ENGRAM: FROM AGENT EXPERIENCE TO WEIGHTS THROUGH VERIFIED CONSOLIDATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Language-model agents use experience through external retrieval and parameter learning. Consolidating stored experience into model weights requires deciding both which experience to train on and when its external records can leave retrieval. Admission based solely on student mastery can exclude useful teacher demonstrations, while removal immediately after training can discard guidance that the model still needs. We introduce ENGRAM, a framework that separates training admission from memory release. A demonstration-aware gate admits skills based on student mastery or sufficient successful teacher demonstrations. The student learns from selected action traces with context from an Engram Trace Graph (ETG). A verification gate then compares the updated model without episodic retrieval against the base model with retrieval on reserved probes. Release requires valid training participation and scores meeting both relative and absolute criteria; records for skills that fail verification remain retrievable. Experiments across interactive and reasoning benchmarks show consistent gains over frozen-weight ETG controls, with ablations supporting demonstration-aware admission and selective adapter activation. A controlled held-out comparison shows that verified release substantially reduces the retrievable footprint while remaining close to retain-all performance and outperforming immediate and budget-matched random release. ENGRAM also improves task performance while reducing input-token usage and model calls. Skill-level verification further reduces retrievable experience through selective archival, retaining external guidance for skills that do not meet the release criteria. Code is available in anonymous.4open.science/r/engram-anonymous.

1 INTRODUCTION

Language-model agents use past experience through external retrieval and parameter updates. Retrieval keeps trajectories and feedback available for future tasks, while parameter learning can make their lessons usable without consulting the original records. Combining these mechanisms raises a practical question: *when has an agent learned enough from stored experience to remove the corresponding records from its retrieval memory?* Retaining every record incurs storage and retrieval costs, but premature removal can discard guidance that the model still needs.

Existing methods illustrate both approaches. Reflexion stores verbal feedback, Voyager accumulates executable skills, and MAGE organizes successful and failed experiences in a co-evolving graph (Shinn et al., 2023; Wang et al., 2023; Yang et al., 2026). Self-training methods such as STaR and ReST^{EM} instead update model parameters using selected generated solutions (Zelikman et al., 2022; Singh et al., 2023). An agent combining retrieval and training must distinguish useful learning material from records that are no longer needed after learning.

First, useful training experience may precede student mastery. A student can fail at prerequisite actions before reaching a more advanced skill. A stronger teacher can demonstrate that skill, but a mastery-only admission rule would still exclude it from training. Admission should therefore consider available demonstrations alongside current competence.

Second, completing training does not establish that memory can be removed. A selected skill may contribute no valid training examples, or the updated model may still depend on retrieved guidance. Premature removal creates *knowledge waste*: external experience becomes unavailable

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

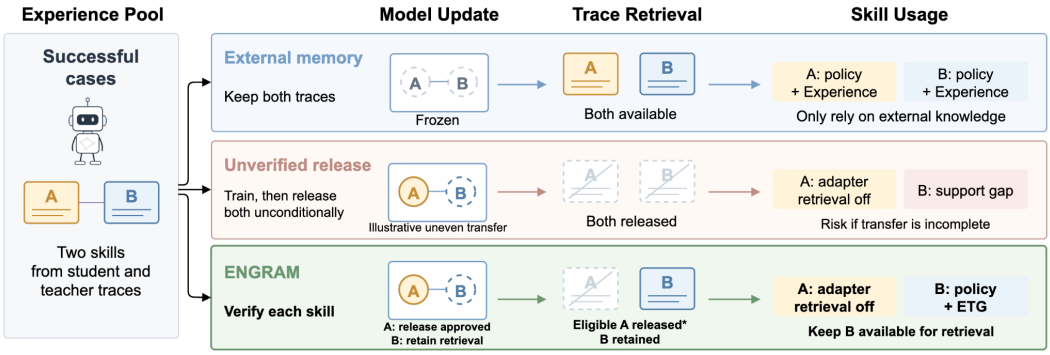

Figure 1: **Readiness to learn differs from readiness to release.** Removing both traces after training can leave skill B unsupported if only A has been learned. ENGRAM verifies each skill: A uses verified weights and releases eligible records, while B retains external memory. Configured residuals remain; colored nodes represent behavior.

before the weights can replace it. This can happen immediately after consolidation, independently of interference from later updates.

Verifying this transfer requires comparing the updated model without episodic retrieval against a retrieval-supported reference under the same task interface. Otherwise, retrieved demonstrations or differences in prompt format can obscure what the weights have learned. Action probes provide local evidence; complete rollouts and subsequent updates assess task performance and retention.

We introduce ENGRAM, a framework that separates training admission from memory release for each skill. The Engram Trace Graph (ETG) organizes experience by skill, outcome, and source. A demonstration-aware gate admits skills with sufficient student mastery or successful teacher demonstrations. Periodic consolidation trains a low-rank adapter on valid action traces with ETG context, incorporating replay when enabled. A verification gate compares the candidate without episodic retrieval against the base model with retrieval on reserved probes. Release requires valid training participation and scores meeting both a relative threshold and an absolute floor. Approved skills use verified weights without episodic retrieval; other skills retain external guidance, optionally with a separately approved adapter. Inspired by complementary learning systems (McClelland et al., 1995), this process connects experience acquisition with parameter learning and selective memory release. Experiments across interactive and reasoning benchmarks show ENGRAMoutperforms most of baselines on multiple benchmarks. Ablations support the contributions of demonstration-aware admission and adapter activation, while showing that external guidance remains useful after training. A separate analysis illustrates how skill-level verification reduces retrievable experience through selective archival. Specifically, our contributions are:

1. **Demonstration-aware training admission.** A per-skill rule admits useful teacher demonstrations independently of current student mastery, expanding the experience available for parameter learning.
2. **Verification before memory release.** A separate gate requires training participation and behavioral evidence from reserved probes before removing records from retrieval.
3. **Performance and efficiency gains.** Benchmark evaluations and ablations demonstrate improved task performance, while serving measurements show reduced token usage and model calls in sequential interaction and lower latency in QA relative to Base+ETG. Skill-level analysis additionally demonstrates selective reduction of retrievable experience.

2 RELATED WORK

Self-evolution through policy learning. STaR and ReST^{EM} improve model parameters by generating, filtering, and learning from the model's own solutions (Zelikman et al., 2022; Singh et al., 2023). Other methods train the behaviors that support iterative improvement: RISE learns multi-turn revision after unsuccessful attempts, while SCoRe learns self-correction through online reinforcement learning (Qu et al., 2024; Kumar et al., 2024). SEAL generates self-edits whose supervised updates

Table 1: **Mechanisms for agent evolution.** W: updates to the acting LLM's weights. M: persistent external experience, skills, or prompts retrieved during training or deployment. A dash indicates that the mechanism is not part of the reported design.

Method	W	M	Adaptation mechanism
RISE (2024)	✓	–	Learning multi-turn revision
Agent-Pro (2024)	–	✓	Updating beliefs and behavioral policies
SEAL (2025)	✓	–	Generating self-edits for parameter updates
EvolveR (2025)	✓	✓	Distilling strategic principles and learning the policy
AgentEvolver (2025)	✓	✓	Generating tasks and reusing experience for policy learning
MAGE (2026)	–	✓	Updating skills, graph memory, and strategies
MemRL (2026b)	–	✓	Learning utilities for episodic retrieval
MemSkill (2026a)	–	✓	Refining memory routines and learning their selector[†]
SkillRL (2026)	✓	✓	Joint policy learning and skill-library refinement
EDGE (2026)	✓	✓	Gain-gated distillation and utility-based experience pruning
Memento-Skills (2026)	–	✓	Developing reusable skills and updating stateful prompts[†]
EvolveMem (2026)	–	✓	Adapting memory and retrieval configurations
ENGRAM	✓	✓	**Demonstration-aware training and verified memory release**

[†]Training a memory controller or skill router does not count as updating the acting LLM. M excludes task-local blackboards, archives used only for gradient updates, and fixed factual corpora.

improve downstream performance (Zweiger et al., 2025). Agent0 extends policy learning through the co-evolution of a task-generating curriculum agent and a tool-using executor (Xia et al., 2025). These methods provide mechanisms for improving an agent's capabilities through parameter updates.

Self-evolution through external memory. External experience also supports adaptation without updating the acting model's weights. Reflexion stores verbal feedback, Voyager accumulates executable skills, and Agent-Pro refines an explicit behavioral policy through reflection and search (Shinn et al., 2023; Wang et al., 2023; Zhang et al., 2024). Memento uses case-based memory, while MAGE organizes skills, successful traces, and teacher corrections in a co-evolving graph (Zhou et al., 2025; Yang et al., 2026). Recent work also adapts how memory is managed: MemRL learns retrieval utilities from environmental feedback; MemSkill learns a memory-skill selector and revises memory routines; and EvolveMem adjusts retrieval configurations through failure diagnosis and regression checks (Zhang et al., 2026b;a; Liu et al., 2026). Memento-Skills develops reusable skills and stateful prompts through a read–write learning loop (Zhou et al., 2026). These approaches illustrate how agents can improve by changing the experience available to them and how it is used.

