Beyond Random Forgetting: Importance-Aware Memory Management for Self-Updatable LLMs

Akshat Bhandari **Ketaki Dabade** **Tushar Mittal**
{ab6247, kvd2112, tm3513}@columbia.edu
COMS 6998: Continual Learning and Memory Models, Columbia University

Abstract

Self-updatable language models must, by construction, forget but need they forget *uniformly*? MemoryLLM augments a frozen Llama-3-8B with a 1.67B-parameter latent memory pool and evicts tokens uniformly at random when new context arrives. We ask whether principled, importance-aware eviction can do better? We implement and evaluate three eviction policies grounded in continual-learning theory: **Attention** (retain by accumulated relevance), **Age** (protect recently injected tokens, operationalizing Complementary Learning Systems), and **Surprise** (evict tokens redundant with incoming knowledge) against the random baseline on knowledge-retention benchmarks over SQUAD v2 and NATURALQA using the pretrained MemoryLLM-8B checkpoint.

No strategy achieves statistically significant improvement over random at $N{=}100$ after Bonferroni correction. Age achieves the highest SQUAD AUC ($+0.28$; normalized $+0.014$), and a position-of-answer analysis confirms the gain is driven by mid-context examples rather than recency bias. On NATURALQA, random leads outright. We use *Layer-Jaccard*, a metric quantifying cross-layer agreement in evicted token sets, and show it explains this pattern: random's evictions fully decorrelate across the 32 layers ($J{\approx}0.01$), giving each fact $\sim$32 independent survival chances per injection step, while age's drops are nearly synchronized ($J{\approx}0.95$), concentrating eviction risk. The key obstacle for importance-aware eviction in multi-layer memory systems is therefore not identifying important tokens, but preserving the structural independence that makes random so competitive.

1 Goal and Motivation

The ability to incorporate new knowledge without retraining is a fundamental requirement for deployed language models, yet most LLMs treat their parameters as static after training.

MemoryLLM (8) addresses this by augmenting a frozen transformer with a fixed-size latent memory pool: new context is encoded directly into memory tokens at inference time, and when the pool is full, K tokens are evicted uniformly at random to make room. The authors prove this yields exponential retention decay-survival probability $(1{-}K/N)^T$ after T updates-and demonstrate stability across $\sim 10^6$ updates. The result is elegant, but the eviction rule is deliberately knowledge-agnostic.

This is a conspicuous gap. The continual learning literature exists precisely to answer the question of *which* information is worth protecting: EWC (2) protects parameters with high Fisher information, Synaptic Intelligence (11) tracks online parameter importance, and the Complementary Learning Systems framework (5) argues that biological memory consolidation itself is an importance-driven process. KV-cache eviction research (12; 1) has shown that attention-score-based retention sub-

stantially outperforms random deletion in the inference context. None of this machinery has been brought to bear on latent-space memory pools.

Research question. Do principled, importance-aware eviction policies retain knowledge better than random dropping in a self-updatable LLM? And if not, why is random so hard to beat?

The second half of that question turned out to be the more consequential one. Our answer, developed through a structural Layer-Jaccard analysis, is that random dropping's strength is an emergent architectural property of the multi-layer memory pool: per-layer independent eviction gives every stored fact $\sim$32 independent survival chances per injection step. Importance-aware strategies that synchronize drops across layers forfeit this property entirely. Understanding this mechanism is, we argue, more useful than any marginal AUC gain would have been.

2 Related Work

Self-updatable LLMs and latent-space memory: MemoryLLM (8) equips a transformer with a fixed-size memory pool $\theta_l \in \mathbb{R}^{N\times d}$ of N learned token embeddings. During generation, hidden states attend over all memory tokens; during self-update, new knowledge is encoded into the pool via the transformer itself. When the pool is full, K tokens are evicted uniformly at random. M+ (9) extends this with a two-tier architecture-a short-term memory (STM) identical to MemoryLLM's pool plus a CPU-resident long-term memory (LTM) that grows without bound. Tokens evicted from STM are merged into LTM rather than discarded, and a co-trained retriever selects relevant LTM entries during generation. Crucially, the STM eviction decision still uses random dropping-the exact point our work addresses. MemGPT (6) takes an OS-inspired virtual-memory view at the text level and does not learn eviction policies.

