MEMQ: INTEGRATING Q-LEARNING INTO SELF-EVOLVING MEMORY AGENTS OVER PROVENANCE DAGS

Anonymous authors
Paper under double-blind review

ABSTRACT

Episodic memory allows LLM agents to accumulate and retrieve experience, but current methods treat each memory independently, i.e., evaluating retrieval quality in isolation without accounting for the dependency chains through which memories enable the creation of future memories. We introduce **MemQ**, which applies TD(λ) eligibility traces to memory Q-values, propagating credit backward through a *provenance DAG* that records which memories were retrieved when each new memory was created. Credit weight decays as $(\gamma\lambda)^d$ with DAG depth d, replacing temporal distance with structural proximity. We formalize this setting as an *Exogenous-Context MDP*, separating externally supplied tasks from the evolving memory store, and represent retrieval-set values by a first-order mean of per-memory values plus an explicit approximation residual. Across five interaction, code-generation, and agentic benchmarks, MemQ improves average held-out, final runtime, and cumulative success rates over the baseline with the highest average for each metric by 2.53, 4.23, and 2.13 percentage points, respectively. Ablations on LiveCodeBench, BFCL, and DSBench support the benefit of bootstrapping and multi-depth credit propagation. Analyses of Q-values and runtime learning show that learned memory values are associated with subsequent success and that MemQ retains more of its historical task coverage. We also explore the limits of MemQ's benefits using multiple-choice reasoning tasks, where limited interaction offers fewer opportunities to reuse procedural experience.

1 INTRODUCTION

Large language models (LLMs) deployed as agents must adapt to novel tasks and changing environments, but updating their weights can be costly. External *episodic memory* offers a complementary route: agents accumulate successes, failures, and discovered strategies (Park et al., 2023; Packer et al., 2024; Zhong et al., 2024; Sumers et al., 2024), then retrieve relevant experience to guide future behavior (Shinn et al., 2023; Zhao et al., 2024; Wang et al., 2024). Early systems score memories with fixed heuristics, typically embedding similarity, without learning memory values from task outcomes.

RL-based methods learn which memories to retrieve. Parametric approaches train retrieval policies (Yan et al., 2026; Zhang et al., 2026e) and broader memory operations (Ma et al., 2026a; Shen et al., 2026; Zhang et al., 2026c); non-parametric approaches attach value estimates to memory entries (Pritzel et al., 2017). Most closely related, MemRL (Zhang et al., 2026b) updates each entry's Q-value by single-step exponential moving average (EMA), equivalent to a contextual-bandit update with $\gamma = 0$. Retrieval quality is learned without changing LLM weights, but feedback remains local to the current retrieval event.

The unresolved gap is *multi-step credit assignment across memory-creation events*. Retrieved memories help produce new memories that are later reused, forming exposure chains such as $m_a \to m_b \to m_c \to r$. Single-step updates give earlier ancestors no outcome signal unless they are also retrieved. Classical eligibility traces propagate TD errors across multiple steps (Sutton, 1988; Singh & Sutton, 1996; Sutton & Barto, 2018); off-policy variants truncate or correct this credit (Watkins, 1989; Peng & Williams, 1996; Munos et al., 2016). True-online traces, generalized advantage estimation, and distributed V-trace instantiate related bias–variance tradeoffs (Seijen &

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.