Joint evolution of weights and memory. Several methods combine external experience with policy learning. EvolveR couples distilled principles with reinforcement learning, and SkillRL jointly develops a hierarchical skill library and the agent policy (Wu et al., 2025; Xia et al., 2026). AgentEvolver reuses an experience pool during policy learning, while EDGE combines gain-gated distillation with utility-based experience pruning (Zhai et al., 2025; Xie et al., 2026). This combination also relates to complementary learning systems, which distinguish rapid experience storage from gradual integration into distributed representations (McClelland et al., 1995). Within this setting, ENGRAM focuses on two decisions: when stored experience should enter training, and when its external records can be removed. Demonstration-aware admission allows learning beyond current student mastery; a separate verification gate requires training participation and compares the updated model without episodic retrieval against the base model with retrieval before authorizing removal.

3 PROBLEM FORMULATION

Consider an agent interacting with an environment over rounds $t = 1, \ldots, T$. At step j, the observation o_j and interaction history h_j form the task input $x_j = P(o_j, h_j)$. The policy generates an action from $\pi_{\theta_t}(\cdot \mid x_j, c_j)$, where $c_j = \mathrm{Retrieve}(x_j, \mathcal{M}_t)$ supplies episodic context from external memory. Without episodic retrieval, $c_j = \varnothing$; task instructions, observations, history, and task-provided evidence remain in x_j.

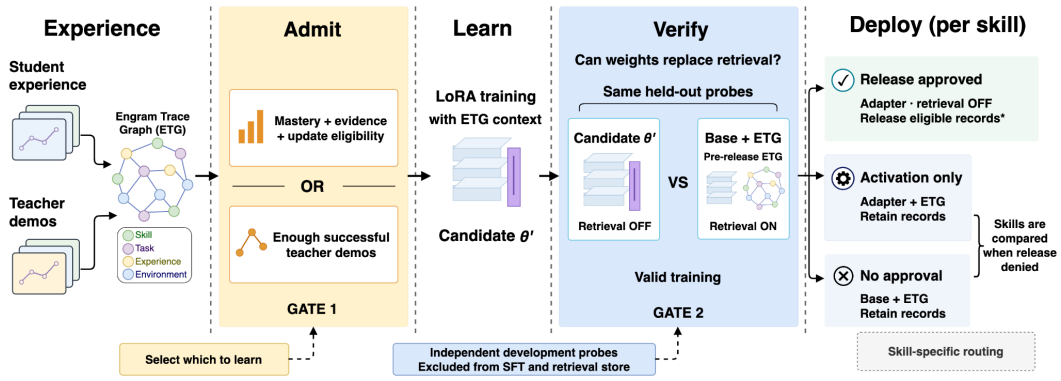

Figure 2: **The ENGRAM consolidation cycle.** Step 1: Student experience and teacher demonstrations enter ETG. Step 2: The admission gate selects skills for training. Step 3: Model fine-tuning based on ETG context. Step 4: The release gate compares the updated model without episodic retrieval against the base model with retrieval. Step 5: Engram deployment. Passing skills release eligible records; other skills retain external support.

Persistent state. The agent maintains policy parameters, an Engram Trace Graph (ETG), and a training replay archive $\mathcal{R}_t$. The ETG associates episodic records $\mathcal{M}_t$ with skills $\mathcal{S}$ and their prerequisites. Each record stores a trajectory, outcome, and source. For each skill s, $m_{t,s} \in [0, 1]$ estimates student mastery from evaluation. Successful teacher demonstrations provide training evidence without increasing this estimate. Replay supports later training and is excluded from deployment retrieval.

Two decisions per skill. Training admission $u_{t,s} \in \{0, 1\}$ determines whether a skill enters the next update. Memory release $z_{t,s} \in \{0, 1\}$ determines whether its external records may be removed afterward. Admission requires useful training evidence; release additionally requires post-training verification. Records that are not released remain available for retrieval and future learning. We evaluate task performance, retrieval and replay storage, and acquisition cost; detailed accounting appears in Appendix E.

4 ENGRAM

ENGRAM alternates experience acquisition with periodic consolidation. Each consolidation round selects skills for training, learns from their experience, and verifies whether their external records can be released. Figure 2 summarizes this cycle, meanwhile Algorithm 1 provides the full procedure.

4.1 DEMONSTRATION-AWARE ADMISSION

The agent collects student experience and requests teacher demonstrations for low-mastery skills. The teacher acts through the same environment interface, and the observed target outcome determines demonstration success. ETG associates each record with its skill, outcome, and source. These associations determine both which examples enter training and which records a release decision may remove. Appendix A details trajectory construction.

ETG records. Let $G_t = (V_t, E_t)$ denote ETG, with typed skill, task, experience, and environment nodes. Skill-prerequisite edges encode dependencies; each experience carries a skill association in its metadata. We write a record as

$$e_i = (s_i, c_i, \tau_i, y_i, p_i), \qquad \mathcal{M}_t(s) = \{e_i : s_i = s\}, \tag{1}$$

where s_i is the associated skill, c_i the task and applicability conditions, τ_i the action trace or solution, y_i the observed outcome, and p_i the source and episode provenance. Successful teacher records in $\mathcal{M}_t(s)$ supply the demonstration count used for admission. Valid state–action pairs from τ_i supply training examples and reserved probes; provenance identifies records from the same acquisition

Algorithm 1 ENGRAM: demonstration-aware consolidation with verified eviction

Require: Base policy θ_0, skills $\mathcal{S}$, cadence K, thresholds $(\tau_m, d_{\min}, \delta, \alpha)$
Initialize training state $\theta \leftarrow \theta_0$, per-skill approvals V, store $\mathcal{M}$, replay $\mathcal{R}$
for evolution round $t = 1, \ldots, T$ **do**
  Collect student experience; update skill mastery
  Acquire successful teacher demonstrations for selected low-mastery skills
  Add valid experience to $\mathcal{M}$ with skill, outcome, and source metadata
  if t is a consolidation round **then**
    Select ready skills S_t using Eq. (2)
    Reserve probes $\{\mathcal{V}_{t,s}\}$; construct $\mathcal{D}_t$ and participation indicators $b_{t,s}$
    if $\mathcal{D}_t \neq \varnothing$ **then**
      Train on $\mathcal{D}_t$ and replay; save and load candidate θ'
      if training and loading succeed **then**
        Compare weights and retrieval; compute $z_{t,s}$ using Eq. (6)
        if at least one skill passes **then**
          Set $V(s) \leftarrow \theta'$ for each passing skill; retain other approvals
          Evict passing skills' records down to the configured residual
        end if
        For rejected skills, optionally approve adapter-plus-memory by a paired probe
        Retain their records; update training-only replay
      end if
    end if
  end if
end for
return Skill-routed policy (θ_0, V), residual store $\mathcal{M}$, decision log

episode. The skill association also determines which records a release decision may remove. Figure 3 illustrates two ETG records: The WebShop record links an option-selection skill to a successful purchase trace; the HotpotQA one links bridge reasoning to a supported answer. Their different verification histories illustrate why successful acquisition and memory release are separate events.

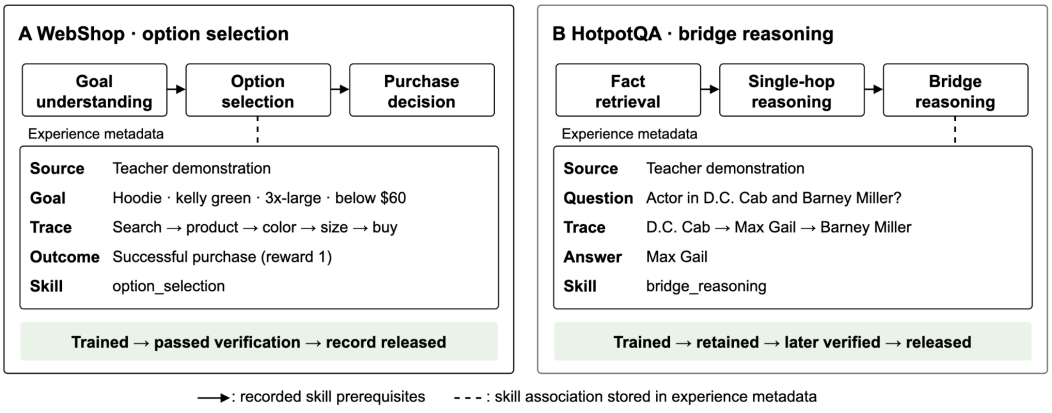

Figure 3: **Two recorded ETG experiences.** Each record links a task, teacher trace, outcome, and skill. Solid arrows show recorded skill prerequisites; dashed lines show the skill association in experience metadata. The examples follow different consolidation paths: immediate verified release (left) and retention until a later update passes (right). Appendix I traces both records through training and deployment.

