AMBER: AGE-AWARE MEMORY EVOLUTION FOR CONTINUAL ADAPTATION OF LARGE LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Continual fine-tuning adapts a large language model to a sequence of tasks. However, fitting a new task tends to overwrite knowledge from earlier tasks whose data is no longer available. Experience replay addresses this difficulty by retaining a small buffer of historical samples and training them along with the current task, but existing methods decide separately which samples enter the buffer and how often each is replayed. Because the buffer has a fixed capacity and is largely frozen once written, adaptation reduces to adjusting the replay frequency, and the stored content cannot keep pace as the gradient relationships between earlier and current tasks drift during training. To tackle this, we propose AMBER, a buffer management framework that manages the Age-aware Memory Buffer with Evolving Replay. The age is defined as the temporal distance between a sample and the current task. Using task temporal distance to encode each sample's historical position and its conflict with the current task, AMBER jointly regulates buffer writing, maintenance, and sampling for dynamic memory evolution. Writing draws candidates from a reservoir stream, a loss-quantile band, and slow-adapter generation. Maintenance replaces unprotected entries with generated traces at an age-dependent rate while protected anchors retain a real-data backbone. Sampling weights replay by age. On an eight-task sequence with Llama-2-7B, AMBER improves overall performance by 21.1% over the dual-adapter baseline and is the only compared method that does not degrade backward transfer. These results indicate that one interpretable age signal can govern buffer composition and replay scheduling together within a fixed storage budget, improving the stability and plasticity trade-off that continual learning must balance.

Keywords: Large Language Models, continual learning, parameter-efficient fine-tuning, experience replay, Catastrophic Forgetting

1 INTRODUCTION

Continual learning for large language models (LLMs) is the problem of updating a deployed model on a stream of tasks while preserving the abilities it already has. Under parameter-efficient fine-tuning, and particularly LoRA, naive sequential updates overwrite parameters that matter for earlier tasks, which produces catastrophic forgetting through parameter conflicts (Yu et al., 2020; Deng et al., 2025). Practical deployments rarely relearn a task once it has passed, and the full datasets of previous tasks is unavailable, so forgetting must be limited using only what the model retained from earlier stages.

Experience replay is the standard remedy, storing a small buffer of historical samples and training on them alongside the current task. Its effectiveness depends on two decisions, which samples are written into the buffer and how often each is replayed. Existing methods refine these decisions separately. On the writing side, reservoir sampling provides broad coverage without assessing informativeness (Vitter, 1985b), loss- or uncertainty-based selection targets samples near the decision boundary (Aljundi et al., 2019), representativeness filters and coreset construction improve coverage

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.

under tight capacity (Welling, 2009; Sener & Savarese, 2018), and generative approaches synthesize samples when the original data cannot be retained (Shin et al., 2017; Huang et al., 2024). On the scheduling side, methods adjust replay frequency using forgetting curves (Feng et al., 2026), surprise-based priorities (Hazard & Fountas, 2025), or recall strength (Lu & He, 2026).

This paper targets the limitation that these two decisions are decoupled and that the buffer's contents, as Figure 1 shows, are effectively frozen at write time. Because sample selection happens at insertion and subsequent scheduling only reweights the stored entries, the composition of a fixed-capacity buffer cannot respond to the way the model changes. As training proceeds, parameters move through the loss landscape and the gradient relationships between earlier and current tasks change, so samples that once caused little interference may become less compatible with later updates (Liang et al., 2025). Adjusting the sampling frequency then alters *which and how much* is replayed but not *what* is replayed. The same rigidity makes the removal of old samples abrupt by construction. Retaining them indefinitely consumes capacity that new-task adaptation needs, whereas discarding them abruptly removes the distributional support of consolidated knowledge and risks representational collapse.

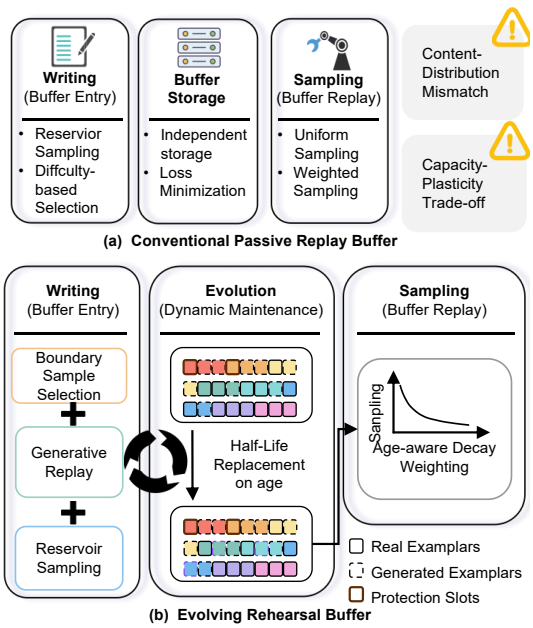


Figure 1: Comparison of the traditional passive replay buffer (a) and the evolving replay buffer (b). (a) Writing, maintenance, and sampling are typically handled without joint coordination, and buffer content is largely fixed after insertion, so adaptation mainly adjusts replay frequency. (b) AMBER treats the buffer as a dynamic system that evolves with sample age, jointly regulating writing, maintenance, and sampling.

AMBER is a continual learning method that defines sample age as the task temporal distance between a sample's source task and the current task. It uses this age to jointly regulate buffer writing, maintenance, and sampling, thereby enabling the memory system to evolve dynamically. Writing draws candidates from three complementary sources and records their source task, so that age is well defined for every entry. Maintenance treats retention as a probabilistic, age-dependent process. Half-life replacement lets unprotected historical entries yield space for replay traces generated by the slow adapter with a probability that grows with age, thereby avoiding both permanent retention and abrupt deletion, while a set of protected true anchors is resampled at every task boundary to keep the buffer from becoming entirely synthetic. Sampling uses exponentially decaying age weights, so recent sources dominate the replay batch while older entries still contribute. Because all three operations read the same age variable, the buffer evolves in content and in priority within a fixed storage budget.

Our contributions are as follows.

- We recast replay buffer management as a dynamic process driven by a single interpretable quantity, sample age, replacing the static storage abstraction in which content is fixed at write time (Section 3.2).
- We instantiate this view as a three-stage mechanism whose write, maintain, and sample operations all read the same age signal, so that buffer content and replay frequency evolve jointly under a fixed storage budget rather than being tuned independently (Sections 3.2.2–3.2.4).
- We show on TRACE-8, an eight-task continual learning sequence extended from the TRACE sequence with Llama-2-7B, that this coupling improves overall performance by 21.1% over the dual-adapter baseline and is the only configuration that does not degrade backward transfer, while an ablation isolates the contribution of each stage (Section 4).

2 Related Work

The Replay Paradigm in Continual Learning for Large Language Models Continual learning methods for LLMs span regularization (Kirkpatrick et al., 2017; Li & Hoiem, 2017), architectural (Wang et al., 2022; Razdaibiedina et al., 2023), and replay (Rolnick et al., 2019; Lopez-Paz & Ranzato, 2017; Chaudhry et al., 2019; Buzzega et al., 2020b) approaches. Among replay methods, I-LoRA (Ren et al., 2024) balances stability and plasticity by interpolating fast and slow LoRA adapters, and OPR (Chen et al., 2026) refreshes historical supervision by generating on-policy rollouts from the current model instead of replaying fixed responses. GeRe (Zhang et al., 2025) replaces replay data with pretraining Text: and SLAO (Qiao & Mahdavi, 2025) merges learned adapters to control memory usage. These methods refine how replay data is stored, generated, or supervised.

Write and Sampling Strategies for the Replay Buffer On the writing side, reservoir sampling (Vitter, 1985b), loss-based boundary selection (Aljundi et al., 2019), herding and coreset filtering (Welling, 2009; Sener & Savarese, 2018), and generative synthesis (Shin et al., 2017; Huang et al., 2024) select what to store, with GRASP (Harun et al., 2023) favoring a curriculum from prototypical to hard samples and BriMA (Zhou et al., 2026) balancing replay coverage across score quantiles. On the sampling side, replay frequency is adjusted through forgetting curves (Feng et al., 2026), surprise priorities (Hazard & Fountas, 2025), or recall strength (Lu & He, 2026), while AdaER (Li et al., 2023) and Adaptive Memory Replay (Smith et al., 2024) select samples dynamically. These methods treat the buffer as static storage whose contents, once written, are not continuously updated during training.

3 Method

3.1 Background

Problem definition. Consider an LLM undergoing continual instruction tuning on a task sequence $\mathcal{T} = \{T_1, T_2, \ldots, T_T\}$. At stage t, the model learns task T_t from its dataset $\mathcal{D}_t = \{(x_i, y_i)\}_{i=1}^{N_t}$ and a fixed-capacity replay buffer $\mathcal{B}$, while preserving performance on $T_1, \ldots, T_{t-1}$. The full datasets of previous tasks are unavailable.

LoRA tuning. LoRA adapts a pretrained model by freezing the original weights and learning a low-rank update (Hu et al., 2022). For a frozen weight matrix $W_0 \in \mathbb{R}^{d\times k}$, LoRA introduces a down-projection $A \in \mathbb{R}^{r\times k}$ and an up-projection $B \in \mathbb{R}^{d\times r}$ with rank $r \ll \min(d, k)$, and computes the layer output as $z = W_0 x + \frac{\alpha_{\text{LoRA}}}{r} BAx$.

Dual-adapter architecture. Dual-adapter architectures are widely used in LLM fine-tuning (Zhao et al., 2024; Yu et al., 2025; Qi et al., 2024). They separate plastic and stable parameter components to reduce interference between new and historical knowledge. The fast adapter θ_{f} receives gradients and learns new tasks, while the slow adapter θ_{s} provides the teacher representation for generated replay signals and the consistency constraint. It aggregates the fast adapter through an exponential moving average, abbreviated as EMA:

$$\theta_{\text{s}} \leftarrow \alpha \cdot \theta_{\text{s}} + (1-\alpha)\cdot \theta_{\text{f}}, \qquad \alpha = \min\left(1 - \tfrac{1}{\text{step}+1}, \alpha_{\max}\right), \tag{1}$$

where step is the training step and $\alpha_{\max}$ bounds the EMA coefficient. Early in training, the slow adapter closely follows the fast adapter. As α increases, its updates become more gradual.