Continual learning and catastrophic forgetting: EWC (2) prevents catastrophic forgetting by penalizing changes to parameters important for prior tasks, using the Fisher Information Matrix as an importance proxy: $\mathcal{L}(\theta) = \mathcal{L}_B(\theta) + \sum_i \frac{\lambda}{2} F_i(\theta_i - \theta^*_{A,i})^2$. Synaptic Intelligence (11) computes parameter importance online during training. Both operate on model weights; in MemoryLLM the weights are frozen and only the memory pool changes. These methods motivate our importance-aware eviction designs, translating the core idea-protect what matters, release what does not-from parameter space to latent memory tokens.

Complementary Learning Systems: J. L. McClelland, B. L. McNaughton, and R. C. O'Reilly (5) proposed that the brain pairs a fast, pattern-separated hippocampus with a slow, integrative neocortex; episodes are buffered rapidly, then consolidated through repeated reinstatement. Our age-stratified strategy directly operationalizes this: the protection window acts as a hippocampal buffer for recent episodes, while inverse-age eviction pressure models neocortical consolidation dynamics.

KV-cache eviction: H2O (12) observes that attention scores follow a power-law distribution and retains "heavy-hitter" plus recent tokens. NACL (1) shows pure attention scoring suffers a recency bias and mitigates it via proxy-token scoring with randomized eviction. SnapKV (4) scores tokens via a recent observation window; StreamingLLM (10) retains attention-sink tokens plus a sliding window. Unlike KV-cache tokens-which are derived from one input and consumed in one generation-memory-pool tokens persist across thousands of updates. This persistence makes each eviction decision higher-stakes and, to our knowledge, principled latent-pool eviction has not previously been examined.

3 Approach

3.1 Model and Eviction Strategies

We build on the public `YuWangX/memoryllm-8b` checkpoint: Llama-3-8B augmented with a 1.67B-parameter memory pool (L=32 layers, 50 blocks $\times$ 256 tokens per layer, d=4096; total N=12,800 tokens/layer). We subclass the model as `MemoryLLMWithStrategies`, overriding only `drop_memory()`; the base architecture, weights, and trained LoRA adapters are untouched. All strategies share the same interface: given the current pool $\theta = \{m_1, \ldots, m_N\}$ with $m_i \in \mathbb{R}^d$ and K incoming tokens $\{c_1, \ldots, c_K\}$, return the $N-K$ indices to retain.

We evaluate four eviction policies. The first serves as our baseline; the remaining three operationalize distinct continual-learning signals:

- **Random.** $I_i \sim \mathcal{U}(0,1)$, resampled independently per layer per call. The original MemoryLLM policy (8); serves as our baseline.
- **Attention.** Accumulate an exponential moving average of attention received by each memory token: $I_i^{(t)} = \alpha I_i^{(t-1)} + (1-\alpha)\bar{a}_i^{(t)}$, where $\bar{a}_i^{(t)}$ is the mean attention across all heads and query positions at step t, with α=0.9. Tokens with the lowest accumulated attention are evicted first, transferring H2O's heavy-hitter intuition (12) from the KV cache to the latent memory pool.
- **Age.** Tokens injected within the last P=256 steps are protected (I_i=$+\infty$); all others receive $I_i = 1/(\tau_i + 1)$, so eviction pressure rises monotonically with token age. This directly operationalizes CLS (5): the protection window acts as a hippocampal buffer, the inverse-age pressure as consolidation dynamics. A per-layer noise term (ε=10^{-3}) breaks ties within age buckets to provide cross-layer diversity.
- **Surprise.** At each injection step, compute cosine similarity between each existing memory token and the incoming `delta_memory` representations:

$$s_i = \max_j \frac{m_i \cdot c_j}{\|m_i\| \, \|c_j\|}$$

  Importance is $I_i = 1 - s_i$, so the tokens most *redundant* with the arriving knowledge are evicted first. Computed on-the-fly with no persistent state, inspired by content-aware selection in SnapKV (4) and StreamingLLM (10).