For skill s, let $n_{t,s}$ count stored success and failure records, and let $d_{t,s}$ count successful teacher-demonstration records. The mastery branch requires sufficient mastery, sufficient records, and eligibility for another update. We denote eligibility by $r_{t,s}$: the skill has not previously been consolidated, or its mastery has increased by a configured amount since its previous consolidation.

Admission is

$$u_{t,s} = \mathbb{I}\big[\big(m_{t,s} \geq \tau_m \wedge n_{t,s} \geq n_{\min} \wedge r_{t,s}\big) \vee d_{t,s} \geq d_{\min}\big]\,. \tag{2}$$

The demonstration branch allows training even when $m_{t,s} = 0$. It therefore admits useful supervision that a mastery-only rule would exclude. Here, $d_{t,s}$ counts stored records rather than independent teacher episodes.

Admission does not guarantee usable training data. Before optimization, we reserve verification records and construct the valid training examples $\mathcal{D}_{t,s}$ for each admitted skill. The participation indicator $b_{t,s} = \mathbb{I}[|\mathcal{D}_{t,s}| > 0]$ prevents memory release for a skill that contributes no valid examples. Successful traces supply action targets; failure corrections are included only when they provide a valid action aligned with an observation.

4.2 Learning from experience

Each valid trajectory step supplies a task input $x = P(o, h)$, its ETG context c, and the target action a. Training uses the same observation format and action interface as deployment. In the ETG-conditioned training setting, c contains the external guidance included in the training prompt. The resulting update is not assumed to make retrieval unnecessary; the release gate tests that condition separately. The student trains a LoRA adapter on examples from admitted skills together with replay from earlier updates. Let $\widetilde{\mathcal{D}}_t$ denote this combined training set. The supervised objective is

$$\mathcal{L}_{\mathrm{SFT}}(\theta) = -\mathbb{E}_{(x,c,a)\sim\widetilde{\mathcal{D}}_t}\left[\log \pi_\theta(a \mid x, c)\right], \tag{3}$$

where the log-likelihood sums over target action tokens. Replay construction, optional regularization, and adapter update settings are specified in Appendix A. The trained candidate θ' is loaded before verification. Training or loading failure leaves records and existing deployment approvals unchanged.

4.3 Verification and memory release

For each admitted skill, reserved probes $\mathcal{V}_{t,s} = \{(x_i, a_i)\}_{i=1}^{N_{t,s}}$ provide targets from successful trajectories. We evaluate two conditions on the same task inputs: the candidate θ' without episodic retrieval, and the base model θ_0 with retrieval from the pre-release store. Let $c_i^M = \mathrm{Retrieve}(x_i, \mathcal{M}_t)$. Their probe scores are

$$\widehat{A}^{\mathrm{W}}_{t,s} = \frac{1}{N_{t,s}} \sum_i g_s(\widehat{a}_i(\theta', \varnothing), a_i)\,, \tag{4}$$

$$\widehat{A}^{\mathrm{M}}_{t,s} = \frac{1}{N_{t,s}} \sum_i g_s\big(\widehat{a}_i(\theta_0, c_i^M), a_i\big)\,, \tag{5}$$

where $\widehat{a}_i(\theta, c)$ is the prediction for x_i, and g_s is a binary criterion shared by both conditions: parsed-action agreement for sequential tasks or thresholded answer correctness for QA. Memory release requires valid training participation, nonempty probes, and both relative and absolute performance conditions:

$$\begin{aligned} z_{t,s} = u_{t,s}\, b_{t,s}\, \mathbb{I}[N_{t,s} > 0] \\ \cdot \mathbb{I}[\widehat{A}^{\mathrm{W}}_{t,s} - \widehat{A}^{\mathrm{M}}_{t,s} \geq \delta] \cdot \mathbb{I}[\widehat{A}^{\mathrm{W}}_{t,s} > \alpha]. \end{aligned} \tag{6}$$

The margin δ specifies the required score difference; negative values allow a bounded decrease relative to retrieval. The strict floor α requires a minimum level of behavior beyond the relative comparison. Skills with no probes remain ineligible for release.

These probes measure behavior on reserved states, not complete task success. In sequential environments, multiple actions can be valid, and errors can change later states. We therefore evaluate complete rollouts separately. Probe construction, data separation, and benchmark-specific scoring are described in Appendix E.

Deployment and retained memory. Skills approved for release use their approved adapter without episodic retrieval; eligible records leave the retrieval store subject to the residual-memory allowance. Otherwise, a separate activation probe may approve the candidate with ETG if it outperforms Base+ETG and exceeds the behavioral floor. Activation alone retains all records and does not authorize release. Failed candidates leave prior approvals intact; skills without an approved adapter

use Base+ETG. The router maps each request or interaction state to the ETG skill taxonomy using only deployment-visible inputs, without gold answers, future outcomes, or held-out labels. The controller then applies the latest approved configuration for that skill, including its adapter and retrieval status.

5 EXPERIMENTS

5.1 BENCHMARKS AND EVALUATION SETUP

We evaluate sequential decision-making on Crafter, ALFWorld, and WebShop, and question answering on HotpotQA, WebQuestions, and FinQA (Hafner, 2021; Shridhar et al., 2021; Yao et al., 2022a; Yang et al., 2018; Berant et al., 2013; Chen et al., 2021). Metrics are achievement progress for Crafter, success rate for ALFWorld and WebShop, answer EM/F1 for HotpotQA and WebQuestions, and numeric-answer accuracy for FinQA. ENGRAM trains LoRA adapters on admitted experience with ETG context (Hu et al., 2021). Appendix B specifies models, data splits, and training settings. Baseline methods, their source references, and implementation details are provided in Appendix D.

5.2 TASK PERFORMANCE

Table 2 reports the progress scores for ENGRAM on Crafter, ALFWorld, HotpotQA, and WebQuestions using Qwen3-8b. It ranks second on WebShop and FinQA, respectively 0.6 and 1.6 percentage points below the highest listed means. These results cover interactive tasks and textual and numerical reasoning. Appendix C futher proves ENGRAM's robustness using Llama-3.1-8b-Instruct, with results shown in Table 6.

Table 2: **Benchmark performance (%).** Green and yellow indicate the highest and second-highest reported means under the source-specific settings in Appendix D. SR: success rate; EM: exact match; Acc.: numeric-answer accuracy.

(a) Achievement-based exploration

Task / Metric	BALROG	MAGE	**ENGRAM**
Crafter / Progress / Peak Prog.	25.5 ± 3.2	33.5 ± 2.3	$\mathbf{34.2 \pm 1.4}$

(b) Interaction (SR)

Method	ALFWorld	WebShop
ReAct	57.0 ± 3.0	40.0 ± 2.4
Reflexion	42.7 ± 4.0	28.8 ± 3.0
Mem0	33.6 ± 3.5	2.00 ± 0.60
MemRL	21.4 ± 3.0	9.20 ± 1.40
EvolveR	73.8 ± 2.5	17.6 ± 2.0
GRPO	77.6 ± 2.2	66.1 ± 2.0
SkillRL	89.9 ± 1.8	72.7 ± 1.8
EDGE	85.9 ± 2.0	82.6 ± 3.8
MAGE	87.3 ± 1.8	$\mathbf{84.0 \pm 1.8}$
ENGRAM	$\mathbf{90.3 \pm 1.5}$	83.4 ± 2.4

(c) Question answering

	HotpotQA		WebQuestions	
Method	EM	F1	EM	F1
CoT	27.0 ± 0.8	39.8 ± 0.9	28.8 ± 1.0	37.8 ± 1.1
ReAct	27.6 ± 1.2	41.3 ± 1.3	27.4 ± 1.3	36.3 ± 1.4
EvolveR	40.3 ± 1.0	54.6 ± 1.1	36.0 ± 1.2	49.8 ± 1.3
SuRe	38.8 ± 0.9	53.5 ± 1.0	34.4 ± 1.1	48.5 ± 1.2
RPG	37.9 ± 1.1	49.2 ± 1.2	34.2 ± 1.2	47.3 ± 1.3
EmbQA	42.0 ± 0.8	55.8 ± 0.9	38.2 ± 1.0	52.1 ± 1.1
RoleRAG	37.4 ± 1.0	49.2 ± 1.1	32.7 ± 1.2	44.1 ± 1.3
ENGRAM	$\mathbf{54.6 \pm 0.9}$	$\mathbf{71.5 \pm 1.0}$	$\mathbf{40.4 \pm 1.0}$	$\mathbf{53.7 \pm 1.1}$

(d) Financial reasoning (Acc.)