Let θ denote the trainable adapter parameters, $f(x;\theta)$ the model's predictive distribution for input x, and $\mathcal{L}_{\text{CE}}$ the autoregressive cross-entropy loss. The ideal continual learning objective minimizes risk over all observed tasks:

$$\min_{\theta} \sum_{i=1}^{t} \mathbb{E}_{(x,y)\sim\mathcal{D}_i}\left[\mathcal{L}_{\text{CE}}(f(x;\theta), y)\right]. \tag{2}$$

Because previous task datasets are unavailable, Eq. equation 2 cannot be evaluated directly. Experience replay approximates historical supervision using a buffer of capacity M. In this setting, we address how to organize and update the buffer's contents under a fixed storage budget, and how to assign replay priorities to optimize the trade-off between adaptation to new tasks and retention of historical tasks.

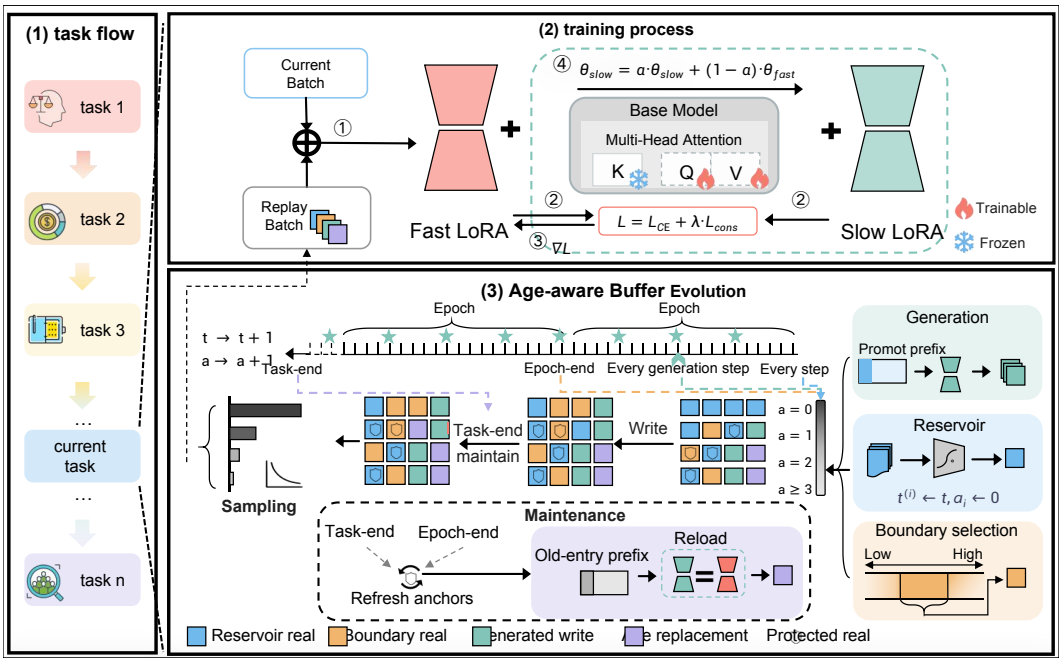


Figure 2: The pipeline of AMBER. First, tasks arrive sequentially and provide the current training batch (1). Next, for task t, the fast LoRA is updated on the current batch and a replay batch sampled with age-decayed weights, while the slow LoRA receives no gradients, is EMA-updated, and supplies the consistency target (2). Then, the age-aware buffer is written from reservoir, boundary-selection, and slow-adapter generation candidates at different stages (3). After each task, maintenance refreshes protected anchors and half-life replaces unprotected old entries. The loop repeats for the next task.

3.2 AMBER: Age-Aware Memory Evolution

3.2.1 Overview

AMBER is built on a dual-adapter architecture (Ren et al., 2024). It coordinates writing, maintenance, and sampling through sample age, so that buffer composition and replay priority are updated jointly rather than tuned independently. At the end of task t, the buffer is $\mathcal{B}_t = \{b_i = (s_i, y_i, \tau_i, z_i)\}_{i=1}^{N_t^B}$ with $N_t^B \le M$, where s_i is the input sequence, y_i is its supervision, τ_i is the source task index, and $z_i \in \{0, 1\}$ identifies a real sample ($z_i = 0$) or a generated replay trace ($z_i = 1$). Here, M is the buffer capacity and N_t^B is its current occupancy. The source age of entry i at task t is

$$a_i(t) = t - \tau_i, \tag{3}$$

this age is the shared reference for the three stages detailed in Sections 3.2.2–3.2.4.

3.2.2 Write-in: Reserving Candidates for Global Conflicts

An effective replay set needs both coverage and informative supervision, but $\mathcal{B}$ stores at most M entries. Reservoir sampling (Vitter, 1985a) supports broad coverage without explicitly assessing informativeness. Boundary selection identifies moderately difficult current-task samples but does not predict future interference. Generated replay reduces reliance on retained real data but lacks direct grounding in the original data distribution. AMBER combines these complementary sources rather than relying on any one of them.

Let $\mathcal{C}_t^{\text{reservoir}}$, $\mathcal{C}_t^{\text{select}}$, and $\mathcal{C}_t^{\text{gen}}$ denote the reservoir, boundary-selection, and generation candidate sets. Their union is

$$\mathcal{C}_t = \mathcal{C}_t^{\text{reservoir}} \uplus \mathcal{C}_t^{\text{select}} \uplus \mathcal{C}_t^{\text{gen}}, \tag{4}$$

where all newly admitted entries receive source task index t. At full capacity, insertion probabilistically replaces unprotected entries.

Reservoir Flow. The reservoir stream processes current-task samples at each training step. Admission follows reservoir sampling based on the number of observed real samples, excluding generated traces, and replacement targets unprotected slots. Anchor protection modifies standard reservoir sampling, so exact uniform coverage is not guaranteed. As more slots become protected, less capacity remains available for replacement.

Boundary Selection. Boundary selection retains samples of moderate difficulty for later replay. Following work that uses loss, uncertainty, or learning difficulty for sample selection (Aljundi et al., 2019), we define boundary candidates operationally as samples within a middle loss-quantile band rather than as explicit geometric decision-boundary samples. At the end of each epoch, we rank its samples by token-normalized loss and retain candidates in $[p_{\text{low}}, p_{\text{high}}]$:

$$\bar{L}_i = -\frac{1}{m_i} \sum_{u:\, y_{i,u} \neq -100} \log p_{\theta_f}\big(y_{i,u} \mid s_{i,<u}\big), \quad \mathcal{C}_t^{\text{select}} = \big\{ i \colon q_{p_{\text{low}}} \leq \bar{L}_i \leq q_{p_{\text{high}}} \big\}, \tag{5}$$

where m_i counts supervised tokens, q_p denotes the empirical loss quantile. The middle band avoids emphasizing either already well-fitted samples or extreme-loss samples that may contain noise or atypical patterns.

Slow Adapter Generation. Slow-adapter generation provides replay signals that reflect the model's accumulated behavior. Every g training steps, the slow adapter generates a continuation conditioned on a current-task prompt prefix (Robins, 1995; Shin et al., 2017; Sun et al., 2020). Prefixes follow a common template, but their actual content depends on the sample and truncation length. The generated buffer entry consists of the concatenated prompt prefix and generated continuation, with the prompt part assigned ignore labels such that supervision is applied only to the newly generated continuation.

The purpose is to retain a low-cost approximation of the slow adapter's conditional output behavior, not to reconstruct original historical examples. Its outputs provide a behavioral snapshot of accumulated knowledge (Tarvainen & Valpola, 2017; Li & Hoiem, 2017; Buzzega et al., 2020a). Generated traces can therefore provide useful retention signals even when they are incomplete or atypical, provided that they preserve relevant teacher behavior.

Replaying these traces alongside later task data encourages the fast adapter to retain earlier responses while acquiring new knowledge. The primary aim is to limit interference with existing capabilities rather than to improve new-task performance directly.

3.2.3 Maintenance: Half-Life Replacement and Protected Anchors

As the task sequence grows, retaining unchanged historical supervision can restrict content renewal within a fixed buffer. We therefore design a half-life-inspired age-based replacement mechanism, referred to as half-life replacement. At the end of each task, AMBER performs a maintenance scan. An unprotected historical entry is replaced by a slow-adapter-generated trace with probability:

$$P_{\text{replace}}(a_i) = 1 - 2^{-a_i(t)/T_{\text{half}}}, \tag{6}$$

Here, $\mathcal{P}_t$ denotes the protected set and T_{half} is the age scale. Each unprotected historical entry is replaced independently with this probability. This probability equals one half when the source age equals the age scale; it is a per-scan replacement probability, not a cumulative survival guarantee.

A replacement inherits the source task index and thus its age, preserving the buffer's temporal organization. Replacement renews supervision within a slot rather than increasing buffer capacity. Before each maintenance scan, the slow adapter reloads from the fast adapter checkpoint, and this parameter state also initializes the slow adapter for the next task. Replacement traces are therefore generated from currently learned knowledge.

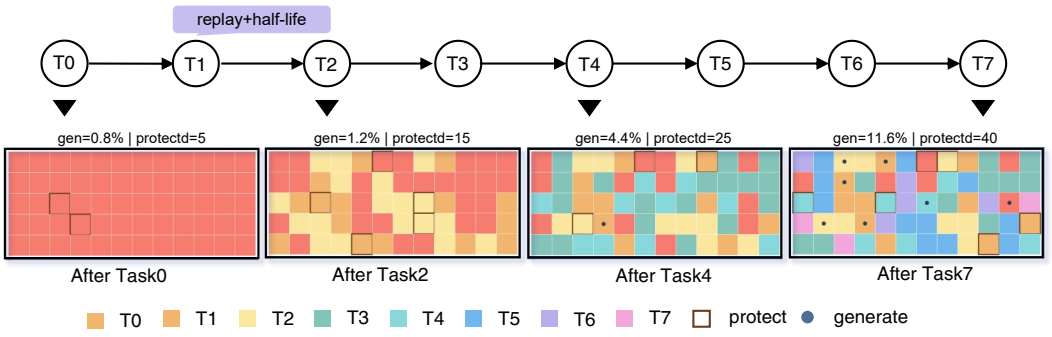


Figure 3: Buffer evolution across TRACE-8. Writing admits reservoir and boundary samples. Generation and half-life replacement add slow-adapter traces. Protection retains a real-data backbone.

