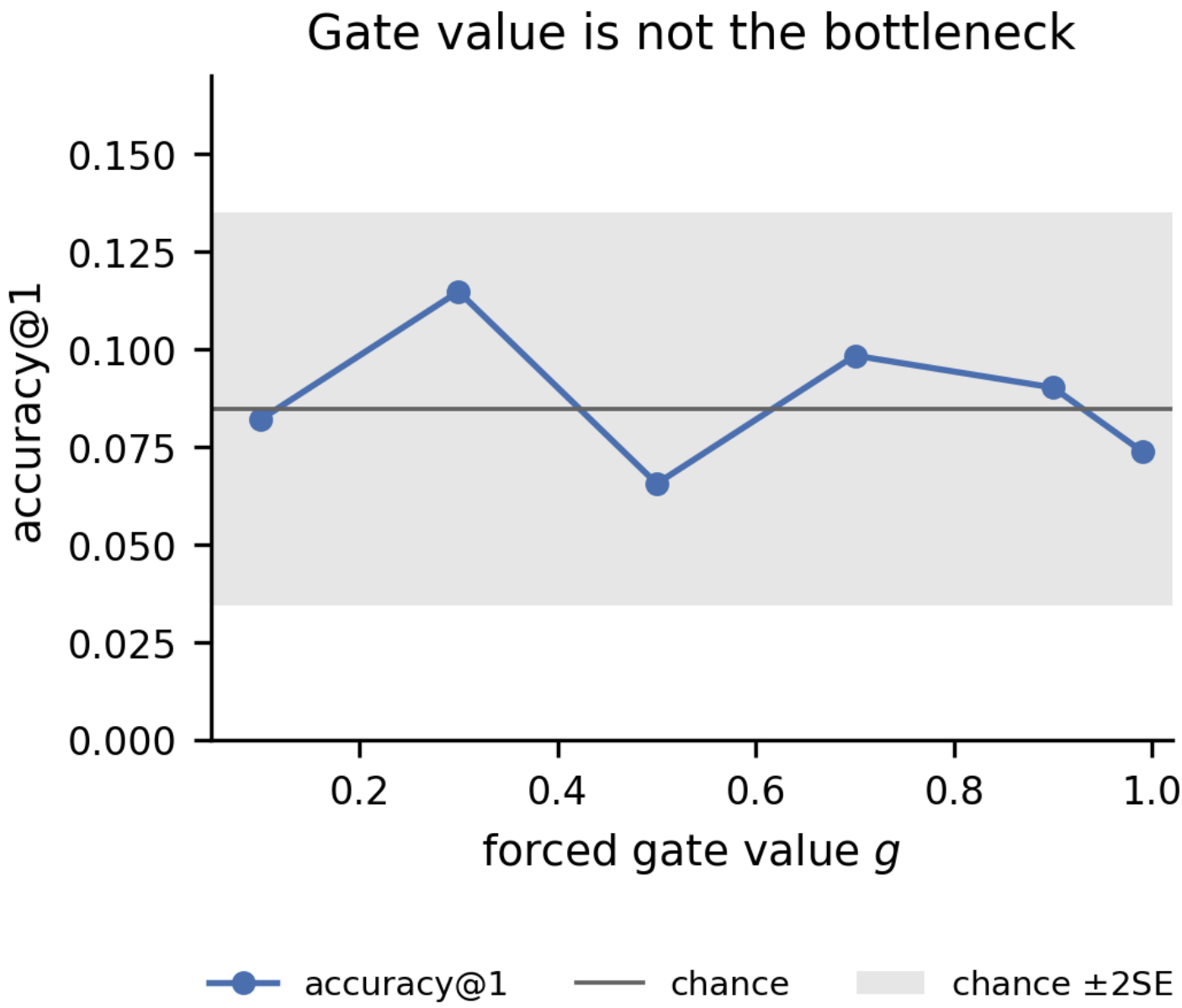


Figure 7. Forced gate sweep on a frozen checkpoint. All six values fall within chance ± 2SE (shaded), and there is no trend. Gate value is not the bottleneck.

Normalization fixes scale, not direction. Two bounded update variants (LayerNorm on $S + R$; gated bounded interpolation) bring state norm from 205.3 down to 64.0 and lift validation accuracy from 0.049 to 0.123, but inter-block cosine stays at 0.99 — they constrain magnitude without restoring discriminability. Both still fail `carried > reset`.

6. Methodological Findings

Each of the following had produced a wrong conclusion in our own pipeline before it was corrected.