Task	CoT	MAGE	FinQANet	APOLLO	**ENGRAM**
FinQA	57.5 ± 2.1	67.5 ± 1.7	62.2 ± 1.5	$\mathbf{70.9 \pm 1.2}$	69.3 ± 1.7

5.3 ABLATION STUDY

Parameter learning and admission. The full method improves over ETG with frozen weights by 3.1–8.4 percentage points across all six benchmarks. SFT without inference-time ETG also exceeds this baseline throughout, indicating that training provides benefits beyond retrieved guidance. Mastery-only admission consistently reduces performance, with the largest decrease on ALFWorld.

This supports admitting successful teacher demonstrations before the student reliably performs the associated skill.

Activation and external guidance. Removing the activation gate lowers scores on every benchmark. Disabling inference-time ETG after SFT produces larger decreases, particularly on ALFWorld and WebShop. The results support combining learned behavior with continued access to external experience: useful parameter updates do not make retrieval unnecessary for every skill.

Table 3: **Ablation results.** Matched traces and task splits; metrics (%) are Crafter Progress, ALFWorld/WebShop success, HotpotQA/WebQuestions F1, and FinQA numeric accuracy.

Variant	Crafter	ALFWorld	WebShop	HotpotQA	WebQ	FinQA
Full ENGRAM	**34.2**	**90.3**	**83.4**	**71.5**	**53.7**	**69.3**
Mastery-only admission	30.5	85.8	79.6	68.3	51.8	66.6
ETG with frozen weights	26.9	82.1	75.0	64.2	50.6	64.0
Without activation gate	32.7	87.3	80.8	70.1	52.6	68.1
SFT, no inference ETG	30.8	85.1	78.6	68.0	51.5	66.9

5.4 SKILL VERIFICATION AND MEMORY RELEASE

Development-set verification. Figure 4 examines a fixed Qwen3-8B SFT checkpoint with 600 HotpotQA experience records: 16 original records and 584 added from official SFT targets.n indicating the number of questions per skill. Release requires a score strictly above 50%, together with the relative-performance and training-evidence checks.

Three of five skills qualify for release. Complex question parsing retains retrieval because the candidate scores below Base+ETG. Multi-answer handling improves over this reference but fails the absolute performance floor. These contrasting cases illustrate the complementary roles of relative and absolute verification. The resulting decisions archive 331 records and leave 269 retrievable, reducing the retrievable footprint by 55.2%.

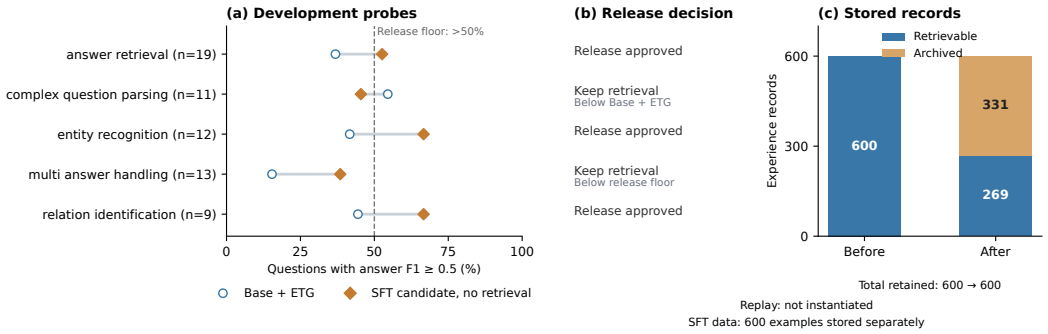

Figure 4: **Skill-level verification and memory release.**

Held-out evaluation. We compare four release policies using the same checkpoint, initial experience store, and held-out questions: retain all, immediate release after training, verified release, and random release at a matched memory budget. We hold the adapter, prompts, retrieval configuration, and decoding settings fixed and freeze release decisions before test evaluation. This comparison isolates release-policy effects from parameter learning and adapter activation. Appendix E.1 provides the detailed protocol and uncertainty estimation.

Table 4 shows that verified release remains close to retain-all performance while making fewer than half of the original records retrievable. Its paired F1 difference is -0.2 points, with a 95% confidence interval of $[-0.6, 0.2]$. This interval is consistent with a small performance change in either direction, rather than establishing that release causes no loss.

Immediate release leaves only 13 retrievable records but lowers F1 from 71.1 to 66.8, indicating that training participation alone is insufficient to justify aggressive release in this setting. At the same retrievable record count, verified release outperforms random release by 3.3 F1 points. EM follows

Table 4: **Held-out performance under alternative memory-release policies.** All conditions use the same HotpotQA checkpoint, initial ETG store, and test questions. EM and F1 are percentages. ΔF1 is the paired difference from retaining all records with a 95% confidence interval.

	Held-out performance			ETG records	
Release policy	EM	F1	ΔF1 [95% CI]	Retrievable	Archived
Retain all	54.4	71.1	Reference	600	0
Immediate release	50.9	66.8	-4.3 $[-4.6, -4.0]$	13	587
Random release	51.6	67.6	-3.5 $[-3.9, -3.1]$	269	331
Verified release	54.0	70.9	-0.2 $[-0.6, 0.2]$	269	331

the same ordering. These results support verification as a means of reducing retrieval dependence while limiting the observed performance cost. All released records remain archived, so the reduction concerns retrieval availability rather than total storage.

5.5 Latency and serving cost

Table 5: **Latency and serving work.** Means per task on paired ALFWorld and WebQuestions using Qwen3-8B. **Green** and yellow : highest and second-highest values. Lower is better for time, calls, and token counts; higher is better for SR/F1. Input/output tokens are in thousands. Reflexion includes up to three attempts and reflection calls. Dashes denote pending supplementary measurements.

Benchmark	Method	Time (s)	Calls	Input (k)	Output (k)	SR / F1 (%; Δ pp)
ALFWorld	ReAct-style	50.27	36.50	73.64	1.95	57.0(-33.3)
	Reflexion	167.86	104.33	252.86	6.92	42.7(-47.6)
	Base + ETG	**43.76**	29.50	81.24	1.66	82.1 (-8.2)
	SFT, no inference ETG	60.53	**22.58**	**35.76**	**1.30**	85.1 (-5.2)
	ENGRAM	65.91	25.50	46.50	1.51	**90.3 $(+0.0)$**
WebQuestions	CoT	1.56	**1.00**	1.74	0.07	37.8(-15.9)
	Self-critique (3 calls)	8.83	3.00	5.61	0.43	40.6(-13.1)
	Base + ETG	1.67	**1.00**	1.78	0.07	50.6(-3.1)
	SFT, no inference ETG	**0.86**	**1.00**	**1.74**	**0.02**	51.5 (-2.2)
	ENGRAM	1.26	**1.00**	1.75	0.04	**53.7 $(+0.0)$**

On ALFWorld, ENGRAM uses 42.8% fewer input tokens and 13.6% fewer calls than Base+ETG, but takes longer overall. SFT without inference-time ETG uses still fewer resources, while ENGRAM improves success by 5.2 percentage points. Thus, the full controller trades additional inference work for higher task performance. On WebQuestions, ENGRAM is 24.6% faster than Base+ETG and improves F1 by 3.1 points. SFT without inference-time ETG is the fastest configuration, but scores 2.2 F1 points below ENGRAM. Together, these results show that ENGRAM achieves the strongest task performance among the measured configurations, while the latency trade-off depends on the benchmark and comparison method.

6 Conclusion

We introduced ENGRAM, a framework that separates training admission from verified memory release. Demonstration-aware admission enables learning beyond current student mastery, while verification determines which external records can leave retrieval. Experiments across two backbones support the framework's effectiveness, with ablations demonstrating the benefits of parameter learning, demonstration-aware admission, and selective adapter activation. A controlled held-out comparison on HotpotQA shows that verified release retains performance close to keeping all records and outperforms immediate and budget-matched random release, supporting verification as a basis for reducing retrieval dependence. Serving evaluations further show reduced input-token usage and model calls on ALFWorld and lower latency on WebQuestions relative to Base+ETG. These findings support combining parameter learning with selective memory release to improve task performance and reduce retrieval and serving costs.

THE USE OF LARGE LANGUAGE MODELS (LLMS)

We used AI-assisted tools to revise wording, improve readability, and assist with LaTeX formatting. The authors reviewed and revised the resulting content and take responsibility for the final manuscript. Language models used as experimental agents and teachers are described in the methodology and experimental setup.

ETHICS STATEMENT

Consolidating agent experience into model parameters can preserve sensitive information, erroneous behavior, or biases present in the training traces. Our release verification evaluates task behavior; it does not establish privacy protection, factual reliability, or safety. Removing records from retrieval also does not erase information encoded in model weights, and archived records remain stored. Applications involving sensitive data or consequential actions therefore require additional data governance, safety evaluation, and human oversight.