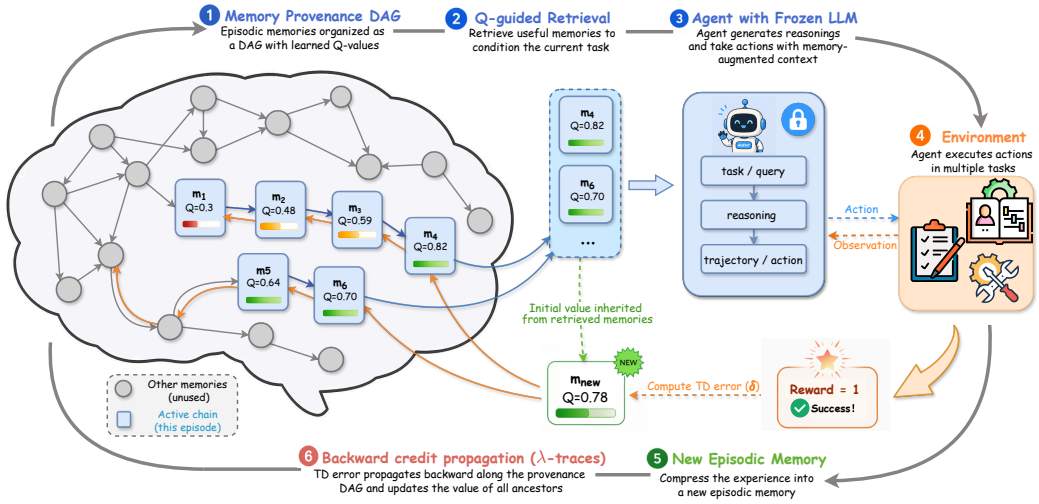


Figure 1: Conceptual overview of MemQ. Retrieved memories support a new memory, and later outcomes send structurally decayed feedback to exposed ancestors in the resulting provenance DAG.

Sutton, 2014; Schulman et al., 2016; Espeholt et al., 2018). Here, distance is structural depth in a memory-construction graph whose edges record retrieval exposure.

We introduce **MemQ**, which records retrieved parents in a *provenance DAG* and propagates TD errors along every backward path, weighting length-d paths by $(\gamma\lambda)^d$. Structural distance replaces temporal step count in TD(λ). An *Exogenous-Context MDP* formalizes externally supplied tasks; a first-order per-memory value projection makes combinatorial retrieval tractable with an explicit approximation residual. Eight-benchmark experiments examine held-out performance and runtime learning, separating interaction, code-generation, and agentic tasks from multiple-choice reasoning. Ablations, value and trajectory analyses, and cost measurements examine the provenance update.

Our contributions are as follows:

1. We identify the *multi-step credit assignment problem* in episodic memory and formalize the setting as an *Exogenous-Context MDP*, whose factored transition decouples exogenous task dynamics from endogenous memory evolution, motivating value decomposition over individual memories.
2. We develop **MemQ**, a provenance-based credit propagation mechanism that applies TD(λ) eligibility traces through the memory construction DAG, replacing temporal distance with structural depth, the first provenance-based credit assignment method for episodic memory valuation.
3. MemQ achieves the highest mean held-out, final runtime, and cumulative success rates on all eight benchmarks. The largest held-out gains over the best baseline for each benchmark occur on DSBench and AutomationBench. Ablations on LiveCodeBench, BFCL, and DSBench support the benefit of propagating credit beyond directly retrieved memories. We further examine MemQ's learning dynamics, the evolution of memory Q-values, and its limitations.

2 Related Works

Memory agents range from heuristic retrieval (Park et al., 2023; Shinn et al., 2023) to parametric policies (Yan et al., 2026; Zhang et al., 2026e) and non-parametric value learning. MemRL (Zhang et al., 2026b) updates per-memory Q-values by single-step EMA ($\gamma = 0$), while Memento 2 (Wang, 2026) learns retrieval through supervised learning in a Reflected MDP that absorbs the frozen LLM into environment dynamics. Classical eligibility traces propagate delayed credit over time (Sutton, 1988; Sutton & Barto, 2018). MemQ instead propagates feedback through recorded memory-creation exposures, retaining the frozen-LLM, non-parametric setting. Appendix I retains the full discussion of memory agents, memory operations, and RL-based retrieval.

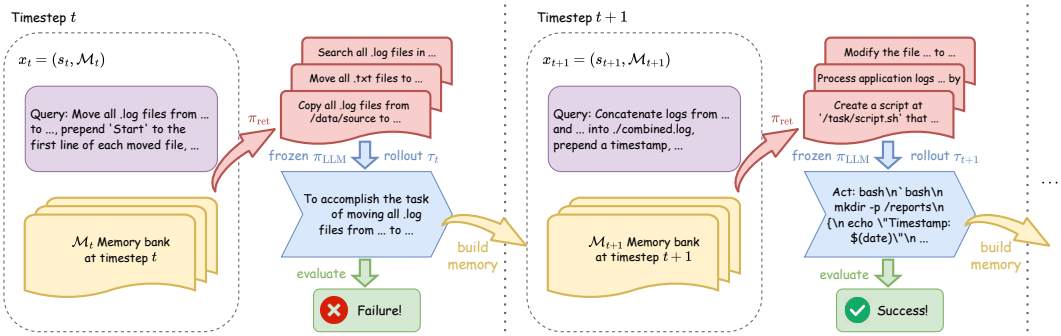


Figure 2: The EC-MDP. The state factors into an exogenous task stream $s_t \sim \rho$ and an endogenous memory store $\mathcal{M}_t$. The retrieval policy π_{ret} selects memories m_t, the frozen agent π_{LLM} produces a trajectory τ_t and receives reward r_t, and the memory is built and stored.