Protected True Anchors. At the end of each epoch and before each maintenance scan, AMBER resamples up to k protected real anchors from each observed task:

$$\mathcal{P}_t = \bigcup_{j=1}^{t} \text{UniformSubset}\left(\{i : \tau_i = j,\ z_i = 0\}, \min\left(k, N_{t,j}^{\text{real}}\right)\right), \tag{7}$$

where $N_{t,j}^{\text{real}} = |\{i : \tau_i = j,\ z_i = 0\}|$ is the number of real entries from task j.Current anchors are exempt from reservoir overwrite and half-life replacement, which reduces distribution drift and degradation from repeated synthesis (Shumailov et al., 2024).

3.2.4 Sampling: Age-Aware Frequency Allocation

Inspired by the temporal decay of human memory, AMBER uses age-aware replay sampling (K., 2025; Feng et al., 2026; Kang et al., 2025). AMBER instead assigns decaying age weights exponentially:

$$\omega_i(t) = \frac{e^{-\lambda a_i(t)}}{\sum_{j \in \mathcal{B}_t} e^{-\lambda a_j(t)}}, \tag{8}$$
$$\mathcal{R}_t \sim \text{WeightedWithoutReplacement}\left(\mathcal{B}_t, \boldsymbol{\omega}(t), b_{\text{rep}}\right),$$

where $\mathcal{B}_t$ now denotes the active entries of the buffer, λ controls age decay, and b_{rep} is the replay batch size. The sampled batch $\mathcal{R}_t$ prioritizes recent sources while allowing older entries to contribute at lower frequency.

Algorithm 1 summarizes the complete procedure. Figure 3 shows how buffer composition evolves across the task sequence. The share of generated traces increases gradually, while the number of protected slots remains bounded. This pattern is consistent with the design of half-life replacement, which renews outdated real supervision gradually rather than removing it at a single cut-off, and with the protected anchors, which maintain a real-data backbone throughout training.

3.2.5 Joint Optimization

The fast adapter may suppress consolidated slow-adapter representations. We combine a supervised cross-entropy term and a one-sided consistency term. The cross-entropy is computed over the current batch and replayed samples, where only valid labels contribute:

$$\mathcal{L}_{\text{CE}} = \frac{1}{\sum_{b \in \mathcal{C}_{\text{batch}} \cup \mathcal{R}_t} m_b} \sum_{b \in \mathcal{C}_{\text{batch}} \cup \mathcal{R}_t} \sum_{u : y_{b,u} \neq -100} -\log p_{\theta_f}\left(y_{b,u} \mid s_{b,<u}\right), \tag{9}$$

where m_b denotes the number of valid label positions in sample b.

The consistency loss penalizes the fast adapter only when its activation falls below the slow adapter activation:

$$\mathcal{L}_{\text{cons}} = \frac{1}{|\mathcal{H}|\,|\mathcal{R}_t|\,S\,d} \sum_{l \in \mathcal{H}} \sum_{b \in \mathcal{R}_t} \sum_{u=1}^{S} \left\| \text{ReLU}\left(\text{sg}\left[\mathbf{h}_{s,b,u}^{l}\right] - \mathbf{h}_{f,b,u}^{l}\right) \right\|_2^2, \tag{10}$$

Table 1: Main results on Llama-2-7B using the eight-task heterogeneous sequence built on TRACE.

Method	AP_1	AP_2	AP_3	AP_4	AP_5	AP_6	AP_7	OP ↑	BWT ↑	Fgt ↓	FWT ↑
Non-replay methods											
Seq-Train†	0.418	0.410	0.312	0.275	0.219	0.137	0.256	0.210	−0.161	0.184	−0.025
EWC†	0.422	0.531	0.359	0.385	0.256	0.263	0.235	0.226	−0.185	0.212	−0.012
L2P†	0.438	0.395	0.240	0.264	0.228	0.154	0.247	0.257	−0.086	0.118	−0.023
PP†	0.372	0.421	0.266	0.284	0.253	0.259	0.263	0.224	−0.124	0.142	−0.047
Replay methods											
ER†	0.408	0.457	0.306	0.293	0.187	0.166	0.228	0.244	−0.148	0.169	−0.023
GEM†	0.388	0.465	0.284	0.274	0.127	0.177	0.285	0.215	−0.154	0.185	+0.000
A-GEM†	0.402	0.440	0.281	0.370	0.198	0.227	0.274	0.296	−0.084	0.095	−0.004
I-LoRA‡	0.505	0.600	0.502	0.527	0.469	0.464	0.390	0.384	−0.084	0.104	+0.062
SSR‡	0.503	0.597	0.509	0.502	0.499	0.487	0.424	0.377	−0.070	0.094	+0.071
FOREVER‡	0.496	0.583	0.498	0.499	0.459	0.445	0.444	<u>0.408</u>	<u>−0.022</u>	**0.030**	+0.069
AMBER (ours)	**0.506**	**0.655**	**0.523**	**0.556**	**0.547**	**0.526**	**0.476**	**0.465**	**+0.002**	<u>0.032</u>	**+0.130**

Zero-shot scores and † results are from (Ren et al., 2024). ‡ Rerun using the official code under a unified protocol. OP = AP_8. FWT is relative to the zero-shot baseline. Bold and underline mark the best and second-best values. Zero-shot per-task scores are provided in Table 3.

where $\mathrm{sg}(\cdot)$ stops gradients through the slow adapter, S and d denote sequence length and hidden dimension, and $\mathcal{H}$ is the set of transformer hidden layers used for consistency. This asymmetric penalty protects consolidated representations while allowing new skill activations to rise freely. The final objective is $\mathcal{L} = \mathcal{L}_{\mathrm{CE}} + \lambda_{\mathrm{cons}} \mathcal{L}_{\mathrm{cons}}$.

4 Experiments

4.1 Experimental Setup

Continual Learning Benchmark. We use the same data splits, task sequence, LoRA configuration, and evaluation script across methods. We build on the TRACE continual learning benchmark (Wang et al., 2023) and extend its task sequence with medical and legal tasks. The eight-task sequence comprises C-STANCE (Zhao et al., 2023), FOMC (Wang et al., 2023; Shah et al., 2023), MeetingBank (Hu et al., 2023), ScienceQA (Lu et al., 2022), NumGLUE-cm (Mishra et al., 2022), 20Minuten (Rios et al., 2021), MedMCQA (Pal et al., 2022), and JEC-QA (Zhong et al., 2020) in that order, which we refer to as TRACE-8. Task formats and per-task evaluation metrics are listed in Appendix B.1.

Comparison Methods. We consider three groups of baselines. Zero-shot provides a reference without task-specific training. Non-replay methods include Seq-Train, EWC (Kirkpatrick et al., 2017), L2P (Wang et al., 2022), and PP (Razdaibiedina et al., 2023), which assess the benefit of replay. Replay methods include ER (Rolnick et al., 2019), GEM (Lopez-Paz & Ranzato, 2017), A-GEM (Chaudhry et al., 2019), I-LoRA (Ren et al., 2024), SSR (Huang et al., 2024), and FOREVER (Feng et al., 2026). Implementation details are provided in Appendix B.2.

Evaluation Metrics. We report average performance, overall performance, backward transfer, forgetting, and forward transfer. Their formal definitions are provided in Appendix A.1. All reported metrics are single-run point estimates.

4.2 MAIN RESULTS

Traditional continual learning methods struggle to balance stability and plasticity. We observe from Table 1 that the non-replay methods suppress parameter drift through sequential fine-tuning, regularization, or prompt learning, yet they all show negative backward transfer and high forgetting, and negative forward transfer. This pattern indicates that constraining parameter drift alone is insufficient for long heterogeneous sequences. Conventional replay methods retain historical samples, but their balance remains limited. A-GEM achieves the lowest forgetting and the least negative backward transfer among ER, GEM, and A-GEM, while its forward transfer stays close to zero. ER and GEM forget more and transfer less. None reaches positive backward transfer, suggesting that static sample replay or gradient constraints alone cannot retain old knowledge while still facilitating new-task learning.

AMBER achieves a better balance between stability and plasticity through age-aware evolution of buffer contents and reaches the best overall performance. I-LoRA shares the same dual-adapter architecture and buffer capacity but relies on static reservoir writing and random sampling, whereas AMBER dynamically evolves buffer contents across the task sequence. This difference yields a 21.1% improvement in overall performance over I-LoRA, pointing to an independent contribution from the evolutionary buffer design. FOREVER optimizes replay frequency using an exponential decay derived from a forgetting curve and reaches a competitive overall performance, but its content composition remains fixed. AMBER improves on FOREVER by 14.0%, indicating that frequency scheduling and content evolution are complementary and that frequency scheduling alone cannot fully address distribution mismatch in long sequences. The generative replay method SSR shows that self-synthesized replay can benefit new-task learning. Compared with SSR, AMBER lets the buffer respond continually to memory decay and distribution drift, further improving new-task adaptation. AMBER is also the only method with non-negative backward transfer. This is consistent with half-life replacement, which under a fixed buffer budget gradually replaces a portion of historical real-sample supervision with replay signals generated by the current slow adapter and conditioned on historical content, adjusting the influence of historical supervision without abruptly discarding it. AMBER's forgetting score is slightly higher than FOREVER's but substantially lower than I-LoRA's. Together with its higher overall performance, these results indicate a favorable retention–adaptation trade-off rather than the elimination of forgetting.