REPRODUCIBILITY STATEMENT

Section 4 describes the training-admission, consolidation, and memory-release procedures. Appendix B defines the evaluation metrics and resource accounting. Appendix D describes the baseline implementations, while Appendix E details data partitioning, verification conditions, and uncertainty estimation.

REFERENCES

BALROG. BALROG LLM Leaderboard, 2026. URL https://balrogai.com/. Accessed 18 September 2026; leaderboard entry dated 11 November 2024.

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. Semantic parsing on Freebase from question-answer pairs. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pp. 1533–1544. Association for Computational Linguistics, 2013. URL https://aclanthology.org/D13-1160/.

Zhiyu Chen, Wenhu Chen, Charese Smiley, Sameena Shah, Iana Borova, Dylan Langdon, Reema Moussa, Matt Beane, Ting-Hao Huang, Bryan Routledge, and William Yang Wang. FinQA: A Dataset of Numerical Reasoning over Financial Data, 2021. URL https://arxiv.org/abs/2109.00122.

Prateek Chhikara, Dev Khant, Saket Aryan, Taranjeet Singh, and Deshraj Yadav. Mem0: Building production-ready ai agents with scalable long-term memory. *arXiv preprint arXiv:2504.19413*, 2025. URL https://arxiv.org/pdf/2504.19413.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024. URL https://arxiv.org/abs/2407.21783.

Danijar Hafner. Benchmarking the Spectrum of Agent Capabilities. *arXiv preprint arXiv:2109.06780*, 2021. URL https://arxiv.org/abs/2109.06780.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-Rank Adaptation of Large Language Models. *arXiv preprint arXiv:2106.09685*, 2021. URL https://arxiv.org/abs/2106.09685.

Zhanghao Hu, Hanqi Yan, Qinglin Zhu, Zhenyi Shen, Yulan He, and Lin Gui. Beyond prompting: An efficient embedding framework for open-domain question answering. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 19975–19990, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN

979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.981. URL https://aclanthology.org/2025.acl-long.981/.

Jaehyung Kim, Jaehyun Nam, Sangwoo Mo, Jongjin Park, Sang-Woo Lee, Minjoon Seo, Jung-Woo Ha, and Jinwoo Shin. Sure: Summarizing retrievals using answer candidates for open-domain qa of llms. In *International Conference on Learning Representations*, volume 2024, pp. 15213–15245, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/428ceef2cd8a53add7213e04d1746479-Paper-Conference.pdf.

Aviral Kumar, Vincent Zhuang, Rishabh Agarwal, Yi Su, John D Co-Reyes, Avi Singh, Kate Baumli, Shariq Iqbal, Colton Bishop, Rebecca Roelofs, Lei M Zhang, Kay McKinney, Disha Shrivastava, Cosmin Paduraru, George Tucker, Doina Precup, Feryal Behbahani, and Aleksandra Faust. Training Language Models to Self-Correct via Reinforcement Learning, 2024. URL https://arxiv.org/abs/2409.12917.

Jiaqi Liu, Xinyu Ye, Peng Xia, Zeyu Zheng, Cihang Xie, Mingyu Ding, and Huaxiu Yao. EvolveMem:Self-Evolving Memory Architecture via AutoResearch for LLM Agents, 2026. URL https://arxiv.org/abs/2605.13941.

Yuanjie Lyu, Zihan Niu, Zheyong Xie, Chao Zhang, Tong Xu, Yang Wang, and Enhong Chen. Retrieve-plan-generation: An iterative planning and answering framework for knowledge-intensive llm generation. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 4683–4702, 2024. URL https://aclanthology.org/2024.emnlp-main.270.pdf.

James L. McClelland, Bruce L. McNaughton, and Randall 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. doi: 10.1037/0033-295X.102.3.419.

Yuxiao Qu, Tianjun Zhang, Naman Garg, and Aviral Kumar. Recursive Introspection: Teaching Language Model Agents How to Self-Improve, 2024. URL https://arxiv.org/abs/2407.18219.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024. URL https://arxiv.org/pdf/2402.03300.

Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language Agents with Verbal Reinforcement Learning. *arXiv preprint arXiv:2303.11366*, 2023. URL https://arxiv.org/abs/2303.11366.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. ALFWorld: Aligning text and embodied environments for interactive learning. In *International Conference on Learning Representations*, 2021. URL https://arxiv.org/abs/2010.03768.

Avi Singh, John D. Co-Reyes, Rishabh Agarwal, et al. Beyond Human Data: Scaling Self-Training for Problem-Solving with Language Models. *arXiv preprint arXiv:2312.06585*, 2023. URL https://arxiv.org/abs/2312.06585.

Jiashuo Sun, Hang Zhang, Chen Lin, Xiangdong Su, Yeyun Gong, and Jian Guo. APOLLO: An optimized training approach for long-form numerical reasoning. In Nicoletta Calzolari, Min-Yen Kan, Veronique Hoste, Alessandro Lenci, Sakriani Sakti, and Nianwen Xue (eds.), *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pp. 1370–1382, Torino, Italia, May 2024. ELRA and ICCL. URL https://aclanthology.org/2024.lrec-main.122/.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An Open-Ended Embodied Agent with Large Language Models. *arXiv preprint arXiv:2305.16291*, 2023. URL https://arxiv.org/abs/2305.16291.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022. URL https://papers.neurips.cc/paper_files/paper/2022/file/9d5609613524ecf4f15af0f7b31abca4-Paper-Conference.pdf.

Rong Wu, Xiaoman Wang, Jianbiao Mei, Pinlong Cai, Daocheng Fu, Cheng Yang, Licheng Wen, Xuemeng Yang, Yufan Shen, Yuxin Wang, and Botian Shi. EvolveR: Self-Evolving LLM Agents through an Experience-Driven Lifecycle, 2025. URL https://arxiv.org/abs/2510.16079.

Peng Xia, Kaide Zeng, Jiaqi Liu, Can Qin, Fang Wu, Yiyang Zhou, Caiming Xiong, and Huaxiu Yao. Agent0: Unleashing Self-Evolving Agents from Zero Data via Tool-Integrated Reasoning, 2025. URL https://arxiv.org/abs/2511.16043.

Peng Xia, Jianwen Chen, Hanyang Wang, et al. SkillRL: Evolving Agents via Recursive Skill-Augmented Reinforcement Learning. *arXiv preprint arXiv:2602.08234*, 2026. URL https://arxiv.org/abs/2602.08234.

Can Xie, Yuyi Zhou, Wen Yang, Ziyi Zhang, Siyao Song, Yingzhuo Deng, Shuo Ren, and Jiajun Zhang. EDGE: Experience-Distillation for Guided Exploration in Agentic Reinforcement Learning. *arXiv preprint arXiv:2608.21946*, 2026. URL https://arxiv.org/abs/2608.21946v2.

Ruiyi Yang, Zechen Li, Hao Xue, Imran Razzak, and Flora D. Salim. MAGE: Multi-Agent Self-Evolution with Co-Evolutionary Knowledge Graphs. *arXiv preprint arXiv:2605.10064*, 2026. URL https://arxiv.org/abs/2605.10064.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. HotpotQA: A Dataset for Diverse, Explainable Multi-hop Question Answering. *arXiv preprint arXiv:1809.09600*, 2018. URL https://arxiv.org/abs/1809.09600.

Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. WebShop: Towards Scalable Real-World Web Interaction with Grounded Language Agents. *arXiv preprint arXiv:2207.01206*, 2022a. URL https://arxiv.org/abs/2207.01206.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing Reasoning and Acting in Language Models. *arXiv preprint arXiv:2210.03629*, 2022b. URL https://arxiv.org/abs/2210.03629.

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah D. Goodman. STaR: Bootstrapping Reasoning With Reasoning. *arXiv preprint arXiv:2203.14465*, 2022. URL https://arxiv.org/abs/2203.14465.

Yunpeng Zhai, Shuchang Tao, Cheng Chen, Anni Zou, Ziqian Chen, Qingxu Fu, Shinji Mai, Li Yu, Jiaji Deng, Zouying Cao, Zhaoyang Liu, Bolin Ding, and Jingren Zhou. AgentEvolver: Towards Efficient Self-Evolving Agent System, 2025. URL https://arxiv.org/abs/2511.10395.

Haozhen Zhang, Quanyu Long, Jianzhu Bao, Tao Feng, Weizhi Zhang, Haodong Yue, and Wenya Wang. MemSkill: Learning and Evolving Memory Skills for Self-Evolving Agents. *arXiv preprint arXiv:2602.02474*, 2026a. URL https://arxiv.org/abs/2602.02474.