Each signal captures a distinct dimension of importance: attention targets *relevance* to past queries, age targets *recency* of encoding, and surprise targets *redundancy* with incoming content.

3.2 Per-Layer Dropping: Matching the Training Regime

A critical design decision is the `drop_memory_per_layer` flag. When `True`, each of the 32 transformer layers independently selects which tokens to evict; when `False`, one decision is broadcast to all layers simultaneously. Auditing the upstream training configurations (`llama_30x256.yaml` and the OpenLLaMA configs) confirms that every shipped training run sets `drop_memory_per_layer: true`. We evaluate accordingly-this is the prerequisite for the Layer-Jaccard analysis in §4.3, which is trivially uninformative under shared dropping where cross-layer agreement is identically 1.0 by construction.

3.3 Evaluation Protocol and Sanity Gate

For each strategy–dataset pair we sample N=100 QA examples from SQUAD v2 (7) and NATURALQA (3), inject the gold context, then inject nuc=20 unrelated distractor contexts sequentially, generating an answer after each injection. Accuracy at step k is normalized substring match (lowercase, punctuation-stripped). The retention summary statistic is

$$\text{AUC} = \int_0^{20} \text{acc}(k)\, dk$$

computed via the trapezoid rule over 21 points; we also report AUC/20 $\in [0,1]$ for cross-dataset comparability. Statistical comparisons use paired permutation tests (10,000 iterations) with Bonferroni correction over three simultaneous comparisons (threshold p=0.017); bootstrap 95% confidence intervals (5,000 iterations) are reported on all AUC estimates.

Before every long evaluation run we execute a blocking sanity check comparing three memory conditions-*normal* (pretrained checkpoint state), *zeroed* (pool set to zero), and *scrambled* (pool positions randomly permuted)-with pass condition $\text{acc}_{\text{normal}} - \text{acc}_{\text{zeroed}} > 0.10$. This verifies that the memory pool contributes meaningfully to retention and that the LoRA decoder adapters have loaded correctly. All reported results are preceded by a passing sanity check ($\Delta \approx +0.6$ in every session).

4 Results

4.1 Memory Contributes Meaningfully to Retention

Table 1 reports sanity check results. The pretrained pool consistently outperforms the zeroed condition by $\Delta \approx +0.6$, well above the pass threshold. A notable secondary finding is that scrambled memory (0.60–0.63) performs almost as well as normal (0.57–0.67) at step 0: token *ordering* within the pool matters far less than the *presence* of trained weights, consistent with the LoRA decoder attending over pool content regardless of position. This hints at intra-block redundancy, which we return to in §4.3.

Table 1: Sanity check results (N=30, nuc=5). The normal–zeroed gap confirms the memory pool contributes substantively to retention.

Condition	Step-0	Step-5	AUC
Normal	0.567–0.667	-	2.45–2.50
Zeroed	0.00–0.03	-	0.00–0.05
Scrambled	0.600–0.633	-	2.23–2.62

4.2 Importance-Aware Strategies Do Not Significantly Outperform Random

Table 2 reports canonical per-layer AUCs. On SQUAD the ranking is age $>$ random $>$ surprise $>$ attention; on NATURALQA it is random $>$ age $\approx$ attention $\approx$ surprise. Age is the only strategy that does not lose to random on SQUAD ($+0.28$ AUC, normalized $+0.014$). No comparison survives Bonferroni correction: bootstrap 95% CIs of ± 0.7–1.4 render a 0.3–0.5 effect undetectable at N=100. All strategy differences are therefore descriptive rather than confirmatory.

Table 2: Canonical per-layer AUC (N=100, nuc=20). Normalized AUC/20: SQUAD random 0.409, age 0.423; NQ random 0.088. p from paired permutation tests (10k iters); none survive Bonferroni correction. †See §4.4.

	SQUAD		NQ	
Strategy	AUC	Δ	AUC	Δ
random	8.18	-	**1.77**	-
attention†	7.63	-0.55 (p=.07)	1.63	-0.14
age	**8.46**	$+0.28$ (ns)	1.71	-0.06
surprise	8.02	-0.17 (ns)	1.63	-0.14

The two datasets exhibit qualitatively different retention regimes. SQUAD retains $> 65\%$ of step-0 accuracy across all 20 distractor steps and never half-lives; NATURALQA half-lives within 6–15 steps with overall accuracy near floor (5–15%), making it the harder continual-retention test. The robust count-examples correct at both step 0 and step 20-is particularly low on NQ: age achieves only 1/100.