M1: Probes can be solvable without memory. We added a lexical ceiling: the accuracy attainable by bag-of-words matching against the final chunk, with no model. Four of six probes are invalidated (Figure Figure 8): ceilings are **1.00** (conflict), **0.63** (isolation), **0.53** (order), and **0.23** (distractor), against best model scores of 1.00, 0.43, 0.60, and 0.63. A probe is uninformative whenever its ceiling meets or exceeds the model. The conflict probe's ceiling of 1.00 means it can

be solved perfectly without reading memory; one arm did score 1.000 on it, which without the ceiling would have been reported as a headline success. The probe required four redesigns, each refuted by a control, and we report it as a cautionary case rather than a usable measurement.

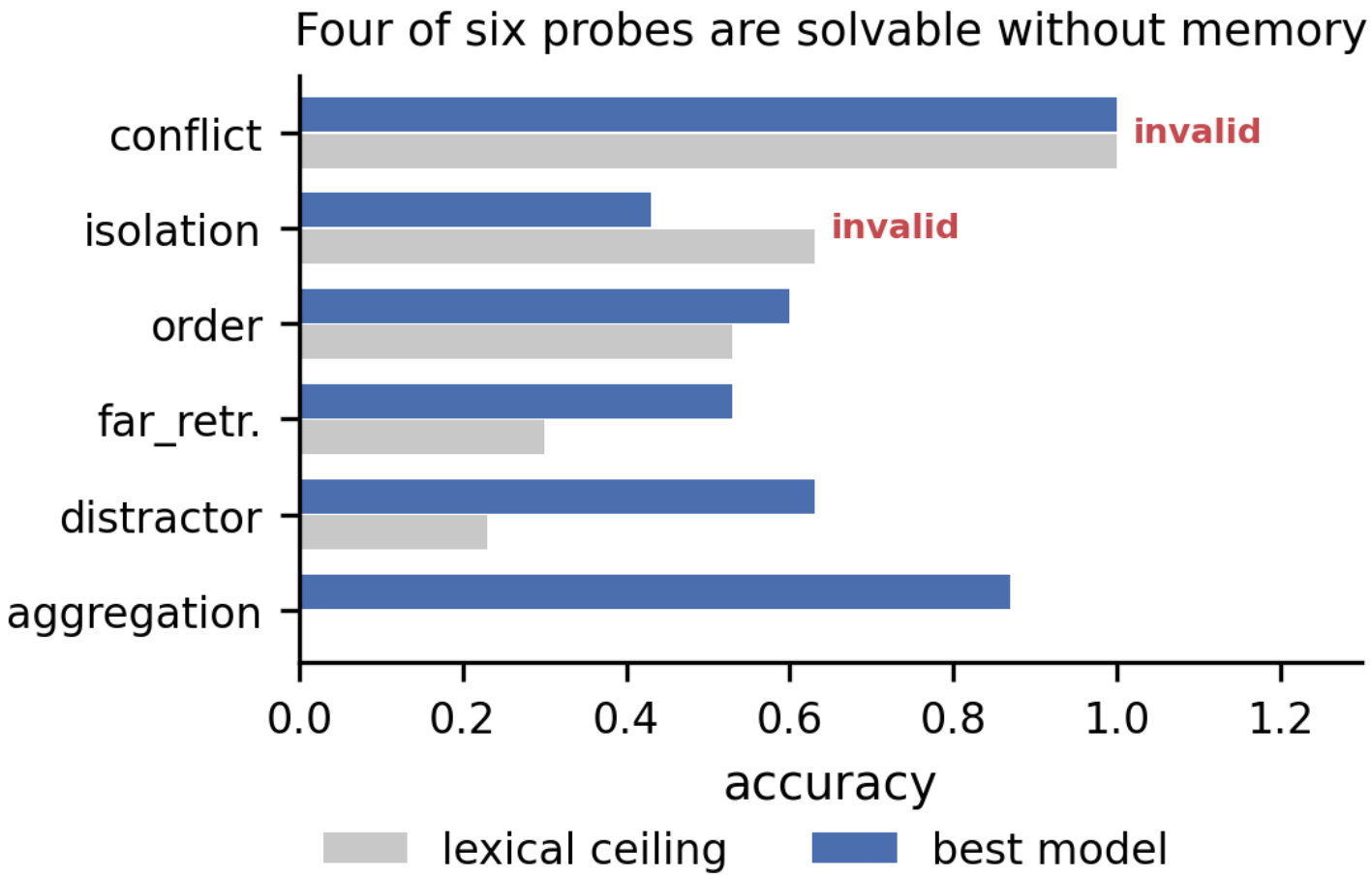


Figure 8. Probe validity. Grey bars are lexical-overlap ceilings; blue bars are the best model score. A probe is uninformative whenever the ceiling meets or exceeds the model. Only `aggregation` is fully clean.