Shengtao Zhang, Jiaqian Wang, Ruiwen Zhou, Junwei Liao, Yuchen Feng, Zhuo Li, Yujie Zheng, Weinan Zhang, Ying Wen, Zhiyu Li, Feiyu Xiong, Yutao Qi, Bo Tang, and Muning Wen. MemRL: Self-Evolving Agents via Runtime Reinforcement Learning on Episodic Memory, 2026b. URL https://arxiv.org/abs/2601.03192.

Wenqi Zhang, Ke Tang, Hai Wu, Mengna Wang, Yongliang Shen, Guiyang Hou, Zeqi Tan, Peng Li, Yueting Zhuang, and Weiming Lu. Agent-Pro: Learning to Evolve via Policy-Level Reflection and Optimization, 2024. URL https://arxiv.org/abs/2402.17574.

Huichi Zhou, Yihang Chen, Siyuan Guo, Xue Yan, Kin Hei Lee, Zihan Wang, Ka Yiu Lee, Guchun Zhang, Kun Shao, Linyi Yang, and Jun Wang. Memento: Fine-tuning LLM Agents without Fine-tuning LLMs, 2025. URL https://arxiv.org/abs/2508.16153.

Huichi Zhou, Siyuan Guo, Anjie Liu, Zhongwei Yu, Ziqin Gong, Bowen Zhao, Zhixun Chen, Menglong Zhang, Yihang Chen, Jinsong Li, Runyu Yang, Qiangbin Liu, Xinlei Yu, Jianmin Zhou, Na Wang, Chunyang Sun, and Jun Wang. Memento-Skills: Let Agents Design Agents, 2026. URL https://arxiv.org/abs/2603.18743.

Yutao Zhu, Jiajie Jin, Hongjin Qian, Zheng Liu, Zhicheng Dou, and Ji-Rong Wen. Single LLM, multiple roles: A unified retrieval-augmented generation framework using role-specific token optimization. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 4837–4856, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-332-6. doi: 10.18653/v1/2025.emnlp-main.243. URL https://aclanthology.org/2025.emnlp-main.243/.

Adam Zweiger, Jyothish Pari, Han Guo, Ekin Akyürek, Yoon Kim, and Pulkit Agrawal. Self-Adapting Language Models, 2025. URL https://arxiv.org/abs/2506.10943.

A Controller and Training Details

A.1 Consolidation configuration

Algorithm 1 specifies the general multi-round controller. The scaled SFT experiments use a single batch update followed by per-skill verification and deployment; replay from earlier consolidation rounds is empty in this setting. Historical multi-round runs and their configurations are reported separately.

A.2 Trajectory construction

Acquisition targets low-mastery skills. A strong teacher executes action sequences through the same environment interface. The environment's reported target outcome determines demonstration success. For achievement-based environments, the stored segment ends at the action that unlocks the achievement, and a target-directed episode can stop once that event occurs. This alignment retains the action that caused success rather than an arbitrary prefix of the rollout.

Teacher demonstrations complement student successes and teacher-written failure corrections. A correction enters action training only if it supplies a valid action for an aligned observation. Source tags distinguish demonstrations, corrections, and student traces.

A.3 Training configuration

Examples from admitted skills are combined with available training-only replay:

$$\mathcal{D}_t = \bigcup_{s:u_{t,s}=1} \mathcal{D}_{t,s}, \qquad \widetilde{\mathcal{D}}_t = \mathcal{D}_t \cup \mathrm{Sample}(\mathcal{R}_t, \rho). \tag{7}$$

The replay fraction ρ is limited by archive availability. Each example contains task input, training context, and the target action. The scaled SFT runs include ETG context and optimize the action-token likelihood in Eq. (3); their candidates restart from the base model. Optional regularization adds

$$\beta\, \mathbb{E}_{(x,c,a)\sim\widetilde{\mathcal{D}}_t}\, \mathrm{KL}(\pi_{\theta_0}(\cdot \mid x, c) \,\|\, \pi_\theta(\cdot \mid x, c)) \tag{8}$$

at supervised response positions. This optional term is not part of the scaled SFT objective. Candidate loading precedes verification, and failures preserve existing approvals and external records.

Admission and probe settings. The demonstration threshold $d_{\min} = 2$ counts successful records, not independent teacher episodes. QA probes use an answer-score threshold of 0.5; final EM and F1 are reported separately. Probe construction and episode separation follow Appendix E. Only records with valid training participation are eligible for release, with up to k_{res} recent residual records retained per memory category.

Retrieval-store accounting. If A_t records arrive and E_t eligible records are released, retrieval occupancy obeys

$$|\mathcal{M}_{t+1}| = |\mathcal{M}_t| + A_t - E_t. \tag{9}$$

Retrieval storage stabilizes when departures keep pace with arrivals. Replay and any retained archives are accounted for separately; removing retrieval records does not by itself imply a reduction in total storage.

B Experimental Setup

B.1 Metric definitions and uncertainty

Task metrics. For Crafter, let r_a denote the fraction of evaluated worlds in which achievement a is attained. We report arithmetic achievement progress,

$$P_{\mathrm{Crafter}} = \frac{100}{22} \sum_{a=1}^{22} r_a. \tag{10}$$

Peak Progress is the highest progress observed across evolution rounds. The geometric achievement score is a distinct metric,

$$S_{\text{Crafter}} = \exp\left(\frac{1}{22}\sum_{a=1}^{22}\log(1+100r_a)\right) - 1. \tag{11}$$

These statistics are computed from achievement rates across worlds; averaging per-world geometric scores yields a different quantity.

ALFWorld and WebShop use binary task success rate. WebShop's fractional reward, when reported, is listed separately. HotpotQA and WebQuestions use normalized answer exact match (EM) and token F1. Local FinQA evaluations use scale-normalized numeric-answer accuracy with a 1% relative tolerance, which is distinct from program execution accuracy.

Uncertainty. We distinguish standard deviations across repeated runs from bootstrap uncertainty over evaluation instances. The local paired analyses use checkpoints trained with seed 42; their instance-level uncertainty does not measure variation across independently trained models. Paired bootstrap samples use the same resampled instances for all compared methods. For Crafter, we resample worlds with their complete achievement vectors and recompute the aggregate statistic in each draw. Appendix E specifies the resampling procedures, including hierarchical uncertainty where applicable. Published comparisons retain the uncertainty reported by their sources.

B.2 Memory, retention, and serving measurements

Memory accounting. We distinguish retrievable ETG records, non-retrievable archived records, and separately stored training data, including replay when instantiated. For consolidation analyses, we report record counts and available byte measurements before and after the update. Moving records to an archive reduces the retrievable footprint but does not reduce total retained storage. Capability–footprint comparisons therefore identify which stores are included. For hard-cap configurations, rejected admissions and forced deletions are recorded separately from verified release.

Retention. In multi-round evaluations, let t_s be the first round in which skill s is consolidated and $R_{t,s}$ its rollout success at round t. We report final success $R_{T,s}$ and the decline from its best observed post-consolidation success,

$$D_{T,s} = \max_{t_s \le t \le T} R_{t,s} - R_{T,s}. \tag{12}$$

These measurements use fresh evaluation episodes. Action-probe scores and training loss provide diagnostics; they do not replace rollout-based retention measurements.

Acquisition and serving cost. We account separately for student and teacher acquisition, verification probes, and training compute. Serving measurements report elapsed time, model calls, and input and output tokens per task, with retrieval tokens identified where available. Paired comparisons use the same evaluation tasks and distinguish weights-only, retrieval-assisted, and per-skill routed deployment. Token usage and wall-clock latency are reported separately because lower token usage does not necessarily imply faster execution.

Release decisions. A skill is approved for release only after satisfying its verification criterion. Candidate-adapter performance or evaluation with retained ETG alone does not authorize release. Development-probe approval and the resulting record movement are reported separately from held-out task performance after release.

C Additional Experiments on Llama3.1-8b-Insturct

Table 6 shows extra experiments done using Llama3.1-8b Instruct (Grattafiori et al., 2024).

Table 6: **Comparison on Llama-3.1-8B-Instruct (%).** Entries are mean ± standard deviation. Green and yellow indicate the highest and second-highest means. SR: success rate; EM: exact match; Acc.: numeric-answer accuracy.