3 Problem Formulation

A frozen LLM agent tackles tasks from an unknown distribution using a growing episodic memory store. Because early retrievals can enable later success, credit assignment must account for their downstream impact across memory-creation chains.

Exogenous-Context MDP. The state separates an *exogenous* task stream, which the agent cannot influence, from an *endogenous* memory store changed by retrieval. We formalize this as an *Exogenous-Context MDP* (EC-MDP).

Definition 1 (Exogenous-Context MDP)**.** *An EC-MDP is a tuple $\langle \mathcal{S}, \mathcal{M}, \mathcal{A}, P_{\text{exo}}, P_{\text{endo}}, R, \gamma \rangle$, where $\mathcal{S}$ is the **exogenous task space**, with $P_{\text{exo}}(s_{t+1}) = \rho(s_{t+1})$ independent of actions and memory; $\mathcal{M} \subseteq 2^{\mathcal{M}_\infty}$[1] is the **endogenous memory state space**; and $\mathcal{A}(\mathcal{M}) = \{A \subseteq \mathcal{M} : |A| \leq k\}$ is the **retrieval action space**. The **endogenous transition kernel** incorporates the frozen LLM:*

$$P_{\text{endo}}(\mathcal{M}_{t+1} \mid s_t, \mathcal{M}_t, A_t) = \sum_{\tau} \pi_{\text{LLM}}(\tau \mid s_t, A_t) \cdot \mathbf{1}\big[\mathcal{M}_{t+1} = \mathcal{M}_t \cup \{\text{Build}(s_t, \tau)\}\big];$$

*$R(s_t, A_t) = \mathbb{E}_{\tau \sim \pi_{\text{LLM}}(\cdot \mid s_t, A_t)}[r(\tau)]$ is the **reward**, and $\gamma \in [0, 1]$ is the discount factor. Only the retrieval policy $\pi_{\text{ret}}(A \mid s, \mathcal{M})$, a distribution over retrieval actions given the task and memory store, is optimized.*

This setting models benchmark instances, independent user queries, and externally scheduled tasks, whose distribution is independent of the current retrieval action.

The EC-MDP joint transition kernel (Figure 2) factors as:

$$P(s_{t+1}, \mathcal{M}_{t+1} \mid s_t, \mathcal{M}_t, A_t) = P_{\text{exo}}(s_{t+1}) \cdot P_{\text{endo}}(\mathcal{M}_{t+1} \mid s_t, \mathcal{M}_t, A_t). \tag{1}$$

Relative to the Reflected MDP (Wang, 2026), this separates externally generated tasks from retrieval-dependent memory. Monotonic growth ($\mathcal{M}_{t+1} \supseteq \mathcal{M}_t$), with the current task and memory store in the state, yields a Markov process under the frozen-LLM transition. Reward $r_t = r(\tau_t)$ and new memory $m_{\text{new}} = \text{Build}(s_t, \tau_t)$ depend on the current task and retrieved set:

$$P(r_t, m_{\text{new}} \mid s_t, \mathcal{M}_t, A_t) = P(r_t, m_{\text{new}} \mid s_t, A_t). \tag{2}$$

Value functions and learning objective. The *state-value* captures the long-term value of the memory store via the expected cumulative discounted reward:

$$V^{\pi_{\text{ret}}}(\mathcal{M}_t) = \mathbb{E}_{s_k \sim \rho,\, A_k \sim \pi_{\text{ret}},\, \tau_k \sim \pi_{\text{LLM}}} \left[\sum_{k=0}^{\infty} \gamma^k r(\tau_{t+k}) \,\middle|\, \mathcal{M}_t \right].$$

The *action-value* for retrieving set A is $Q(s, A; \mathcal{M}) = \mathbb{E}_\tau\big[r(\tau) + \gamma V^{\pi_{\text{ret}}}(\mathcal{M}') \,\big|\, s, A, \mathcal{M}\big]$, where $\mathcal{M}' = \mathcal{M} \cup \{m_{\text{new}}(\tau)\}$. The learning objective is thus to maximize the initial state value:

$$\max_{\pi_{\text{ret}}} V^{\pi_{\text{ret}}}(\mathcal{M}_0).$$

[1] Each memory state $\mathcal{M}$ is a finite subset of the universe of possible experiences $\mathcal{M}_\infty$.