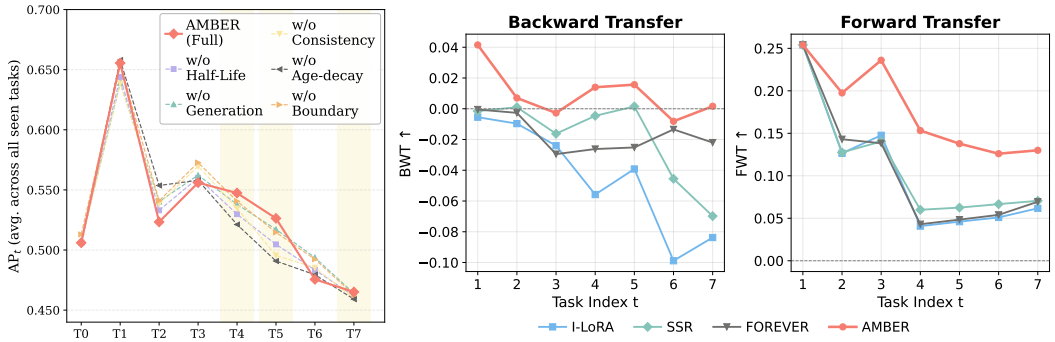
4.3 ABLATION STUDY

To validate the contribution of each design choice in AMBER, we disable one component at a time while keeping the remaining mechanisms unchanged, with results summarized in Table 2. First, removing boundary selection is the only ablation that makes BWT negative, suggesting that boundary selection helps maintain backward transfer rather than merely improving overall performance. Second, removing half-life replacement lowers OP and FWT while also reducing Fgt. This trade-off indicates that half-life replacement mainly supports adaptation by renewing historical supervision. Retaining old real samples without replacement preserves past-task performance more closely but appears to constrain new-task adaptation. Third, removing generative replay or age-decay weighted sampling also degrades OP, with age-decay weighting causing one of the largest drops. This confirms that age is central to both content evolution and replay frequency allocation. Finally, removing the consistency loss leads to a clear OP drop, linking fast–slow adapter alignment to new-task adaptation. Although every ablation reduces Fgt, none matches the full model in OP. Overall, the full design favors overall performance over minimizing forgetting alone.

Table 2: Ablation study that removes each buffer component one at a time. ΔOP is the performance drop relative to the full model's 0.465. Each variant is evaluated in a single run under the same protocol as the full model. Best values are in **bold**.

Variant	OP↑	BWT↑	Fgt↓	FWT↑	ΔOP
AMBER (full)	**0.465**	+0.002	0.032	**+0.130**	—
− Boundary selection	0.463	−0.006	0.021	+0.128	−0.002
− Half-life replacement	0.462	+0.001	0.018	+0.121	−0.003
− Generative replay	0.464	+0.004	0.023	+0.121	−0.001
− Age-decay weight	0.459	**+0.005**	0.016	+0.119	−0.006
− Consistency loss	0.460	**+0.005**	**0.015**	+0.101	−0.005

4.4 Analysis



(a) (b)

Figure 4: (a) Average performance of AMBER and its ablations on TRACE-8. The horizontal axis denotes continual learning stages and the vertical axis denotes AP_t. AMBER trails some ablations at T3 and T4 but leads from T5 onward. (b) BWT and FWT trajectories of I-LoRA, SSR, FOREVER, and AMBER on TRACE-8.

Figure 4 shows that the late-stage advantage of AMBER emerges from T5 onward, suggesting that age-aware replay becomes more beneficial as the sequence grows and historical supervision increasingly requires updating. Its BWT remains close to zero while its FWT leads from the early stages. These results indicate that AMBER improves stability and plasticity jointly rather than sacrificing current-task performance. The corresponding performance matrices are provided in Appendix 5.

5 Conclusion

We propose AMBER, a continual learning method that transforms the replay buffer from a static storage system into one that evolves with task age. AMBER couples the three stages of writing, maintenance, and sampling through the dimension of sample age. The reservoir and boundary selection provide diverse signals for content updates, half-life replacement ensures a smooth fade-out of old samples, and age-decay weighting enables the replay frequency to track the recency of the task. Experiments on the TRACE-8 benchmark with Llama-2-7B demonstrate that AMBER achieves a superior overall performance compared with existing replay methods, and the ablation study confirms the necessity and complementarity of each component.

The current study has four main limitations. All reported results rest on a single run per configuration, so the margins between AMBER and the strongest replay baselines are not accompanied by variance estimates, and no significance test is performed. The evaluation is restricted to continual instruction fine-tuning in the text modality, to a single storage budget, and to the dual-adapter configuration adopted above. Although the buffering mechanism does not depend on a specific form of LoRA parameters and can therefore be transferred to other continual fine-tuning frameworks, the scope of the experiments reported here is limited to that dual-adapter configuration. Consequently, neither its transfer to multimodal settings nor its behavior under a very different capacity is tested. The half-life schedule is a fixed exponential prior with a single age scale T_{half}, and we do not report a systematic sweep over T_{half} or the decay coefficient λ, so the sensitivity of the results to those two choices remains unquantified. Finally, the generative maintenance stage relies on the slow adapter producing behavioral traces that remain useful after several rounds of replacement, and the long-sequence regime where this assumption fails is not characterized here. In future work, we plan to extend the evolutionary buffering mechanism to multimodal continual learning, to report multi-seed results, and to learn scheduling parameters such as the half-life and the decay coefficient online from training data in place of the fixed decay prior.

REPRODUCIBILITY STATEMENT

The task sequence is the eight-task TRACE-8 benchmark described in Section 4.1, comprising C-STANCE, FOMC, MeetingBank, ScienceQA, NumGLUE-cm, 20Minuten, MedMCQA, and JEC-QA in that order; the first six tasks follow the TRACE definitions and order. All experiments use Llama-2-7B as the base model with LoRA adapters, and the buffer capacity is fixed at $M = 500$ for the replay methods, with FOREVER sampling 2% of each historical task as its memory. The hyperparameters of the buffering mechanism are the half-life T_{half}, the age-decay coefficient λ, the generation interval g, the boundary quantiles p_{low} and p_{high}, and the number of protected anchors k, all of which are listed in Algorithm 1; the EMA bound $\alpha_{\max}$ is defined in Eq. equation 1. Baselines marked † in Table 1 use results reported by Ren et al. (Ren et al., 2024), baselines marked ‡ were rerun from their official implementations under the unified protocol described in Section 4.1, and the I-LoRA EMA coefficient was set to 0.75 for that rerun. The data splits and the per-task evaluation metrics are given in Appendix B.1. Code will be publicly released upon acceptance.

AI USE STATEMENT

In this work, we used generative AI tools to assist with literature search and expression polishing, and all AI-assisted content has been manually reviewed. The experiments, code, and results were independently completed and verified by the authors. We take full responsibility for the final content of this work, including the Text, conclusions, and all artifacts produced with the aid of generative AI.

REFERENCES

Rahaf Aljundi, Min Lin, Baptiste Goujaud, and Yoshua Bengio. Gradient based sample selection for online continual learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

Pietro Buzzega, Matteo Boschini, Angelo Porrello, Davide Abati, and Simone Calderara. Dark experience for general continual learning: a strong, simple baseline. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020a.

Pietro Buzzega, Matteo Boschini, Angelo Porrello, Davide Abati, and Simone Calderara. Dark experience for general continual learning: a strong, simple baseline. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020b.

Arslan Chaudhry, Puneet K. Dokania, Thalaiyasingam Ajanthan, and Philip H. S. Torr. Riemannian walk for incremental learning: Understanding forgetting and intransigence. In *European Conference on Computer Vision (ECCV)*, 2018.

Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. Efficient lifelong learning with A-GEM. In *International Conference on Learning Representations (ICLR)*, 2019.

Tianze Chen et al. On-policy replay for continual supervised fine-tuning. *arXiv preprint arXiv:2605.29495*, 2026.

Junze Deng, Qinhang Wu, Peizhong Ju, Sen Lin, Yingbin Liang, and Ness Shroff. Unlocking the power of rehearsal in continual learning: A theoretical perspective. In *International Conference on Machine Learning (ICML)*, volume 267 of *Proceedings of Machine Learning Research*, pp. 13277–13326, 2025.

Yujie Feng, Hao Wang, Jian Li, Xu Chu, Zhaolu Kang, Yiran Liu, Yasha Wang, Philip S. Yu, and Xiao-Ming Wu. FOREVER: FORgEtting curVe-inspired Memory Replay for language model continual learning. *arXiv preprint arXiv:2601.03938*, 2026.

Md Yousuf Harun, Jhair Gallardo, Junyu Chen, and Christopher Kanan. GRASP: A rehearsal policy for efficient online continual learning. *arXiv preprint arXiv:2308.13646*, 2023.

Christopher Hazard and Zafeirios Fountas. SuRe: Surprise-driven prioritised replay for continual LLM learning. *arXiv preprint arXiv:2511.22367*, 2025.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Gao, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations (ICLR)*, 2022.

Yebowen Hu, Timothy Ganter, Hanieh Deilamsalehy, Franck Dernoncourt, Hassan Foroosh, and Fei Liu. MeetingBank: A benchmark dataset for meeting summarization. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 16409–16423, 2023.

Jianheng Huang, Leyang Cui, Ante Wang, Chengyi Yang, Xinting Liao, Linfeng Song, Junfeng Yao, and Jinsong Su. Mitigating catastrophic forgetting in large language models with self-synthesized rehearsal. *arXiv preprint arXiv:2403.01244*, 2024.

D. K. Human-like forgetting curves in deep neural networks. *arXiv preprint arXiv:2506.12034*, 2025.

Haeyong Kang et al. Do your best and get enough rest for continual learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences (PNAS)*, 114(13):3521–3526, 2017.

Samuli Laine and Timo Aila. Temporal ensembling for semi-supervised learning. In *International Conference on Learning Representations (ICLR)*, 2017.

Xingyu Li, Bo Tang, and Haifeng Li. AdaER: An adaptive experience replay approach for continual lifelong learning. *Neurocomputing*, pp. 127204, 2023.

Zhizhong Li and Derek Hoiem. Learning without forgetting. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 40, pp. 2935–2947, 2017.

Xize Liang, Lin Yang, Jie Wang, Yiyang Lu, Runyu Wu, Hanzhu Chen, and Jianye Hao. Boosting multi-domain fine-tuning of large language models through evolving interactions between samples. In *International Conference on Machine Learning (ICML)*, volume 267 of *Proceedings of Machine Learning Research*, pp. 37427–37441, 2025.