4.3 Per-Layer Independence Explains Random's Competitiveness

The most informative result is structural rather than numerical. For each (example, step) we log the evicted index set per layer and compute the mean pairwise *Layer-Jaccard*-the fraction of evicted tokens shared across all $32 \times 31/2$ layer pairs, averaged over all steps. Table 3 reports the results.

Table 3: Layer-Jaccard: mean cross-layer agreement in evicted token sets. Values for random, attention, and age are bit-identical across SQUAD and NATURALQA-a data-independence fingerprint exploited in §4.4.

Strategy	J (SQUAD)	J (NQ)
random	0.010	0.010
attention	0.010	0.010
surprise	0.618	0.602
age	0.946	0.946

Random's $J{\approx}0.01$ means the 32 layers evict nearly disjoint token sets: a stored fact must be simultaneously unlucky across all 32 independent eviction decisions to be lost, giving random $\sim$32 independent survival chances per fact per injection step. Age's $J{\approx}0.95$ means layers evict the same tokens in near-perfect lockstep-ages are synchronized by construction and the 10^{-3} tie-break noise barely diversifies them-so a single unlucky draw eliminates a fact everywhere at once. Surprise falls between ($J{\approx}0.61$) because per-layer `delta_memory` representations differ in activation space while encoding the same semantic content.

The shared-vs-per-layer ablation (Appendix A.1) supports the structural reading. Quoting the per-layer gain (per-layer minus shared, the negative of Table 4's Δ): random is the only strategy that gains on *both* datasets in per-layer mode ($+0.175$ SQUAD, $+0.215$ NATURALQA), and gains most on NATURALQA where every importance strategy degrades. Age, with the highest J, is essentially flat (-0.005 SQUAD; -0.165 NATURALQA). The monotonic relationship between J and per-layer sensitivity is consistent with the structural hypothesis (a controlled test is proposed in §5.2).

This also motivates a theoretical extension. The original exponential-decay result $(1{-}K/N)^T$ assumes a single shared pool; with L independently-dropping layers the effective per-fact survival against total loss is approximately $1{-}(K/N)^L$, substantially higher. Importance strategies that synchronize across layers implicitly collapse L independent pools into one.

A step-20 example-overlap analysis adds a complementary perspective: random's still-correct examples overlap with each importance strategy by only $J{\approx}0.51$ at step 20, whereas age and surprise agree on 0.68 of their retained examples. Random retains *structurally different knowledge*, suggesting a hybrid policy-a random reservoir combined with an importance-weighted budget-is more promising than replacing random entirely.

4.4 Attention Reduces to a Noise Baseline

Three observations converge on a single explanation for attention's underperformance. First, its Layer-Jaccard ($J{=}0.0101$) is numerically indistinguishable from random's ($J{=}0.0101$), far below the moderate value expected of a functioning relevance signal (cf. surprise at $J{=}0.618$). Second, attention's Jaccard is bit-identical across SQUAD and NATURALQA to 15 decimal places, matching data-independent strategies and contrasting with the genuinely data-dependent surprise (0.618 vs. 0.602). Third, it is the only strategy whose deficit approaches significance ($p{=}0.072$ uncorrected on SQUAD).

These observations are jointly inconsistent with a functioning EMA signal. The `update_attention_scores` method refreshes the EMA only when the in-sequence length exceeds the full pool (`shape[-1] > N = 12,800`). During injection, however, only the last `num_tokens`$= 256$ memory tokens are placed in-sequence. The guard never fires, the EMA remains all-zeros, and importance reduces to a 10^{-6} tie-break: pure noise. Attention operates as a second independent random baseline, not a relevance signal.

Critically, the Layer-Jaccard fingerprint detected this failure *from the data alone*, before any code inspection. This validates Layer-Jaccard as a diagnostic tool: a functioning importance signal produces dataset-varying Jaccard values; a dead signal does not.