(a) Achievement-based exploration

Task / Metric	BALROG	MAGE	ENGRAM
Crafter / Progress	20.6 ± 3.4	$\mathbf{28.1 \pm 2.8}$	27.4 ± 2.1

(b) Interaction (SR)

Method	ALFWorld	WebShop
ReAct	25.4 ± 3.5	26.8 ± 2.8
Reflexion	43.3 ± 4.4	32.6 ± 3.4
Mem0	30.6 ± 3.8	19.4 ± 3.0
MemRL	36.6 ± 3.6	25.2 ± 3.1
EvolveR	61.2 ± 3.3	40.6 ± 3.0
GRPO	70.1 ± 3.0	55.8 ± 2.9
SkillRL	80.6 ± 2.5	66.8 ± 2.9
EDGE	78.4 ± 2.8	68.2 ± 3.1
MAGE	79.9 ± 2.6	63.4 ± 2.7
ENGRAM	$\mathbf{83.6 \pm 2.2}$	$\mathbf{70.6 \pm 2.8}$

(c) Question answering

	HotpotQA		WebQuestions	
Method	EM	F1	EM	F1
CoT	33.8 ± 1.0	47.6 ± 1.1	26.4 ± 1.2	35.6 ± 1.3
ReAct	35.6 ± 1.3	49.8 ± 1.4	27.1 ± 1.4	36.8 ± 1.5
EvolveR	43.2 ± 1.2	58.7 ± 1.3	32.8 ± 1.3	45.2 ± 1.4
SuRe	40.9 ± 1.1	55.8 ± 1.2	31.7 ± 1.2	44.1 ± 1.3
RPG	39.7 ± 1.2	53.9 ± 1.3	30.8 ± 1.3	42.9 ± 1.4
EmbQA	45.1 ± 1.0	61.0 ± 1.1	35.5 ± 1.1	48.6 ± 1.2
RoleRAG	44.3 ± 1.2	60.2 ± 1.3	34.1 ± 1.3	46.7 ± 1.4
ENGRAM	$\mathbf{49.2 \pm 1.1}$	$\mathbf{65.8 \pm 1.2}$	$\mathbf{37.2 \pm 1.2}$	$\mathbf{50.4 \pm 1.3}$

(d) Financial reasoning (Acc.)

Task	CoT	MAGE	FinQANet-inspired adaptation	APOLLO-inspired adaptation	ENGRAM
FinQA	51.4 ± 2.5	60.8 ± 2.2	57.1 ± 2.3	63.4 ± 2.1	$\mathbf{64.6 \pm 2.0}$

D BASELINE IMPLEMENTATIONS

All baseline scores in our comparison tables are obtained from our own runs rather than copied from published results. Each table identifies the evaluated backbone. References identify the methods on which our implementations are based.

Baseline methods. Our comparisons include BALROG (BALROG, 2026), ReAct (Yao et al., 2022b), Reflexion (Shinn et al., 2023), Mem0 (Chhikara et al., 2025), MemRL (Zhang et al., 2026b), EvolveR (Wu et al., 2025), GRPO (Shao et al., 2024), SkillRL (Xia et al., 2026), EDGE (Xie et al., 2026), and MAGE (Yang et al., 2026). QA comparisons additionally include chain-of-thought prompting (Wei et al., 2022), SuRe (Kim et al., 2024), RPG (Lyu et al., 2024), EmbQA (Hu et al., 2025), and RoleRAG (Zhu et al., 2025). For financial reasoning, we include adaptations based on FinQANet (Chen et al., 2021) and APOLLO (Sun et al., 2024). These references identify the source methods; the reported scores come from our implementations.

Implementation and adaptation. We evaluate prompting, external-memory, and policy-learning baselines under the benchmark configurations described in Appendix B. Method-specific components, including reflection, memory updates, and policy optimization, are retained in the corresponding implementations. FinQANet-inspired and APOLLO-inspired adaptations are labeled explicitly to distinguish them from the original systems.

Comparison conditions. We distinguish task-level baseline comparisons from component ablations. Baseline comparisons evaluate the implemented methods, including their respective learning and inference procedures. Component ablations use matched experience and task splits to examine individual design choices in ENGRAM. Acquisition, training, and serving costs are accounted for separately; using the same backbone does not imply equal computational cost.

E Evaluation and Verification Protocol

Data partitions and leakage control. We distinguish experience used for acquisition and training, development data used for controller verification, and held-out data used for final evaluation. The intended unit of separation is the originating episode: segments from the same episode should remain in one partition even when they are associated with different skills. For QA, normalized questions are deduplicated across partitions before answers are attached. Question–answer matching provides an additional check rather than replacing question-level separation. The final evaluation pool is excluded from gate tuning, checkpoint selection, and prompt development.

Sequential probe construction. In ALFWorld, the consolidator groups identical observation–action–prompt trajectories across skill labels, preserves the reserved groups across rounds, and excludes trajectories already present in replay. Other sequential environments reserve a deterministic tail of successful memory records per skill after sorting by node identifier. Reserved records are excluded from the current consolidation batch. Observation–action probes include up to 16 recent history entries and are subject to a probe-step cap.

Verification conditions. The candidate arm uses the adapted model without episodic retrieval; the reference arm uses the base model with ETG. Both arms use the same task interface and scoring procedure. The implemented reference arm retrieves from the pre-release store. Removing a record from the training batch does not necessarily remove it from this retrieval store, so this condition is distinct from source-excluded retrieval.

For sequential tasks, probes measure agreement with recorded actions using the same environment prompt builder and action parser. Agreement is a local behavioral diagnostic: it can depend on the action classes represented in the probe and does not capture all valid alternative actions or subsequent environment states.

For the QA release analysis in Figure 4, each question is counted as successful when answer F1 is at least 0.5. The skill-level score is the fraction of successful development questions. Release requires this fraction to exceed 50%, together with the relative-performance and training-evidence checks. These answer-level scores are distinct from next-action agreement.

Statistical unit. Paired comparisons match methods by environment world or task instance. Bootstrap draws resample the same instances for all compared methods and recompute the reported metric. For Crafter, each sampled world retains its complete achievement vector; progress or geometric score is then computed from the resulting achievement rates.

Paired differences are summarized with percentile 95% bootstrap intervals. For evaluations with multiple evolution seeds, seed-level variation is distinguished from instance-level uncertainty, with hierarchical resampling where applicable. Single-checkpoint analyses quantify variation over evaluation instances rather than variation across independently trained models. The fraction of bootstrap draws favoring a method is descriptive and is not reported as a hypothesis-test p-value.

Interaction and compute accounting. We distinguish student acquisition, teacher acquisition, verification, consolidation, and final evaluation costs. Serving measurements include all model calls and tokens used by each method, including reflection and repeated generation. Last-attempt success and best-of-attempt success are different outcomes and are identified separately when reported. Matched-data controls reuse the same acquired experience; this does not imply equal end-to-end acquisition or training cost.

Decision and memory accounting. Release decisions are associated with the evaluated checkpoint and the supporting training and probe evidence. The relevant quantities include student mastery, available teacher demonstrations, realized training examples, probe size, candidate and reference scores, and the configured relative threshold and absolute floor.

Memory accounting distinguishes records available for retrieval, records moved to a non-retrievable archive, and separately retained training data. Counts before and after release are reported for the affected stores. Archival release reduces retrieval availability without necessarily reducing total storage.

LLM-only control. For QA, LLM-only denotes one greedy zero-shot chain-of-thought generation from the unadapted student using the question and the benchmark-supplied context, without episodic retrieval, additional search, or an adapter. Interactive controls with handcrafted strategy hints or action substitutions are identified separately rather than labeled LLM-only.

E.1 Controlled evaluation of memory release

Shared experimental conditions. We use the fixed Qwen3-8B SFT checkpoint and the initial store of 600 HotpotQA experience records analyzed in Figure 4. All conditions share the checkpoint, skill assignments, prompt construction, retrieval settings, and decoding configuration. No additional parameter updates are performed during this comparison. Development probes determine release decisions before held-out evaluation; test results are not used to revise these decisions or their thresholds.

Release-policy controls. We compare the following policies:

- **Retain all.** All experience records remain retrievable, and all skills use the fixed adapted model with ETG.
- **Immediate release.** Skills that contributed valid training examples release their eligible records after training, without post-training behavioral verification.
- **Verified release.** Eligible records are released only for skills that satisfy the training-evidence checks, relative verification threshold, and absolute performance floor.
- **Random release.** Release selections are sampled without using probe scores and matched to the verified policy's retrievable record count. We repeat the random selection to measure variation due to the choice of released memories.

For every policy, skills selected for release use the same adapted model without episodic retrieval. Other skills use that model with the remaining ETG. Adapter activation is fixed rather than independently recomputed for each policy. Record eligibility, shared-record handling, and any configured residual retention are applied consistently across conditions. Released records are moved to a non-retrievable archive; the comparison therefore measures retrieval reduction rather than total-storage reduction.

Held-out performance and uncertainty. We evaluate every condition on the same held-out questions and report aggregate answer EM and token F1. For each release policy, we compute the paired F1 difference relative to retaining all records. Paired bootstrap draws resample test questions using the same indices for both conditions and recompute the score difference, yielding percentile 95% confidence intervals.

For random release, we report performance averaged over the sampled release selections and account for both selection-level and question-level variation through hierarchical bootstrap resampling. These intervals quantify uncertainty for the fixed checkpoint, rather than variation across training seeds.

F Implementation and Reproducibility

Table 7 and 8 listed hyperparameters used for experiments using Qwen3-8b and llama3.1-8b-instruct.