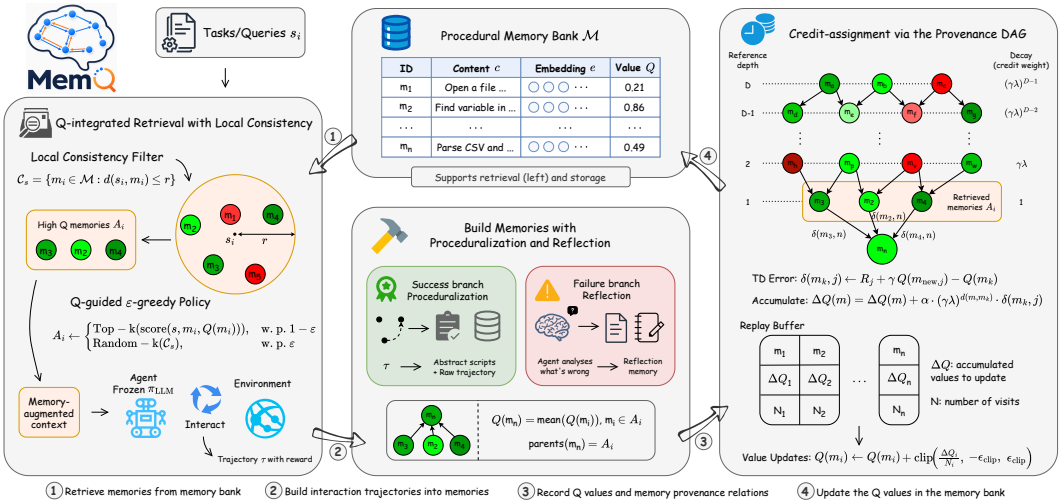


Figure 3: MemQ framework overview. Retrieve: Select locally relevant memories with an ϵ-greedy, Q-guided policy. Build: Distill task trajectories into new memories linked to retrieved parents in the provenance DAG. Update: Propagate TD errors backward through the DAG to update ancestor Q-values via multi-step credit assignment.

Directly representing the action value of every retrieval subset is intractable. We therefore use a parameter-free first-order projection onto per-memory values, while retaining an explicit residual for interactions within the retrieved set:

$$Q^\pi(s, A; \mathcal{M}) = \frac{1}{|A|} \sum_{m \in A} q^\pi(m) + \xi^\pi(s, A; \mathcal{M}), \quad A \sim \pi_{\text{ret}}(\cdot \mid s, \mathcal{M}). \tag{3}$$

The residual ξ^π captures synergy, redundancy, and distraction among co-retrieved memories. MemQ uses the first-order approximation $Q^\pi(s, A; \mathcal{M}) \approx |A|^{-1} \sum_{m \in A} q^\pi(m)$ and learns $Q(m)$ as an estimate of the per-memory term $q^\pi(m)$. Appendix B.3 gives the approximation-error and one-step selection-regret bounds.

4 Method

MemQ learns per-memory Q-values for the EC-MDP decomposition in Eq. 3. For each new memory, it records $\text{parents}(m_{\text{new}}) = A$, forming a provenance DAG (directed acyclic graph) whose edges mean "was retrieved for a task that produced the next memory." TD(λ) traces propagate backward through this graph, crediting ancestors by structural depth. The method comprises a Q-augmented memory store, two-phase retrieval, and provenance credit propagation; Algorithm 1 in Appendix A gives the full procedure.