4.5 Age's Advantage Is Mechanistic, Not an Artifact of Answer Position

A natural confound for age's SQUAD advantage is recency bias: if answer spans tend to appear late in source passages, age's protection window would trivially retain answer-bearing tokens longer. We

rule this out by partitioning examples into tertiles by the relative position of the gold answer span and measuring step-20 accuracy per bin (Figure 1).

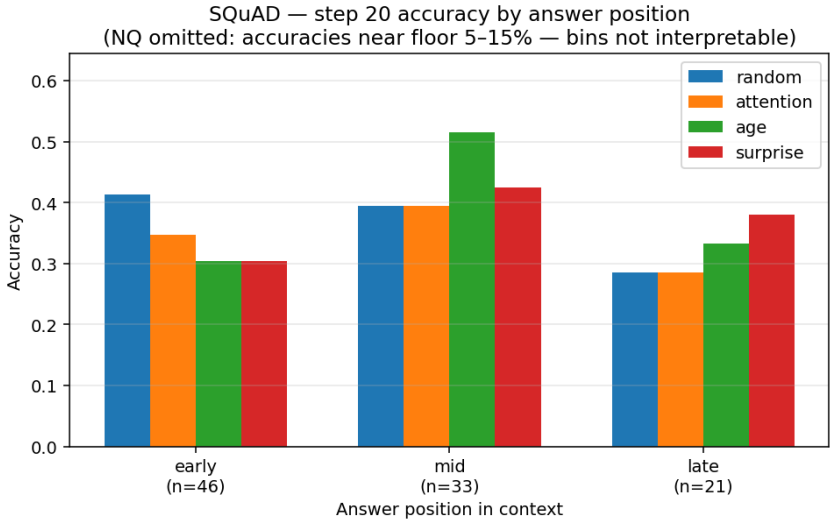

Figure 1: Step-20 accuracy by gold-answer position tertile on SQUAD. Age is *worst* on early answers (0.30 vs. random 0.41) and best on **mid**-position answers (0.52 vs. 0.39, Δ=+0.12). A recency-bias confound would predict a late-position advantage; the data show the opposite. NATURALQA bins are near-floor and uninterpretable.

Age performs *worst* on early answers and *best* on mid-position answers-the opposite of the recency-bias prediction. By step 20 the gold context has aged out of the protection window under every strategy, so the differentiator is which *other* tokens are retained. Age's inverse-age rule preserves the most recently injected distractor block, supplying lexical context the decoder uses to reconstruct mid-context facts-a form of indirect priming rather than direct retention.

4.6 Decay Profiles Reveal Strategy-Level Behavioral Differences

Aggregate AUCs compress per-example variation that illuminates distinct behavioral signatures across strategies.

Surprise exhibits a stability–accuracy trade-off. Surprise has the gentlest SQUAD decay curve (accuracy drop of 0.07 over 20 steps vs. 0.14–0.16 for others) but the lowest step-0 accuracy (0.43 vs. random 0.54). Evicting the most redundant tokens preserves pool diversity but may sacrifice tokens immediately useful for retrieval.

Importance signals exhibit an initialization transient. Surprise and attention both show a "shock minimum" at steps 2–3 followed by partial recovery, while random and age decline monotonically from step 0. This is consistent with importance signals requiring several injection steps to accumulate reliable statistics before their eviction decisions improve over noise.

5 Conclusion and Future Work

5.1 Conclusion

We investigated whether principled, importance-aware eviction policies can outperform uniform random dropping in a self-updatable LLM. Across three strategies and two knowledge-retention benchmarks, the answer is: not reliably, and not for the reasons one might expect.

The central finding is structural. Random dropping's competitiveness does not stem from any epistemic property of randomness; it is an emergent consequence of per-layer independent eviction in

a 32-layer memory pool. Each stored fact receives $\sim$32 independent survival chances per injection step-a property that importance-aware strategies forfeit when their signals synchronize across layers. Our cross-layer Jaccard analysis of evicted-token sets quantifies this directly, and its monotonic relationship with per-layer sensitivity is strong evidence for the structural hypothesis; the per-layer-noise ablation in §5.2 provides the controlled causal test.