David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

Pan Lu, Swaroop Mishra, Tony Xia, Liang Qiu, Kai-Wei Chang, Song-Chun Zhu, Oyvind Tafjord, Peter Clark, and Ashwin Kalyan. Learn to explain: Multimodal reasoning via thought chains for science question answering. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Shuo Lu and Chunlei He. MSSR: Memory-aware adaptive replay for continual LLM fine-tuning. *arXiv preprint arXiv:2603.09892*, 2026.

Swaroop Mishra, Arindam Mitra, Neeraj Varshney, Bhavdeep Sachdeva, Peter Clark, Chitta Baral, and Ashwin Kalyan. NumGLUE: A suite of fundamental yet challenging mathematical reasoning tasks. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2022.

Ankit Pal, Logesh Kumar Umapathi, and Malaikannan Sankarasubbu. MedMCQA: A large-scale multi-subject multi-choice dataset for medical domain question answering. In *Conference on Health, Inference, and Learning*, volume 174 of *Proceedings of Machine Learning Research*, pp. 248–260, 2022.

Biqing Qi, Bowen Zhou, Fangyuan Li, et al. Online DPO: Online direct preference optimization with fast-slow chasing. *arXiv preprint arXiv:2406.05534*, 2024.

Dan Qiao and Mehrdad Mahdavi. Merge before forget: A single LoRA continual learning via continual merging. *arXiv preprint arXiv:2512.23017*, 2025.

Anastasia Razdaibiedina, Yuning Mao, Rui Hou, Madian Khabsa, Mike Lewis, and Amjad Almahairi. Progressive prompts: Continual learning for language models. In *International Conference on Learning Representations (ICLR)*, 2023.

Weijieying Ren, Xinlong Li, Lei Wang, Tianxiang Zhao, and Wei Qin. Analyzing and reducing catastrophic forgetting in parameter efficient tuning. *arXiv preprint arXiv:2402.18865*, 2024.

Annette Rios, Nicolas Spring, Tannon Kew, Marek Kostrzewa, Andreas Säuberli, Mathias Müller, and Sarah Ebling. A new dataset and efficient baselines for document-level text simplification in German. In *Proceedings of the Third Workshop on New Frontiers in Summarization (NewSum)*, pp. 152–161, 2021.

Anthony Robins. Catastrophic forgetting, rehearsal and pseudorehearsal. *Connection Science*, 7(2): 123–146, 1995.

David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy Lillicrap, and Gregory Wayne. Experience replay for continual learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. In *International Conference on Learning Representations (ICLR)*, 2018.

Agam Shah, Suvan Paturi, and Sudheer Chava. Trillion dollar words: A new financial dataset, task & market analysis. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 6664–6679, 2023.

Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 2990–2999, 2017.

Ilia Shumailov, Zakhar Shumaylov, Yiren Zhao, Yarin Gal, Nicolas Papernot, and Ross Anderson. The curse of recursion: Training on generated data makes models forget. *Nature*, 2024.

James Seale Smith, Lazar Valkov, Shaunak Halbe, Vyshnavi Gutta, Rogerio Feris, Zsolt Kira, and Leonid Karlinsky. Adaptive memory replay for continual learning. *arXiv preprint arXiv:2404.12526*, 2024.

Fan-Keng Sun, Cheng-Hao Ho, and Hung-yi Lee. LAMOL: LAnguage MOdeling for lifelong language learning. In *International Conference on Learning Representations (ICLR)*, 2020.

Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

Jeffrey S. Vitter. Random sampling with a reservoir. *ACM Transactions on Mathematical Software (TOMS)*, 11(1):37–57, 1985a.

Jeffrey S. Vitter. Random sampling with a reservoir. *ACM Transactions on Mathematical Software (TOMS)*, 11(1):37–57, 1985b.

Xiao Wang, Yuansen Zhang, Tianze Chen, Songyang Gao, Senjie Jin, Xianjun Yang, Zhiheng Xi, Rui Zheng, Yicheng Zou, Tao Gui, et al. TRACE: A comprehensive benchmark for continual learning in large language models. *arXiv preprint arXiv:2310.06762*, 2023.

Zifeng Wang, Zizhao Zhang, Chen-Yu Lee, Han Zhang, Ruoxi Sun, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, and Tomas Pfister. Learning to prompt for continual learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

Max Welling. Herding dynamical weights to learn. In *International Conference on Machine Learning (ICML)*, 2009.

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Gradient surgery for multi-task learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Xiaobing Yu, Jin Yang, Xiao Wu, Peijie Qiu, and Xiaofeng Liu. FM-LoRA: Factorized low-rank meta-prompting for continual learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2025.

Zheng Zhang et al. GeRe: General samples replay for anti-forgetting in continual LLM fine-tuning. *arXiv preprint arXiv:2508.04676*, 2025.

Chenye Zhao, Yingjie Li, and Cornelia Caragea. C-STANCE: A large dataset for Chinese zero-shot stance detection. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 13369–13385, 2023.

Linglan Zhao, Xuerui Zhang, Ke Yan, Shouhong Ding, and Weiran Huang. SAFE: Slow and fast parameter-efficient tuning for continual learning with pre-trained models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Haoxi Zhong, Chaojun Xiao, Cunchao Tu, Tianyang Zhang, Zhiyuan Liu, and Maosong Sun. JEC-QA: A legal-domain question answering dataset. In *AAAI Conference on Artificial Intelligence*, pp. 9701–9708, 2020.

Kanglei Zhou, Chang Li, Qingyi Pan, and Liyuan Wang. BriMA: Bridged modality adaptation for multi-modal continual action quality assessment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 38904–38914, 2026.

A PRELIMINARIES

A.1 CONTINUAL LEARNING METRICS

Let $\mathbf{R} \in \mathbb{R}^{T \times T}$ denote the performance matrix where $R_{i,j}$ is the test score on task j after the model finishes training on task i. The four metrics in the main text are all read from this single matrix and follow standard definitions in the continual learning literature. We restate them with their original sources.

Overall performance is $\text{OP} = \text{AP}_8$, where

$$\text{AP}_t = \frac{1}{t} \sum_{i=1}^{t} R_{t,i}.$$

It averages the final row over all eight tasks and measures how well the model retains the whole sequence. The definition follows the TRACE benchmark (Wang et al., 2023) and I-LoRA (Ren et al., 2024).

Backward transfer is

$$\text{BWT} = \frac{1}{T-1} \sum_{j=1}^{T-1} (R_{T,j} - R_{j,j}).$$

It averages the change of each earlier task between the stage when it was learned and the final stage and originates from Lopez-Paz and Ranzato (Lopez-Paz & Ranzato, 2017). A positive value means earlier tasks improve on average and a value near zero means they are neither improved nor degraded.

Forgetting is

$$\text{Fgt} = \frac{1}{K-1} \sum_{j=1}^{K-1} \left(\max_{u \in \{j,\ldots,K\}} R_{u,j} - R_{K,j} \right),$$

with $K = T$. It averages the drop of each task from its best recorded score and follows Chaudhry et al. (Chaudhry et al., 2018). A lower value indicates less forgetting.

Forward transfer is

$$\text{FWT} = \frac{1}{T-1} \sum_{j=2}^{T} (R_{j-1,j} - b_j),$$

where b_j is the zero-shot score on task j as listed in Table 3. It measures how much training on the previous task helps the current task relative to a zero-shot model and follows (Lopez-Paz & Ranzato, 2017; Ren et al., 2024).

Table 3: Zero-shot per-task scores on TRACE-8. From Ren et al. Ren et al. (2024).

Zero-shot baseline (per-task score)								
Method	**C-STANCE**	**FOMC**	**MeetingBank**	**ScienceQA**	**NumGLUE**	**20Minuten**	**MedMCQA**	**JEC-QA**
Zero-shot	0.350	0.240	0.100	0.230	0.280	0.300	0.240	0.120

All reported metrics use a single run, the corresponding scores should therefore be read as point estimates rather than as estimates with quantified variance.

A.2 NOTATION SUMMARY

Table 4 lists the main symbols used in the paper.

A.3 ONE-SIDED CONSISTENCY REGULARIZATION

Classical consistency regularization encourages a model to produce stable outputs under perturbations of its inputs or parameters. Temporal Ensembling (Laine & Aila, 2017) keeps an exponential

Table 4: Notation used in the paper, with references to the defining equations or sections.

Symbol	Meaning	Reference in main text
T	number of tasks in the sequence, $T = 8$	Appendix A.1
T_t	the t-th task	Sec. 3.1
$\mathcal{D}_t$	training set of task t	Eq. (2), Sec. 3.1
N_t	number of samples in $\mathcal{D}_t$	Sec. 3.1
$\mathcal{B}$, $\mathcal{B}_t$	replay buffer and its state after task t	Sec. 3.2.1
M	buffer capacity	Sec. 3.1, Sec. 3.2.1
N_t^B	buffer occupancy after task t	Sec. 3.2.1
$b_i = (s_i, y_i, \tau_i, z_i)$	buffer entry: input, supervision, source task, real/generated flag	Sec. 3.2.1
$a_i(t) = t - \tau_i$	sample age of entry i at task t	Eq. (3)
$z_i \in \{0, 1\}$	real sample when $z_i = 0$; generated trace when $z_i = 1$	Sec. 3.2.1
θ_{f}	fast adapter parameters	Eq. (1), Sec. 3.1
θ_{s}	slow adapter parameters	Eq. (1), Sec. 3.1
α	exponential moving average coefficient	Eq. (1)
$\alpha_{\max}$	upper bound of the EMA coefficient	Eq. (1)
T_{half}	half-life age scale in replacement	Eq. (6)
λ	age-decay coefficient in replay sampling	Eq. (8)
g	generation frequency in training steps	Sec. 3.2.2, Alg. 1
k	number of protected anchors per task	Eq. (7)
p_{low}, p_{high}	lower and upper loss quantiles for boundary selection	Eq. (5)
$\bar{L}_i$	token-normalized loss of sample i	Eq. (5)
m_i	number of supervised tokens in sample i	Eq. (5)
m_b	number of valid label positions in sample b	Eq. (9)
q_p	empirical loss quantile	Eq. (5)
$\mathcal{P}_t$	protected anchor set	Eq. (7)
$N_{t,j}^{\mathrm{real}}$	number of real entries from task j in the buffer	Eq. (7)
$\mathcal{C}_t$	candidate set for buffer writing	Eq. (4)
$\mathcal{C}_t^{\mathrm{reservoir}}, \mathcal{C}_t^{\mathrm{select}}, \mathcal{C}_t^{\mathrm{gen}}$	reservoir, boundary-selection, and generation candidate sets	Eq. (4)
$\omega_i(t)$	sampling weight of entry i at task t	Eq. (8)
b_{rep}	replay batch size	Eq. (8)
$\mathcal{R}_t$	sampled replay batch	Eq. (8), Eq. (9)
$\mathcal{C}_{\mathrm{batch}}$	current task training batch	Eq. (9)
$\mathcal{L}_{\mathrm{CE}}$	cross-entropy loss	Eq. (9)
$\mathcal{L}_{\mathrm{cons}}$	one-sided consistency loss	Eq. (10)
λ_{cons}	weight of the consistency loss	Sec. 3.2.5
$\mathcal{H}$	set of hidden layers used for consistency	Eq. (10)
S, d	sequence length and hidden dimension	Eq. (10)
η	learning rate	Alg. 1
$R_{i,j}$	test score on task j after training on task i	Appendix A.1