4.1 Procedural Construction of Q-Augmented Memory

Let $\mathcal{M} = \{m_1, m_2, \ldots, m_N\}$ denote the episodic memory store, where N grows as the agent accumulates experience. Each memory entry is a tuple

$$m_i = (c_i, \mathbf{e}_i, Q_i),$$

where $c_i \in \Sigma^*$ is the memory content (natural language text encoding a past experience), $\mathbf{e}_i = \phi(c_i) \in \mathbb{R}^d$ is its retrieval embedding computed by a frozen encoder ϕ, and $Q_i \in \mathbb{R}$ is the *Q-value*, the agent's current estimate of m_i's value when retrieved for a relevant task.

After each interaction, an LLM *proceduralizes* trajectory τ into an abstract 3–5 step script, storing both script and raw trajectory. Successful trajectories are stored directly; failures additionally yield a separately stored reflection. To mitigate cold start, new memories inherit their retrieved parents'

mean Q-value, a parameter-free estimate consistent with Eq. 3:

$$Q(m_{\text{new}}) = \frac{1}{|A_j|} \sum_{m_i \in A_j} Q(m_i),$$

Here A_j is task j's retrieved set. For $A_j = \emptyset$, we set $Q(m_{\text{new}}) = Q_0$.

4.2 Q-integrated Retrieval Policy within Local Consistency

Retrieval selects $A \subseteq \mathcal{M}$ in two phases: *locality* restricts candidates to a neighborhood where the LLM can reliably generalize; *Q-guided selection* favors candidates with high learned long-run value.

Locality filtering. Capable LLMs can interpolate similar experiences without exact matches. By *LLM local consistency* (Wang, 2026), memories within distance r (the *radius of competence*) of state s yield a small optimality gap $\varepsilon_{\text{LLM}}(r)$. Any state-space metric suffices; we use cosine similarity $\text{sim}(s, m_i) = \phi(s)^\top \mathbf{e}_i / (\|\phi(s)\| \cdot \|\mathbf{e}_i\|)$ and candidates $\mathcal{C}_s = \{m_i \in \mathcal{M} : \text{sim}(s, m_i) \geq \theta_{\text{sim}}\}$. The threshold expresses competence in similarity space. An empty candidate set indicates a novel task and triggers zero-shot fallback.

Q-guided selection. Within the local candidate set, we combine raw similarity scores in $[0, 1]$ with Q-values normalized to $[0, 1]$:

$$\text{score}(s, m_i) = w_s \cdot \text{sim}(s, m_i) + w_q \cdot Q(m_i),$$

where w_s, w_q balance relevance and long-run value learned by the TD updates below. An ε-greedy policy selects

$$A = \begin{cases} \text{Top-}k \text{ by score}(s, m_i) & \text{with probability } 1 - \varepsilon, \\ \text{Uniform sample of } k \text{ from } \mathcal{C}_s & \text{with probability } \varepsilon, \end{cases}$$

ensuring continued exploration of currently undervalued memories.

4.3 Provenance-Based Credit Assignment

In classical TD(λ) (Sutton & Barto, 2018; Sutton, 1988), the λ-*return* $G_t^\lambda = (1-\lambda) \sum_{n=1}^{\infty} \lambda^{n-1} G_t^{(n)}$ interpolates between the one-step bootstrap $G_t^{(1)} = r_t + \gamma\, Q(s_{t+1}, a_{t+1})$ and the Monte Carlo return $G_t^{(\infty)} = \sum_{k=0}^{\infty} \gamma^k r_{t+k}$. The λ-return advantage is standardly expressed as a discounted sum of future TD errors:

$$G_t^\lambda - Q(s_t, a_t) = \sum_{k=0}^{T-t-1} (\gamma\lambda)^k\, \delta_{t+k}, \tag{4}$$

where $\delta_t = r_t + \gamma\, Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)$. MemQ applies the geometric weighting in this identity to backward provenance paths, replacing elapsed time with DAG depth. The resulting structural update is analyzed directly: Appendix B proves convergence and fixed-point preservation in a fixed-DAG evaluation setting.

Per-memory TD error. For task s_j with retrieved set A_j, outcome R_j, and constructed memory $m_{\text{new},j}$, the per-memory TD error is

$$\delta(m_i, j) = R_j + \gamma\, Q(m_{\text{new},j}) - Q(m_i), \quad \forall\, m_i \in A_j. \tag{5}$$

Retrieved memories share the reward and bootstrap target but have different Q-values. Setting $\gamma = 0$ removes the bootstrap term, yielding single-step EMA based only on immediate reward.