Age-stratified dropping is the most promising of the three strategies, achieving the highest SQUAD AUC ($+0.28$ over random) with a mechanistic advantage that survives a position-of-answer analysis. That its drops are nearly fully synchronized across layers ($J{\approx}0.95$) suggests its true potential is yet to be measured: a version of age that preserves cross-layer diversity would combine the temporal structure of CLS with the structural independence that makes random hard to beat.

More broadly, this work establishes that the right question for importance-aware eviction in multi-layer memory systems is not *which tokens are important* but *how to preserve structural independence while being selective*. That reframing, and the use of cross-layer Jaccard as a diagnostic for eviction independence, are the primary contributions of this paper.

5.2 Future Work

Short-term: The most immediate priority is repairing the attention interface: the fix is localized to the in-sequence-length guard in `update_attention_scores`, which must accumulate the EMA over the 256 in-sequence memory tokens present at injection rather than gating on the full-pool length. A working attention signal would convert our most prominent null result into a genuine test of relevance-based eviction, and would enable direct comparison with NACL-style (1) randomized attention eviction—precisely the kind of hybrid that our structural analysis suggests is valuable.

A second priority is wiring the Fisher-inspired scorer into the evaluation pipeline: the scoring routine estimates token importance by measuring the KL divergence of the output distribution when each memory block is masked, $F_i = D_{\mathrm{KL}}(p_\theta(y|x,\theta) \,\|\, p_\theta(y|x,\theta \setminus m_i))$, transferring EWC-style importance (2) from parameter space to latent memory tokens. Connecting it to the eval loop would complete the four-strategy comparison originally scoped and provide the sensitivity-based counterpoint missing from the current results. Increasing evaluation to $N{=}300$ would narrow bootstrap confidence intervals by $\sqrt{3}$; given the current uncorrected $p{\approx}0.44$ for the age–random SQUAD comparison this is unlikely to reach significance on its own, but it is the cheapest step toward a powered, confirmatory rather than descriptive comparison.

Finally, a targeted ablation injecting increasing per-layer noise into the age signal—measuring AUC as J falls from 0.95 toward 0.01—would provide the cleanest causal test of the structural hypothesis: if performance rises monotonically with decreasing J, that is direct evidence that cross-layer independence, not age-based selection per se, drives the effect.

Long-term: The step-20 example-overlap analysis ($J{\approx}0.51$ between random and importance strategies) motivates a hybrid eviction policy: a random reservoir that preserves structural independence augmented by a fraction of importance-weighted selection. Moving evaluation to a continual domain stream-sequential injection across science, history, law, and medical domains-would test where importance-aware eviction should matter most; neither MemoryLLM (8) nor M+ (9) evaluates cross-domain interference. Finally, coupling the eviction policy with M+'s long-term memory store so that eviction becomes *demotion* rather than deletion would reframe the binary keep/evict decision as an EWC- (2) and CLS-style (5) consolidation schedule-the natural thesis-scale extension of this work.

References

[1] Y. Chen et al. NACL: A general and effective KV cache eviction framework for LLMs at inference time. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 7913–7926, 2024.

[2] J. Kirkpatrick et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526, 2017.

[3] T. Kwiatkowski et al. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466, 2019.

[4] Y. Li, H. Ye, and T. Cai. SnapKV: LLM knows what you are looking for before generation. *arXiv preprint arXiv:2404.14469*, 2024.

[5] J. L. McClelland, B. L. McNaughton, and R. C. O'Reilly. Why there are complementary learning systems in the hippocampus and neocortex: Insights from the successes and failures of connectionist models of learning and memory. *Psychological Review*, 102(3):419–457, 1995.

[6] C. Packer et al. MemGPT: Towards LLMs as operating systems. *arXiv preprint arXiv:2310.08560*, 2023.

[7] P. Rajpurkar et al. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2383–2392, 2016.

[8] Y. Wang et al. MemoryLLM: Towards self-updatable large language models. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, volume 235, pages 50453–50466, 2024.

[9] Y. Wang et al. M+: Extending MemoryLLM with scalable long-term memory. *arXiv preprint arXiv:2502.00592*, 2025.