moving average of predictions over epochs and penalizes the gap between the current prediction and this average. Mean Teacher (Tarvainen & Valpola, 2017) averages the model weights and uses the resulting teacher as the consistency target. Both align a student model with a slowly updated reference to stabilize training.

I-LoRA (Ren et al., 2024) adopts the dual-adapter form of Mean Teacher. The consistency term in I-LoRA is a mean squared error between the fast learner embedding and the slow learner embedding on replayed samples. This alignment is symmetric because squared error penalizes deviations in both directions equally. The slow learner is updated only through the moving average and is never reloaded between tasks. Formally, the fast learner minimizes

$$\min_{\theta_w} \mathcal{L} = \mathbb{E}_{x \in \mathcal{M} \cup \mathcal{D}_t} \mathcal{L}_{\mathrm{CE}}\big(f(x; \theta_w), y\big) + \gamma\, \mathbb{E}_{x \in \mathcal{M}} \mathcal{L}_{\mathrm{MSE}}\big(f_o(x; \theta_w), z\big),$$

where f_o denotes the embedding extractor of the model and $z = f_o(x; \theta_l)$ records the embedding produced by the slow learner. The slow learner is updated only as an exponential moving average

Table 5: Statistics of the eight tasks in TRACE-8.

Task	Metric	Source	Language	Avg len	Train/Test
C-STANCE	accuracy	(Zhao et al., 2023)	Chinese	163	5000/2000
FOMC	accuracy	(Shah et al., 2023)	English	329	5000/496
MeetingBank	ROUGE-L	(Hu et al., 2023)	English	16113	5000/692
ScienceQA	accuracy	(Lu et al., 2022)	English	320	5000/2000
NumGLUE-cm	accuracy	(Mishra et al., 2022)	English	206	5000/81
20Minuten	SARI	(Rios et al., 2021)	German	2419	5000/200
MedMCQA	accuracy	(Pal et al., 2022)	English	258	5000/2816
JEC-QA	accuracy	(Zhong et al., 2020)	Chinese	221	5000/500

Table 6: Prompt templates for the eight tasks in TRACE-8. The prompts of C-STANCE and JEC-QA, whose source data is Chinese, are shown in English translation.

Task	Prompt
C-STANCE	Judge the stance of the following text toward the given target. Choose one: A. support, B. against, C. neutral. Output A, B, or C.
FOMC	What is the monetary policy stance for the following text: A. dovish, B. hawkish, C. neutral. Choose one from A, B, and C.
MeetingBank	Write a summary of the following meeting transcripts.
ScienceQA	Choose an answer for the following question and give your reasons.
NumGLUE-cm	Solve the following math problem.
20Minuten	Provide a simplified version of the following paragraph in German.
MedMCQA	Solve the following medical problem by choosing the correct answer from the following four choices.
JEC-QA	Based on the following legal question, choose the correct answer from options A, B, C, and D.

of the fast learner,$\theta_l^k = \lambda\theta_l^{k-1} + (1-\lambda)\theta_w^k$with a fixed step size λ and update step k. In AMBER's notation, the replay memory $\mathcal{M}$, the fast learner θ_w, and the slow learner θ_l correspond to the buffer $\mathcal{B}$, the fast adapter θ_{f}, and the slow adapter θ_{s}, and the step size λ corresponds to the EMA coefficient α.

AMBER keeps only the lower bound direction of this alignment. Its consistency loss applies a ReLU gate so that the fast adapter is penalized only when its hidden activation falls below the slow adapter activation and is left free when it rises above. This asymmetric design lets the fast adapter build new representations without being pulled back toward old ones while still preventing it from suppressing the consolidated representations of the slow adapter. The full definition of $\mathcal{L}_{\mathrm{cons}}$ appears in the method section.

B Experimental Setup

B.1 Datasets and Task Sequence

Tables 5 and 6 summarize the task statistics and prompt templates. The first six tasks follow the task definitions and fixed order of TRACE (Wang et al., 2023). MedMCQA and JEC-QA extend the sequence with medical and legal question answering. For each task we report the evaluation metric, the original source, the language, the average prompt length in characters, the number of training and test samples, and the prompt template. C-STANCE and JEC-QA are originally Chinese datasets, so their prompts are shown here in English translation for readability while the underlying training data remains in Chinese. The tasks span Chinese stance detection, financial question answering, meeting summarization, scientific reasoning, numerical reasoning, text simplification, and specialized medical and legal question answering, with substantial variation in domain and task format.

One representative training example per task is shown below, with the prompt and the reference answer separated by a rule.

C-STANCE

Prompt

Judge the stance of the following text toward the given target. Choose one, A. support, B. against, C. neutral. Output A, B, or C.
Text:
How to wear a mask correctly. When wearing a disposable mask, the end with the metal nose clip faces upward. After fully unfolding the mask, press the upper metal strip against the bridge of the nose so that the mask fits the face completely. Remember to wash your hands before wearing a mask. The mask should be replaced after four consecutive hours of wear.
Target,
Wearing a mask
Stance,

Answer

A

FOMC

Prompt

What is the monetary policy stance for the following text? A. dovish, B. hawkish, C. neutral. Choose one from A, B and C.
Text:
I mean, we—the—again, the relationship between slack in the economy and inflation is weak, has been weak.
Stance:

Answer

C

MeetingBank

Prompt

Write a summary of the following meeting transcripts.
Meeting transcripts:
Councilmember Clark, would you please put resolution 2a1 on the floor for adoption. As council president of the council, resolution 201 be adopted. Thank you. We've got a mover in a second. Comments and questions by members of Council on Resolution 201. Councilmember CdeBaca. Thank you. I just want to go on record voting no for this tonight. Pepsi is not necessarily a group that needs $1,000,000 incentive to stay in Denver, and they are well on their way to completing their move with or without this incentive. So definitely just want to make sure that I'm clear in not supporting this tonight. Thank you. Thank you. And I'll go ahead. And since this is in my council district, District 11, we as community members, along with some of the original steering committee members from the far northeast neighborhood plan, reconvened and talked through all of the pros and possible challenges of welcoming a large scale manufacturer into District 11. And the community saw fit that this was a great use for this area and this will bring close to 500 jobs to the region as well. And I would ask my colleagues to support it tonight. Madam Secretary, roll call on Council Resolution 22, Dash 201, please. See tobacco? No. Clark. I. Swen. I. Herndon, I. Hi. All right. Cashman. Kenny Ortega. Sandoval. I. Sawyer. I. Torres. I. Black. Madam President, I. Madam Secretary, please close the voting and announce the results. One day, 11 eyes. 11 Eyes. Council Resolution 22, dash 201 has been adopted. Madam Secretary, please put the next item up on our screens. Councilmember Flynn, go ahead with your comments on resolution 159, please.
Summary:

Answer

A resolution approving a proposed Business Incentive Fund (BIF) contract with Bottling Group, LLC for an amount not to exceed $1,000,000 for 7 years and for performance based criteria that include establishment of a state-of-the-art manufacturing facility and associated significant investments, as well as job retention and creation and associated investments into upskilling and training of the manufacturing facility's Denver-based workforce.
Approves a Business Incentive Fund (BIF) contract with Bottling Group, LLC for an amount not to exceed $1,000,000 and for 7 years through 12-31-2028 for performance based criteria that include establishment of a state-of-the-art manufacturing facility and associated significant investments, as well as job retention and creation and associated investments into upskilling and training of the manufacturing facility's Denver-based workforce (OEDEV-202160849-00). The last regularly scheduled Council meeting within the 30-day review period is on 4-11-2022. The Committee approved filing this item at its meeting on 3-9-2022.

ScienceQA

Prompt

Choose an answer for the following question and give your reasons.
Question:
What information supports the conclusion that Trent acquired this trait?
Hint:
Read the description of a trait.
Trent knows how to fly a kite.
Choices:
A. Trent's neighbor taught him how to repair a kite.
B. Trent likes to fly a kite with his younger brother.
C. Trent's friend taught him how to fly a kite.
Answer:

Answer

C
Organisms, including people, have both inherited and acquired traits. Inherited and acquired traits are gained in different ways.
Inherited traits are passed down from biological parents to their offspring through genes. Genes are pieces of hereditary material that contain the instructions that affect inherited traits. Offspring receive their genes, and therefore gain their inherited traits, from their biological parents. Inherited traits do not need to be learned.
Acquired traits are gained during a person's life. Some acquired traits, such as riding a bicycle, are gained by learning. Other acquired traits, such as scars, are caused by the environment. Parents do not pass acquired traits down to their offspring.