Serving conditions. A verified skill uses its approved adapter with episodic retrieval disabled. Other skills use the base model with retained ETG. Task instructions, current observations, recent interaction history, and shared environment rules remain part of the input in both conditions. Previously approved versions remain available when a later candidate fails verification.

Routing implementation. At inference time, the controller assigns the current request or interaction state to a skill using only deployment-visible information and the same skill taxonomy used during ETG construction. The router does not observe gold answers, episode success, future environment states, verification outcomes, or held-out annotations. Given the resulting skill key, the controller loads the most recent approved adapter version for that skill; if no approved version exists, it executes the Base+ETG route.

Table 7: Benchmark-specific SFT settings for Qwen3-8B in the scaled evaluations. Training examples count decisions for ALFWorld and WebShop, answers for WebQuestions, and decision windows from 32 training worlds for Crafter.

Setting	ALFWorld	WebShop	WebQuestions	Crafter
Maximum training tokens	16,384	8,192	8,192	8,192
Data passes	3	3	3	3
Optimizer updates	5,004	2,149	450	1,281
Training seed	42	42	42	42
Learning rate	2×10^{-5}	2×10^{-5}	2×10^{-5}	2×10^{-5}
LoRA rank / scaling	16 / 32	16 / 32	16 / 32	16 / 32
LoRA dropout	0.05	0.05	0.05	0.05
Microbatch / accumulation	1 / 4	1 / 4	1 / 4	1 / 4
Weight decay	0.01	0.01	0.01	0.01
Warm-up fraction	0.1	0.1	0.1	0.1
Base-model KL coefficient	0	0	0	0
Training weight loading	4 bit	4 bit	4 bit	4 bit
Initial retrieval records	19	43	16	185

Table 8: Benchmark-specific SFT settings for Llama-3.1-8B-Instruct in the scaled evaluations. Training examples count decisions for ALFWorld and WebShop, answers for WebQuestions, and decision windows from 32 training worlds for Crafter.

Setting	ALFWorld	WebShop	WebQuestions	Crafter
Maximum training tokens	16,384	8,192	8,192	8,192
Data passes	3	3	3	3
Optimizer updates	5,004	2,149	450	1,281
Training seed	42	42	42	42
Learning rate	2×10^{-5}	2×10^{-5}	2×10^{-5}	2×10^{-5}
LoRA rank / scaling	16 / 32	16 / 32	16 / 32	16 / 32
LoRA dropout	0.05	0.05	0.05	0.05
Microbatch / accumulation	1 / 4	1 / 4	1 / 4	1 / 4
Weight decay	0.01	0.01	0.01	0.01
Warm-up fraction	0.1	0.1	0.1	0.1
Base-model KL coefficient	0	0	0	0
Training weight loading	4 bit	4 bit	4 bit	4 bit
Initial retrieval records	19	43	16	185

Controller implementation. The consolidator selects skills, constructs training examples and reserved probes, continues the adapter, loads the candidate into the serving engine, and evaluates release eligibility. The environment prompt builder is shared by training and evaluation. Training or loading failure preserves the source records. Skills with no valid training examples or no eligible probe remain external. The sequential gate filters probe states to the corresponding deployment route. Retrieval storage and replay storage are recorded separately.

G LIMITATIONS

Release verification relies on finite development probes, whose coverage may not capture every deployment condition. Our controlled release study evaluates a fixed checkpoint; extending this analysis to longer sequences of consolidation and changing task distributions would clarify the durability of release decisions. Consolidation schedules and verification thresholds are also configured per benchmark. Adapting these settings online is a promising direction for broader deployment.

H ESSENTIAL PROMPT INTERFACES

Training and deployment share the benchmark's task interface. The layouts below retain the task fields and output constraints; benchmark-specific environment rules are omitted for readability.

Task execution. Sequential tasks provide the goal, current observation, valid actions, and recent interaction history. The model returns one executable action. Supervised examples pair this pre-action input with the recorded action; the resulting state supplies outcome evidence.

```
Goal: {task goal}
Observation: {current environment state}
Valid actions: {admissible actions}
Recent history: {observations and executed actions}
Output exactly ONE action on its own line.
```

QA uses the same question and evidence in training, verification, and matched evaluation arms. The context is omitted in the question-only control.

```
Answer the following question based on the provided context. Think
    step by step, then give your final answer after 'Answer:'.

Context: {evidence}
Question: {question}
Let's think step by step:
```

Experience acquisition and retrieval. Teacher acquisition adds the selected skill and its target outcome to the task brief. Successful demonstrations retain their executed actions and observed outcomes. When ETG retrieval is enabled, the task input also contains relevant successful traces or corrections, including their conditions of use. A correction contributes a supervised action only when it has a valid target for an aligned observation.

Verification. The two probe arms receive identical task inputs and targets. The candidate arm uses the updated adapter with episodic retrieval disabled; the reference arm uses the base model with ETG retrieval enabled. Both use the same output parser and benchmark-specific probe criterion. Eq. (6) determines release eligibility from these scores and valid training participation.

I ETG RECORDS AND SUCCESSFUL DEPLOYMENT CASES

We follow two successful deployments from the completed WebShop and HotpotQA evaluations. Each case pairs the held-out result with an acquired ETG record, its training participation, and the skill's verification history. Figure 3 shows the key fields of the actual records.

I.1 WEBSHOP: RETAINED EXPERIENCE AND APPROVED WEIGHTS COMPLETE ONE PURCHASE

Acquisition and consolidation. A teacher successfully purchases a kelly-green hoodie in size 3x-large for less than $60. The ETG associates the trajectory with *option selection* and records its observations, executed actions, teacher source, and reward of 1. The first consolidation update includes this trace among 288 action-training examples. On the reserved option-selection probe, both the candidate without ETG and the base model with ETG match all three eligible target actions. The controller approves the adapter for this skill and releases eligible records, including the illustrated teacher record, while keeping residual experience.

The same update retains query formulation, result selection, and purchase decision. A later candidate receives no new approval; deployment continues using the previously verified version for approved skills. This preserves useful weights and external traces across updates.

Successful held-out execution. A different goal requests a *navy-blue* hoodie in size 3x-large for less than $60. This exact goal is absent from the acquired graph snapshots. Table 9 shows the five executed actions. The initial search uses the base model with retained ETG guidance; the remaining four actions use the approved adapter with episodic retrieval disabled.

Table 9: WebShop successful trajectory in a separate completed run. The search query is shortened for display; product identifiers and subsequent actions are preserved.

Step	Executed action	Policy / experience access
1	`search[...navy blue 3x-large]`	Base + retained ETG
2	`click[b01hqtwl6s]`	Approved adapter; ETG off
3	`click[navy blue]`	Approved adapter; ETG off
4	`click[3x-large]`	Approved adapter; ETG off
5	`click[buy now]`	Approved adapter; ETG off

The environment awards full purchase reward after five actions. Under the same goal and interface, Base repeatedly searches and inspects unsuitable products, exhausting 30 actions with reward 0. This separate 100-goal evaluation obtains 47 successes with ENGRAM and 44 with Base: 14 tasks change from failure to success, and 11 change in the opposite direction. This case exhibits the combined deployment path, with external guidance supporting the query and verified weights handling the subsequent interaction.

I.2 HOTPOTQA: A RETAINED BRIDGE TRACE IS LATER RELEASED AFTER VERIFICATION

An ETG record with a reusable reasoning pattern. A successful teacher example asks which actor appeared in both *D.C. Cab* and *Barney Miller*. The stored trace identifies Max Gail in the film cast and connects him to his television role. The record carries the teacher source tag, supplied evidence, answer, and *bridge reasoning* skill association.

The record contributes valid training data to both consolidation updates. After the first update, the weights arm passes 2 of 5 reserved bridge-reasoning probes and the memory arm passes 4 of 5; the record remains in ETG. After the second update, the scores become 3 of 5 and 2 of 5. The controller approves that skill's new adapter version and releases the illustrated record. Replay remains available for later training. Thus the record's release follows verified behavior after a later update, rather than the completion of its first training exposure.

Successful reasoning without episodic retrieval. The held-out question asks:

> In which University is Richard Williams' former assistant a member of the Athletics Hall of Fame?

Both models receive the same ten supplied paragraphs. The relevant evidence identifies Rick Stansbury as Williams' former assistant and states that Stansbury belongs to the Campbellsville University Athletics Hall of Fame. The base model answers *Mississippi State*, the university where Williams coached. The approved adapter answers *Campbellsville University*, matching the reference answer.

The question is absent from the acquired ETG and reserved-probe records. The evaluated request uses the second approved adapter with no episodic retrieval. Its response follows the intermediate person to the requested affiliation, while Base selects the university associated with the first named person. The acquisition trace and this evaluation question involve different entities but share the bridge-reasoning skill.