[10] G. Xiao et al. Efficient streaming language models with attention sinks. In *Proceedings of the 12th International Conference on Learning Representations (ICLR)*, 2024.

[11] F. Zenke, B. Poole, and S. Ganguli. Continual learning through synaptic intelligence. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, pages 3987–3995, 2017.

[12] Z. Zhang et al. H_2O: Heavy-hitter oracle for efficient generative inference of large language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 36, 2023.

A Appendix

A.1 Shared-Drop vs. Per-Layer Contrast

To isolate the contribution of per-layer independence, Table 4 reports AUCs under shared dropping alongside the canonical per-layer numbers. Δ = shared − per-layer; negative means per-layer independence helped.

Table 4: Shared-drop AUCs alongside per-layer canonical results. Random is the only strategy consistently helped by per-layer independence on both datasets-the empirical core of the structural hypothesis (§4.3).

	SQUAD		NATURALQA	
Strategy	AUC	Δ	AUC	Δ
random	8.005	-0.175	1.550	-0.215
attention	7.675	$+0.045$	1.780	$+0.150$
age	8.465	$+0.005$	1.875	$+0.165$
surprise	7.745	-0.270	1.915	$+0.285$

Three patterns are notable. First, random is the only strategy helped on *both* datasets, consistent with the structural hypothesis. Second, the NATURALQA column is more diagnostic: every importance strategy degrades while random gains $+0.215$-the sharpest demonstration that synchronized drops are costly when retention is near floor. Third, age's Δ is near-zero on SQUAD ($+0.005$) and positive on NATURALQA ($+0.165$, i.e. per-layer mode did not help age), consistent with $J{\approx}0.95$: age synchronizes across layers regardless of dropping mode.

A.2 Retention Decay Curves

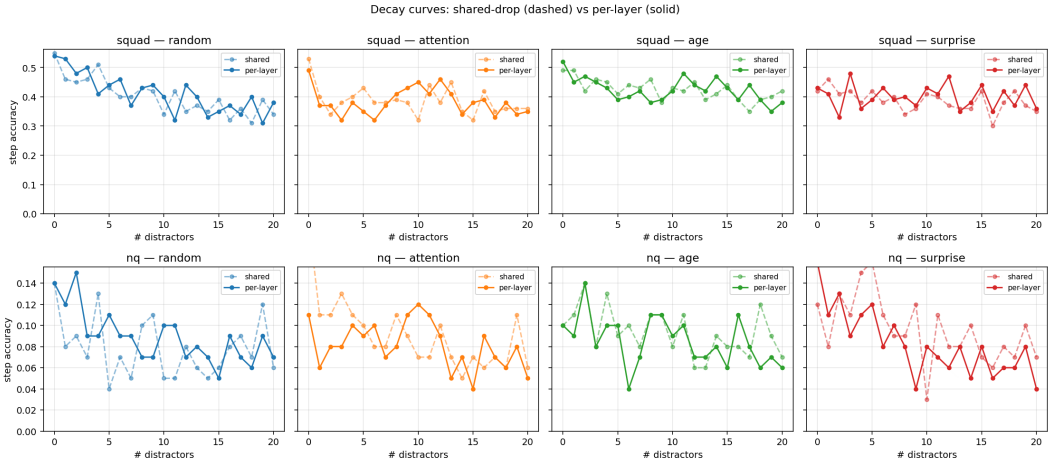

Figure 2: Shared-drop (dashed) vs. per-layer (solid) retention decay curves for every strategy–dataset combination. Per-layer mode lifts random the most on both datasets; importance strategies show mixed responses consistent with their Layer-Jaccard values.

Figure 2 shows the full 4×2 strategy$\times$dataset decay matrix under both dropping modes. The benefit of per-layer independence scales inversely with the strategy's Layer-Jaccard.