NumGLUE-cm

Prompt

Solve the following math problem.
Question:
In a car parking lot, Greyson observed that there are 48 car wheels. Find out the number of cars in the parking lot.
Answer:

Answer

12

20Minuten

Prompt

Provide a simplified version of the following paragraph in German.
Paragraph:
Im bernischen Lyss hat die Polizei eine illegale "Spielhölle" ausgehoben und zwei Frauen im Alter von 30 und 41 Jahren festgenommen. Bei der Aktion am Freitagabend wurden gut 40 Personen wegen des Verdachts auf illegale Glücksspiele und Wetten kontrolliert. Die beiden Frauen gingen im Lokal mutmasslich ohne Bewilligung einer Erwerbstätigkeit nach und sollen wegen Widerhandlungen gegen das Ausländer- und Integrationsgesetz angezeigt werden, wie die Regionale Staatsanwaltschaft Berner Jura-Seeland und die Kantonspolizei Bern am Samstag mitteilten. Im kontrollierten Lokal seien verbotene Glücksspiele und Wettgeschäfte angeboten worden, hiess es. Im Zuge der Kontrolle und einer damit verbundenen Hausdurchsuchung seien unter anderem mehrere elektronische Geräte und ein Spielautomat, mehrere zehntausend Franken Bargeld sowie mutmassliches Gold sichergestellt worden. Der Aktion waren den Angaben zufolge umfangreiche Ermittlungen vorausgegangen. Die Staatsanwaltschaft ermittelt weiter insbesondere zu den Betreibern des Lokals. In Lyss war es schon mehrfach zu Razzien wegen illegalen Glücksspiels und illegaler Sportwetten gekommen. 2018 wurden in einem Lokal 25 Personen kontrolliert und Bargeld, Wettcomputer und Spielautomaten sichergestellt. Ähnliche Aktionen gab es unter anderem auch schon 2016 und 2015.
Simplification:

Answer

In Lyss BE führte die Kapo Bern eine gezielte Kontrolle durch. Bei der Aktion wurde Geld sichergestellt und zwei Frauen festgenommen. Bereits in den Jahren zuvor kam es zu ähnlichen Kontrollen.

MedMCQA

Prompt
Solve the following medical problem by choosing the correct answer from the following four choices.
A patient presented with hemoptysis and persistent cough. The chest X-ray of the patient was normal. The next best investigation is?
A. Helical CT
B. High resolution CT
C. Bronchoscopy
D. Angiography

Answer
B

JEC-QA

Prompt
Based on the following legal question, choose the correct answer from options A, B, C, and D.
Question:
According to the provisions of the Company Law, after the inaugural meeting of a joint-stock company adopts the articles of association and elects the board of directors and the supervisory board, which body or person shall submit the documents for company establishment and apply for registration to the company registration authority?
A. The board of directors
B. The supervisory board
C. The representative designated by all shareholders
D. The agent jointly entrusted by all shareholders

Answer
A

B.2 Full Hyperparameter Tables

Table 7 lists the hyperparameters reported in the I-LoRA paper (Ren et al., 2024). The paper does not state the memory size or the memory update frequency, so our implementation follows its open-source code with a buffer size of 500 and a replay ratio of 0.3.

Table 7: Hyperparameters reported in the I-LoRA paper (Ren et al., 2024).

Setting	Value
Base model	Llama-2-7B
LoRA rank r	8
LoRA alpha	16
LoRA target modules	query and value
Learning rate	1×10^{-4}
Warmup ratio	0.2
Batch size	16
Training samples per task	5000
Evaluation samples per task	500
Hardware	2 RTX 4090 GPUs
Memory size	not reported in the paper
Update frequency	not reported in the paper

Table 8 compares the hyperparameters of the four implementations evaluated in this paper.

All baselines follow their original papers and official implementations under the same TRACE-8 protocol to keep the comparison fair, with their shared and method-specific settings listed in Table 8. The only adjustment is I-LoRA's EMA coefficient, set to 0.75 instead of the reported 0.25 so that its slow adapter consolidates enough history under the unified protocol.

The AMBER-specific choices follow the design goals in the main text. The EMA coefficient lets the slow adapter serve as a stable teacher for generation and consistency while still tracking the fast adapter within a task. The half-life 50 is mild, so the oldest sample of age seven has a per-scan

Table 8: Hyperparameters of the four implementations evaluated in this paper.

Setting	I-LoRA	SSR	FOREVER	AMBER
Base model	Llama-2-7B	Llama-2-7B	Llama-2-7B unsloth	Llama-2-7B
LoRA rank r	8	8	8	8
LoRA alpha	16	16	16	16
LoRA dropout	0.05	0.05	0.05	0.05
Target modules	q, v	q, v	q, v	q, v
Learning rate	1×10^{-4}	1×10^{-4}	1×10^{-4}	1×10^{-4}
Epochs	3	3	3, first task 10	3
Warmup ratio	0.2	0.2	0.2	0.2
Max length	1024	1024	1024	1024
Batch size	1	1	1	1
Grad accumulation	8	8	8	4
Buffer or memory	500	10% rehearsal	2% memory	500
Method-specific	replay ratio 0.3, EMA 0.75	rehearsal ratio 0.10	steps per day 24, memory epochs 2, trigger days 1, 2, 4, 7, 15, 30, 60, 90, 120	EMA 0.75, half-life 50, generation every 10 steps, boundary 0.4 and 0.6, protect 5

replacement probability of only about 9%, renewing historical supervision gradually rather than at a single cut-off. The boundary band selects the middle-difficulty samples most informative for replay, and the protected-anchor budget reserves forty slots, about 8% of the capacity, as a fixed real-data backbone.

AMBER uses the dual-adapter architecture described in the main text. At each task boundary the slow adapter is reloaded from the fast adapter checkpoint before the maintenance scan. Generation uses a prefix length of 20 and a maximum of 50 new tokens. The remaining methods use bf16 precision.

C Detailed Descriptions of Compared Methods

This appendix describes the baselines compared in Section 4 in detail. They fall into two groups. The non-replay methods, EWC (Kirkpatrick et al., 2017), L2P (Wang et al., 2022), and Progressive Prompts (PP) (Razdaibiedina et al., 2023), test whether parameter regularization, prompt reuse, or prompt isolation can avoid catastrophic forgetting without storing historical samples. The replay methods, ER (Rolnick et al., 2019), GEM (Lopez-Paz & Ranzato, 2017), A-GEM (Chaudhry et al., 2019), I-LoRA (Ren et al., 2024), SSR (Huang et al., 2024), and FOREVER (Feng et al., 2026), test different ways of storing, constraining, generating, or scheduling historical supervision. All methods are evaluated under the same TRACE-8 protocol, and their implementation details are summarized in Appendix B.2.

C.1 Non-Replay Methods

EWC. Elastic Weight Consolidation (EWC) (Kirkpatrick et al., 2017) is a regularization-based continual learning method. After training on a task, it estimates the importance of each parameter, typically through the diagonal Fisher information matrix, and adds a quadratic penalty that discourages later updates from moving important parameters away from their previous values. EWC does not store raw historical samples, and therefore serves as a reference for how much retention can be achieved by parameter regularization alone.

L2P. Learning to Prompt (L2P) (Wang et al., 2022) maintains a pool of learnable prompts together with matching keys. For each input, a query representation is compared with the prompt keys, and a small set of prompts is retrieved and prepended to the input. Only the prompt pool and the task-relevant trainable parameters are updated, while the backbone is kept frozen. This design stores task knowledge in a shared prompt space rather than in the model weights, allowing different tasks to reuse prompts without explicitly replaying historical data.

Progressive Prompts (PP). Progressive Prompts (Razdaibiedina et al., 2023) allocates a new set of prompts to each incoming task and freezes the prompts learned for previous tasks. When learning task t, the model can attend to the newly introduced prompts together with the frozen prompts from earlier tasks, so earlier knowledge is preserved in a parameter-isolated prompt subspace. This avoids overwriting previous prompts but increases the number of stored prompts linearly with the number of tasks. PP represents a non-replay architectural baseline that trades memory growth for reduced interference.

C.2 Replay Methods

ER. Experience Replay (ER) (Rolnick et al., 2019) keeps a fixed-capacity buffer of historical samples. New samples are written into the buffer, usually by reservoir sampling, and each training step interleaves the current batch with a replay batch drawn from the buffer. The replay samples are optimized with the same supervised objective as the current task. ER is the simplest replay baseline, it does not use gradient constraints, generated data, or adaptive scheduling, and its buffer contents are essentially determined at insertion time.

GEM. Gradient Episodic Memory (GEM) (Lopez-Paz & Ranzato, 2017) stores a small set of samples from each previous task and uses them to constrain the update direction on the current task. At each step, GEM computes the gradients of the stored episodic memories and projects the current gradient so that it does not increase the loss on any previous task. When the current gradient conflicts with a previous-task gradient, the projection keeps the update within the feasible region defined by the old-task constraints. GEM is effective at limiting interference but requires solving a quadratic program over the stored task gradients, which becomes expensive as the number of tasks grows.

A-GEM. A-GEM (Chaudhry et al., 2019) simplifies GEM by replacing the per-task constraints with a single averaged gradient constraint. It stores episodic memories from previous tasks and computes one average gradient over them. If the current gradient has a negative dot product with this average gradient, it is projected onto the half-space that does not increase the average old-task loss. This reduces the computational cost of GEM while preserving much of its ability to control forgetting. A-GEM is therefore a more scalable gradient-constraint baseline for long task sequences.

I-LoRA. I-LoRA (Ren et al., 2024) uses a dual-adapter architecture. A fast LoRA adapter receives gradients and learns the current task, while a slow LoRA adapter is updated as an exponential moving average of the fast adapter and serves as a stable reference. During training, historical samples from a replay buffer are combined with the current batch, and a consistency loss aligns the fast adapter with the slow adapter on replayed samples. In the original formulation, this consistency is symmetric mean squared error between the fast and slow embeddings. I-LoRA shares the same dual-adapter backbone as AMBER and therefore isolates the effect of buffer management.

SSR. Self-Synthesized Rehearsal (SSR) (Huang et al., 2024) replaces the real historical buffer with synthetic rehearsal data generated by the language model itself. It typically proceeds in three stages: generating synthetic inputs for previous tasks from a small number of demonstrations, refining their outputs with the most recent model to preserve acquired abilities, and selecting diverse and high-quality synthetic instances for later rehearsal. Because SSR does not rely on retaining original historical samples, it tests whether self-generated supervision can substitute for real replay data. In our experiments, SSR uses a rehearsal ratio of 0.10, following its official configuration.