Credit propagation via the provenance DAG. Unlike single-step updates to retrieved memories, MemQ propagates each root's observed TD error $\delta(m_0, j)$, $m_0 \in A_j$, along every backward provenance path to its ancestors:

$$\Delta Q(m) \mathrel{+}= \alpha \sum_{m_0 \in A_j} \sum_{p \in \mathcal{P}_D(m_0 \rightsquigarrow m)} (\gamma\lambda)^{|p|}\, \delta(m_0, j), \tag{6}$$

Table 1: Held-out success rate (Eval SR, %) on interaction, code-generation, and agentic tasks. The best-epoch memory bank is selected on the memory-accumulation split before held-out evaluation. Entries are mean ± sample SD. Approximate 95% CIs are shown below the averages. Best in **bold**; second-best underlined.

Benchmark	Model	No Mem.	RAG	Self-RAG	Mem0	MemP	MemRL	*MemQ*
LLAB OS Interaction	`4o-mini`	$66.89_{\pm 1.26}$	$68.89_{\pm 0.38}$	$70.45_{\pm 0.39}$	$70.00_{\pm 0.67}$	$71.11_{\pm 0.38}$	$\underline{74.44}_{\pm 1.02}$	$\mathbf{74.67}_{\pm 0.67}$
LiveCodeBench Coding	`Gemma-4-E4B-it`	$44.76_{\pm 2.29}$	$\underline{50.48}_{\pm 1.65}$	$49.52_{\pm 1.65}$	$47.62_{\pm 1.65}$	$49.52_{\pm 1.65}$	$45.71_{\pm 2.86}$	$\mathbf{51.43}_{\pm 2.86}$
BFCL Function Call	`Qwen3.5-35B-A3B`	$56.71_{\pm 1.71}$	$55.45_{\pm 2.62}$	$60.07_{\pm 2.49}$	$54.79_{\pm 1.68}$	$\underline{61.39}_{\pm 0.99}$	$60.07_{\pm 2.29}$	$\mathbf{62.38}_{\pm 1.71}$
DSBench Data Science	`DeepSeek-v4-flash-0731`	$43.97_{\pm 0.70}$	$41.09_{\pm 1.79}$	$43.97_{\pm 2.28}$	$\underline{45.40}_{\pm 0.50}$	$44.54_{\pm 3.98}$	$42.53_{\pm 0.99}$	$\mathbf{47.70}_{\pm 0.50}$
AutomationBench Workflow Automation		$30.28_{\pm 1.74}$	$41.25_{\pm 1.77}$	$39.58_{\pm 0.59}$	$33.33_{\pm 2.36}$	$41.67_{\pm 2.36}$	$\underline{43.17}_{\pm 2.36}$	$\mathbf{44.69}_{\pm 2.36}$
Average		48.52 [47.54, 49.51]	51.43 [50.33, 52.53]	52.72 [51.63, 53.80]	50.23 [49.26, 51.20]	<u>53.65</u> <u>[52.09, 55.20]</u>	53.18 [51.93, 54.44]	**56.17** **[54.98, 57.37]**

Here $\mathcal{P}_D(m_0 \rightsquigarrow m)$ contains all distinct backward paths of length at most D, including the length-zero path for $m = m_0$. Each path contributes separately with weight $(\gamma\lambda)^{|p|}$ and increments N_m once, transferring Eq. 4's temporal decay to structural distance. Paths below weight 10^{-12} are pruned. Updates accumulate within each epoch and are then flushed:

$$Q(m_i) \leftarrow Q(m_i) + \text{clip}\Big(\frac{\Delta Q(m_i)}{N_i},\ -\epsilon_{\text{clip}},\ \epsilon_{\text{clip}}\Big).$$

5 Experiments

We evaluate MemQ on eight benchmarks through four research questions on held-out and runtime performance, provenance credit assignment, memory use during learning, and the limits of its gains.

5.1 Experimental Setup

We compare MemQ with representative state-of-the-art memory methods under a frozen-backbone setting: **Self-RAG** (Asai et al., 2024), **Mem0** (Chhikara et al., 2025), **MemP** (Fang et al., 2026), and **MemRL** (Zhang et al., 2026b), alongside **No Memory** and **RAG**. We use GPT-4o-mini for LLAB, Qwen3.5-35B-A3B (FP8) for BFCL, DeepSeek-v4-flash-0731 (284B) for DSBench and AutomationBench, and Gemma-4-E4B-it for LiveCodeBench, MMMU Pro, ERQA, and GPQA. Appendix C details the baselines, benchmarks, data splits, and backbones.