A.3 AUC Confidence Intervals: Shared vs. Per-Layer

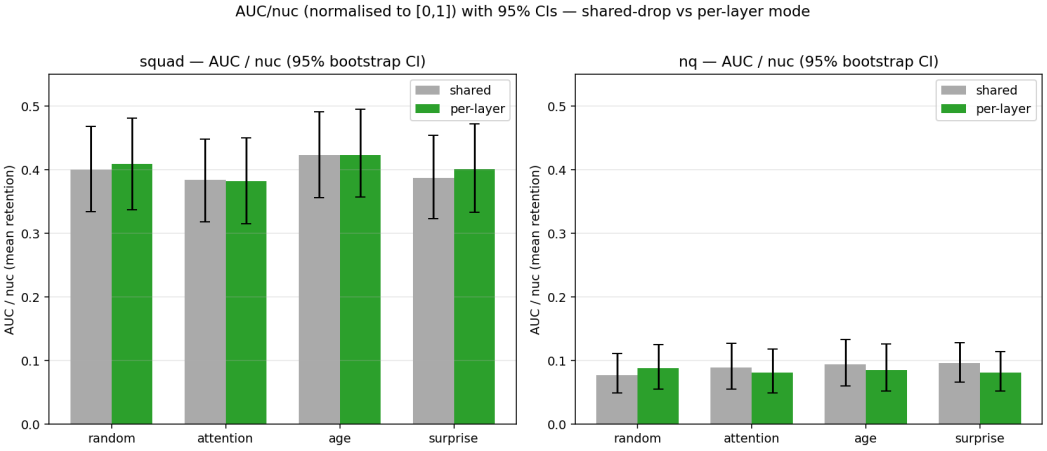

Figure 3: AUC/nuc with 95% bootstrap CIs, shared vs. per-layer mode side-by-side. Overlapping CIs across all strategy pairs make the $N=100$ underpowering explicit: all differences are descriptive, not confirmatory.

The complete overlap of confidence intervals at $N=100$ (CIs of ±0.7–1.4) motivates the $N=300$ confirmation run identified in §5.2.

A.4 Per-Strategy Example Breakdown

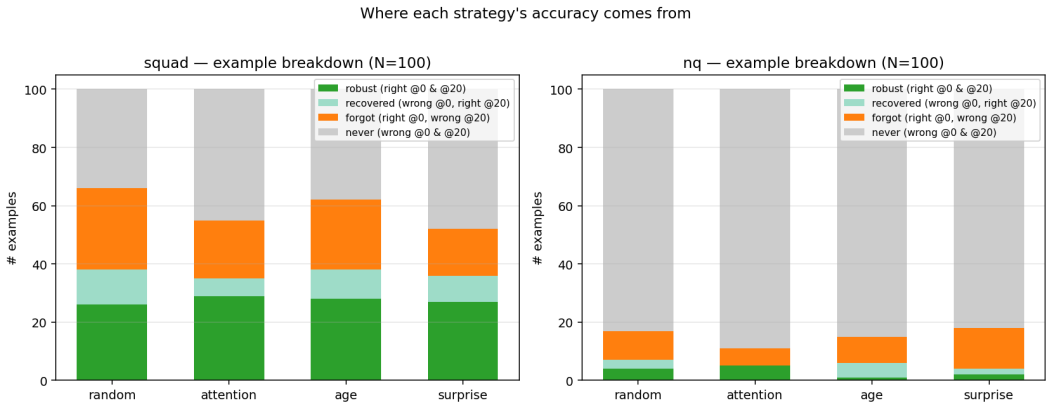

Figure 4: Per-strategy example breakdown: *robust* (correct at step 0 and step 20), *recovered* (wrong at step 0, correct at step 20), *forgot* (correct at step 0, wrong at step 20), *never* (wrong at both). Left: SQUAD; right: NATURALQA.

Robust counts confirm NQ's difficulty. On NATURALQA, robust counts are single-digit for every strategy (age: 1/100). AUC differences on NQ are driven almost entirely by early-step accuracy, not long-run retention.

Random retains structurally different examples. At step 20, random's correct examples overlap with each importance strategy's by only $J \approx 0.51$, whereas age and surprise agree on 0.68 of theirs. The *recovered* category is non-negligible for random ($\sim$12 on SQUAD), indicating random retains knowledge importance strategies systematically evict. This directly motivates the hybrid policy in §5.2.

Attention's stability is an artifact of noise. Attention's high robust count on SQUAD reflects the low variance of pure-noise eviction in aggregate, not systematic preservation of important tokens (see §4.4).