FOREVER. FOREVER (Feng et al., 2026) schedules replay according to an Ebbinghaus-style forgetting curve. It accumulates the stepwise norm of trainable-parameter updates into a model-time signal, calibrates a virtual day from the first 24 steps, and maps forgetting-curve intervals onto this virtual time to produce a dense-to-sparse replay schedule. It further scales the replay loss by an instability ratio estimated from the current update speed. This makes replay frequency adaptive to training dynamics, but the content of the memory remains fixed once written. In our implementation, FOREVER follows its official configuration on the unsloth Llama-2-7B checkpoint, with a 2% memory, 24 steps per virtual day, and two rehearsal epochs per memory item.

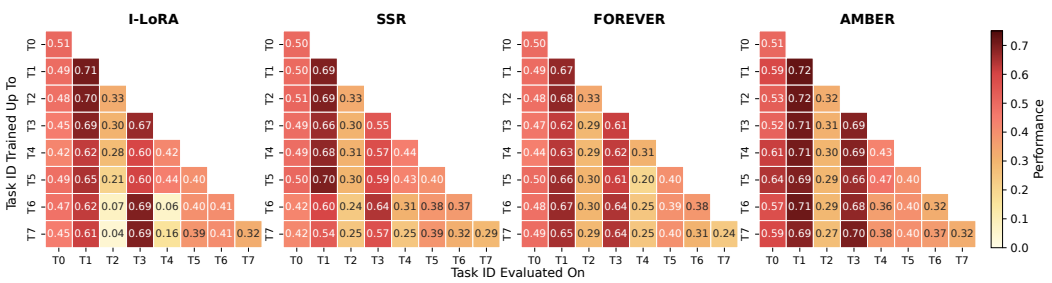


Figure 5: Task performance matrices of I-LoRA, SSR, FOREVER, and AMBER on TRACE-8.

D Additional Experimental Results

D.1 Performance Matrices

Figure 5 reports the task performance matrices of I-LoRA, SSR, FOREVER, and AMBER on TRACE-8. Entry (i, j) denotes test performance on task j after training on task i. Diagonal entries show performancte immediately after learning each task, and the lower triangle shows retention on earlier tasks. The upper triangle, corresponding to unseen tasks, is masked. AMBER maintains comparable diagonal performance while improving retention in the lower triangle.

D.2 Buffer Content Analysis

We dissect the final replay buffer to verify that age-aware maintenance produces the composition described in the main text. Figure 6 reports the source-task distribution of the 500 entries and the growth of generated traces across the eight task boundaries. The final buffer holds 442 real samples and 58 slow-adapter generations, a generated fraction of 11.6%. The real samples span all eight source tasks with a mild skew toward the later tasks, consistent with half-life replacement that retires older supervision gradually. The generated traces accumulate monotonically, while the number of protected anchors stays fixed at 40, five per task. This mirrors the observation in Figure 3 of the main text that the generated fraction grows as the sequence advances while the real-data backbone stays bounded.

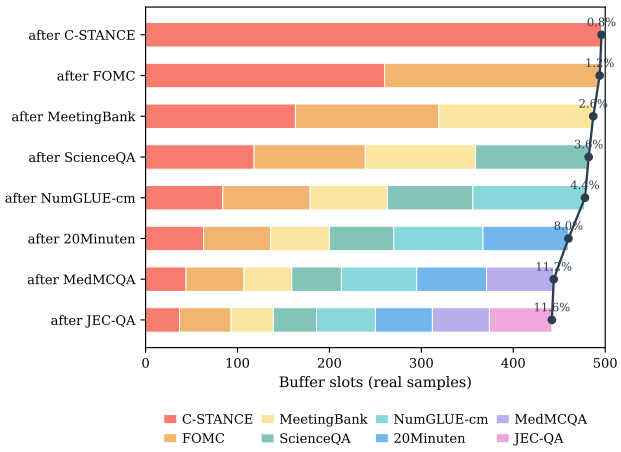


Figure 6: Composition of the final replay buffer after the full TRACE-8 sequence. The left panel stacks real and generated entries per source task. The right panel plots the cumulative number of generated entries across the eight task boundaries, which grows from 4 to 58 while the buffer size stays fixed at 500.

Table 9 lists representative slow-adapter generations retained in the buffer. Each row shows the twenty-token conditioning prefix in bold followed by the generated continuation in regular type.

Table 9: Reconstructed rehearsal samples generated by the slow adapter.

Sample	Conditioning prefix and generated continuation
Sample 1	**Write a summary of the following meeting transcripts. Meeting transcripts:** (4-5 sentences) The board of directors for the local grocery store held a meeting to discuss store inventory and staffing. The meeting began at 10:00 and lasted for 20 minutes
Sample 2	**Choose an answer for the following question and give your reasons. Question:** Water will freeze at 0°C or 32°F. Suppose that you are studying the freezing point of water. Which of the following temperatures would you expect this experiment to have? A. 0.000
Sample 3	**Solve the following math problem. Question: Everyday Charlotte spends 16**.50 and uses 1.50 coins. She has a total of 30 coins. How many coins does she have? Solution dev-null
Sample 4	**Provide a simplified version of the following paragraph in German. Paragraph: The** federal government has further tightened the corona rules for the opening phase. What is new is in italics. The purpose control is now also for the purchase
Sample 5	**Solve the following medical problem by choosing the correct answer from the following four choices.** A 70-year-old woman is evaluated in the office. She says that she has been having difficulty breathing. She describes her symptoms as feeling like she is sitting on something hard. She says that she has been feeling this way

D.3 Learning Curves

Figure 7 shows the training loss of the full AMBER model across the TRACE-8 sequence. The loss is logged every 50 steps, and each task contributes about 37 log points. Task boundaries are marked by vertical dashed lines. The loss rises at each task boundary because the model switches to a new task and then decreases within the task, confirming that a single shared adapter re-adapts to each incoming task from the checkpoint reached on the previous one. The first task shows a steep early drop as the adapter leaves its random initialization, and later tasks settle to task-dependent plateaus rather than returning to a common floor, so the varying final loss across tasks reflects differences in task difficulty instead of forgetting of earlier supervision.

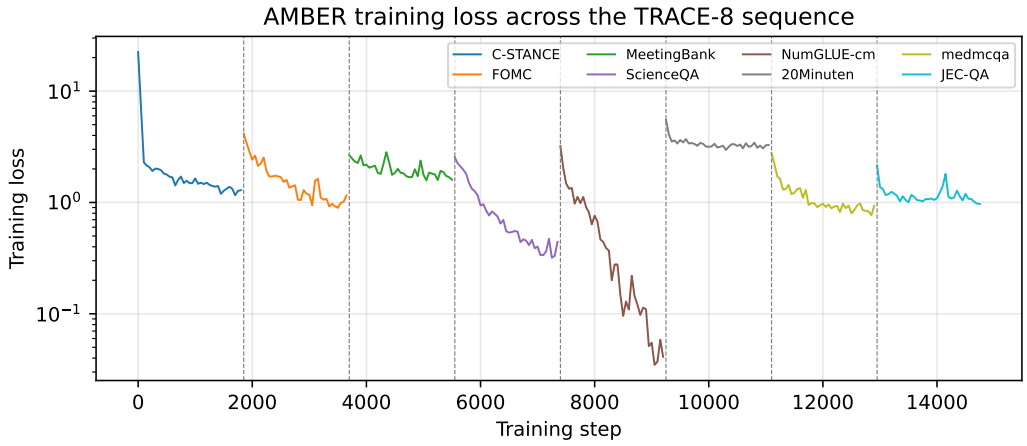


Figure 7: Training loss of the full AMBER model across the TRACE-8 sequence. Colors indicate tasks and vertical dashed lines mark task boundaries.

E Pseudocode Verification

Algorithm 1 summarizes one training task of AMBER.

Algorithm 1 AMBER training one task t.

Input: data $\mathcal{D}_t$, buffer $\mathcal{B}$, fast parameters θ_{f}, slow parameters θ_{s}; hyperparameters T_{half}, λ, g, k, α, b_{rep}, p_{low}, p_{high}, λ_{cons}
Output: updated θ_{f}, θ_{s}, $\mathcal{B}$

for each step, take a batch (x, y) from $\mathcal{D}_t$ **do**
  $\mathcal{B}.\mathrm{reservoirWrite}(x, y, t, \mathrm{real})$
  if $t > 0$ **then**
    $(x_b, y_b) \leftarrow \mathcal{B}.\mathrm{WeightedWithoutReplacement}(\mathcal{B}, \boldsymbol{\omega}(t), b_{\mathrm{rep}})$
    $\mathcal{L} \leftarrow \mathcal{L}_{\mathrm{CE}}\big(f([x; x_b]; \theta_{\mathrm{f}}), [y; y_b]\big) + \lambda_{\mathrm{cons}} \mathcal{L}_{\mathrm{cons}}(x_b)$
  else
    $\mathcal{L} \leftarrow \mathcal{L}_{\mathrm{CE}}\big(f(x; \theta_{\mathrm{f}}), y\big)$
  end if
  $\theta_{\mathrm{f}} \leftarrow \theta_{\mathrm{f}} - \eta \nabla \mathcal{L}$
  $\theta_{\mathrm{s}} \leftarrow \mathrm{EMA}(\theta_{\mathrm{s}}, \theta_{\mathrm{f}}, \alpha)$
  if step $\bmod g = 0$ **then**
    $\mathcal{B}.\mathrm{write}\big(\mathrm{slowGen}(\mathrm{prefix}_p(x)), t, \mathrm{gen}\big)$
  end if
end for
At epoch end: $\mathcal{B}.\mathrm{write}\big(\mathrm{lossQuantile}(\mathcal{D}_t, [p_{\mathrm{low}}, p_{\mathrm{high}}]), t, \mathrm{real}\big)$
At task end: $\theta_{\mathrm{s}} \leftarrow \mathrm{reload}(\theta_{\mathrm{f}})$; $\mathcal{B}.\mathrm{refreshProtected}(k)$; $\mathcal{B}.\mathrm{halfLifeReplace}(t, T_{\mathrm{half}})$