M2: Filtering before splitting leaks baseline performance. An earlier construction dropped samples on which BM25's top-1 was an evidence block. The filter uses no labels but does use the baseline's behavior, turning the test set into ``the samples where BM25 failed''. Removing it raised the sample count from 592 to 881 and the BM25 test F1 from **0.140 to 0.302**, a 2.2× underestimate.

M3: A missing projection made a working model look broken. The evaluator returned pre-projection pooled vectors while the model's embedding is $\mathrm{project}\backslash\mathrm{tol}_2$. A model at 0.656 train accuracy measured 0.16 (below chance) and all arms became indistinguishable. When train performance far exceeds evaluation, suspect the evaluator first.

M4: Full fine-tuning destroys a pretrained signal. The Stage-1 encoder has block-level separation $\cos(\mathrm{gold}) - \cos(\mathrm{other}) = +0.021$ with no training (mean gold rank 3.42 of 6). Fine-tuning on 514 examples at 10^{-5}, 3×10^{-5}, and 10^{-4} reduces separation to $+0.0044$, $+0.0032$, and 0.0000 respectively, while gradients remain healthy (norm 14.3). A frozen backbone with a 524.8K-parameter head reaches 0.215 F1 in 170 seconds, beating 25 minutes of full fine-tuning. Small-data fine-tuning of large encoders is over-parameterized fine-tuning (Figure Figure 9).

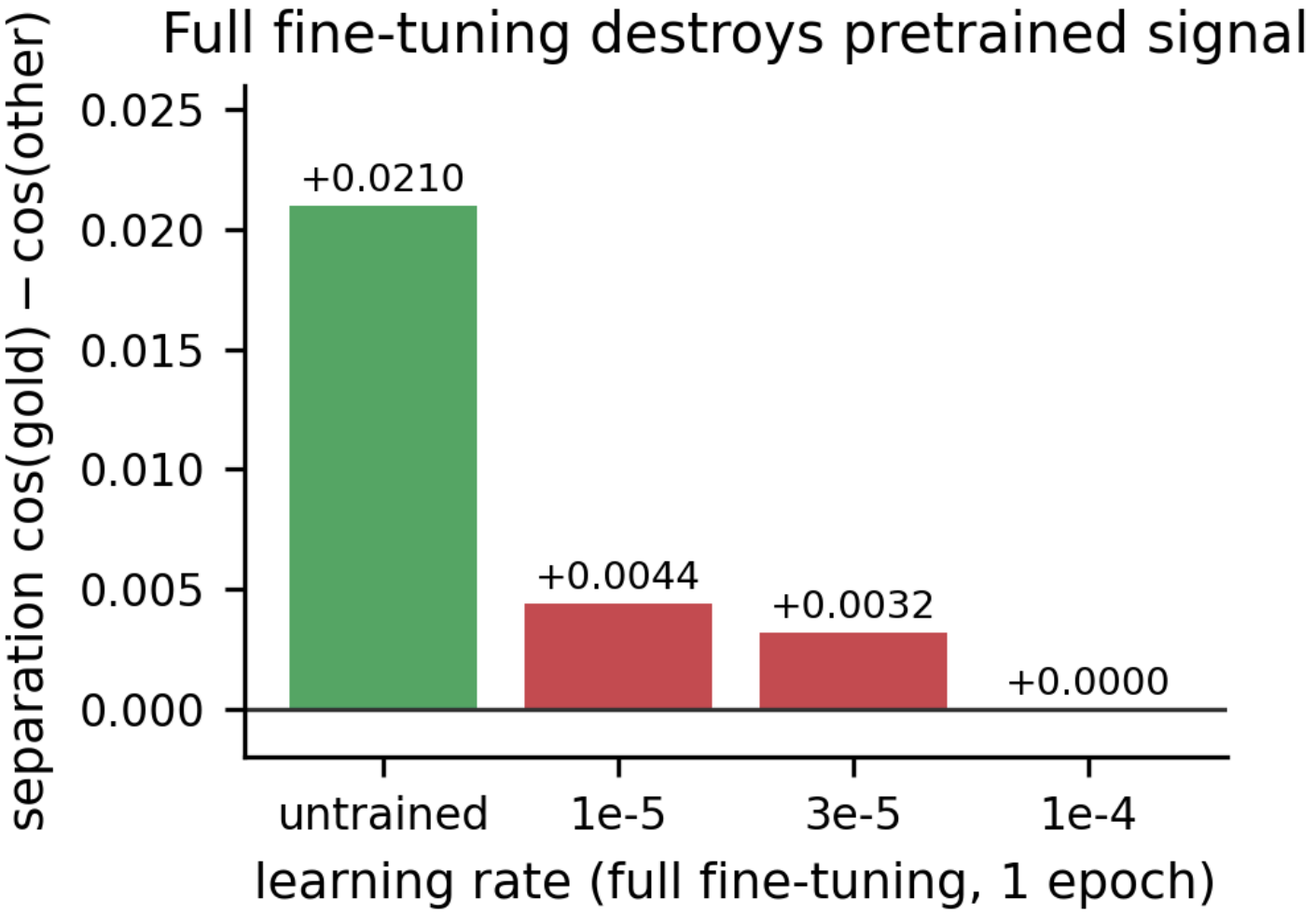


Figure 9. Block-level separation before and after one epoch of full fine-tuning. Every learning rate collapses the signal, confirming the effect is not a tuning artifact.

M5: Small-sample positives vanish. At $n = 300$, `carried{}` appeared to beat both `reset{}` and `concat{}` for the first time in the study ($+0.050$). At $n = 1500$ the effect collapsed to $+0.002$. Separately, a ``spread > 0.02'' criterion claimed the gate mattered; replaced by a monotonicity-and-± 2SE test, the conclusion reversed. Directional agreement across runs is not significance.

7. Established Facts

Measured directly, independent of interpretation: (1) the 350M encoder has block-level signal with zero training (+0.021 separation; mean gold rank 3.42); (2) full fine-tuning destroys it at every learning rate tested; (3) state is worse than the chunk's own vector as a block representation (-0.0785, 5/5 runs); (4) state is order-sensitive (`carried{}` above `shuffled{}` by $+0.0321$, 5/5), so it carries paper-level information; (5) **the encoder's semantic representation is weaker than BM25** on this task (0.28–0.31 versus 0.322, zero training); (6) slot differentiation does not predict quality; (7) state norm diverges ($37.2 \rightarrow 205.3$) and inter-block cosine approaches 1; (8) the gate degenerates to a constant despite 141% weight movement.

8. What This Does and Does Not Show

Supported. In this implementation, latent slots add nothing over a parameter-matched attention pooler, and a recurrent state is unsuitable as a per-chunk representation while carrying only information that joint encoding already provides.

Not supported. That neural memory is impossible; that the state channel can never work; that the designs are wrong *in principle*. We tested one pathway, one objective (NLI contrastive), one base checkpoint, and one task family.

Why we stopped. The pathology is structural (output path, update rule, gate), three independent measurements agree, longer sequences worsen rather than mitigate it, and every surface-level repair we tried (differentiation, normalization, gate forcing) left the mechanism-layer test failing. Continuing would require a different pathway, not more tuning.

The one untested question. Every experiment here asks whether state can *carry* information. None asks whether a model can *judge what to remember* — salience, importance, type. That requires no working state channel and is the only direction this work does not refute.

9. Limitations

Test sets are small (168–179 per split; SciFact 228; probes 30). Only three independent splits exist, with three seeds on one. Only one task family (block-level evidence localization) and one base checkpoint were tested. BM25 was not trained as the model was, so the comparison is not equal-condition. Most importantly, training used maximum length 128 while the architecture declares 8192: **long-sequence capability was never truly trained or tested**, so the compression regime that motivated latent slots remains unexamined. Multilinguality is untested.

10. Reproduction

Code, per-run configs (seed, hyperparameters), summaries (steps, throughput, peak memory), and evaluation JSON are released at https://modelscope.cn/models/LightNeko/Jaume together with the frozen pre-registration and figure-generation script. The full consolidated report is in `docs/memory-report.md` (Chinese) and `docs/memory-report.en.md` (English).

References

[locatello2020] F. Locatello, D. Weissenborn, T. Unterthiner, A. Mahendran, G. Heigold, J. Uszkoreit, A. Dosovitskiy, and M. Kipf.
Object-centric learning with slot attention.
In *NeurIPS*, 2020. arXiv:2006.15055.

[reimers2019] N. Reimers and I. Gurevych.
Sentence-BERT: Sentence embeddings using Siamese BERT-networks.
In *EMNLP*, 2019.

[gao2021] L. Gao, Y. Zhang, J. Han, and J. Callan.
Scaling deep contrastive learning batch size under memory limited setup.

arXiv:2101.06983, 2021.

[thakur2021] N. Thakur, N. Reimers, A. R{\"u}ckl{\'e}, A. Srivastava, and I. Gurevych.
BEIR: A heterogeneous benchmark for zero-shot evaluation of information retrieval models.
In *NeurIPS Datasets and Benchmarks*, 2021.

[dasigi2021] A. Dasigi, K. Lo, I. Beltagy, A. Cohan, N. A. Smith, and M. Gardner.
A dataset of information-seeking questions and answers anchored in research papers.
In *NAACL*, 2021.