5.2 RQ1: Does MemQ improve held-out performance and runtime learning?

Across the five interaction, code-generation, and agentic benchmarks, including tool-mediated DSBench and AutomationBench, MemQ attains average held-out, final runtime, and cumulative SRs of 56.17%, 71.50%, and 78.88% (Tables 1 and 2). Its held-out average exceeds MemP, the strongest baseline, by 2.53 pp; final runtime SR and CSR exceed their strongest baseline, MemRL, by 4.23 pp and 2.13 pp.

Held-out performance. MemQ's consistent held-out advantage indicates that its benefits extend to new tasks. Its two largest margins over each task's strongest baseline occur on DSBench and AutomationBench, both using the 284B DeepSeek backbone. These larger gains suggest that model capability and learned memory selection can be complementary in agentic tasks. A plausible explanation is that more capable models make better use of retrieved procedures: they can adapt past experience to new requirements and translate it into effective tool-use decisions, increasing the value of selecting useful memories.

Runtime learning. MemQ also leads final SR and CSR on each of these benchmarks. Relative to MemRL, its final-SR gains reach 5.83 pp on LiveCodeBench, 3.77 pp on BFCL, 3.52 pp on

Table 2: Runtime learning results (%). Within each benchmark, rows show final-epoch success rate (Final SR) and cumulative success rate (CSR), respectively, on the memory-accumulation split. CSR is the fraction of tasks solved at least once. Entries are mean ± sample SD. Approximate 95% CIs are shown below the averages. Best in **bold**; second-best underlined.

Benchmark	Model	No Mem.	RAG	Self-RAG	Mem0	MemP	MemRL	*MemQ*
LLAB OS Interaction	4o-mini	$63.60_{\pm 0.28}$ –	$65.20_{\pm 1.14}$ $68.47_{\pm 0.84}$	$62.33_{\pm 0.62}$ $75.27_{\pm 0.74}$	$62.53_{\pm 0.25}$ $78.40_{\pm 0.43}$	$76.07_{\pm 0.52}$ $79.27_{\pm 0.52}$	$\underline{77.13_{\pm 0.68}}$ $\underline{80.80_{\pm 0.91}}$	$\mathbf{80.34}_{\pm 2.22}$ $\mathbf{82.27}_{\pm 1.88}$
LiveCodeBench Coding	Gemma-4-E4B-it	$47.14_{\pm 1.43}$ –	$51.96_{\pm 2.11}$ $57.38_{\pm 0.89}$	$45.89_{\pm 2.78}$ $65.00_{\pm 2.42}$	$49.52_{\pm 1.68}$ $56.19_{\pm 0.34}$	$53.04_{\pm 0.78}$ $55.48_{\pm 0.89}$	$\underline{58.93_{\pm 0.62}}$ $\underline{66.19_{\pm 1.47}}$	$\mathbf{64.76}_{\pm 1.49}$ $\mathbf{69.76}_{\pm 2.18}$
BFCL Function Call	Qwen3.5-35B-A3B	$54.52_{\pm 1.30}$ –	$55.28_{\pm 2.31}$ $70.77_{\pm 0.31}$	$64.91_{\pm 0.78}$ $80.99_{\pm 1.33}$	$57.37_{\pm 0.83}$ $73.62_{\pm 1.28}$	$65.91_{\pm 1.20}$ $80.15_{\pm 1.14}$	$\underline{71.27_{\pm 0.47}}$ $\underline{85.85_{\pm 1.44}}$	$\mathbf{75.04}_{\pm 0.31}$ $\mathbf{86.43}_{\pm 1.13}$
DSBench Data Science	DeepSeek-v4-flash-0731	$58.36_{\pm 1.03}$ –	$66.38_{\pm 4.54}$ $82.38_{\pm 3.06}$	$63.81_{\pm 2.33}$ $\underline{88.29_{\pm 0.29}}$	$66.19_{\pm 1.75}$ $87.43_{\pm 0.29}$	$64.67_{\pm 1.72}$ $83.43_{\pm 2.23}$	$\underline{68.00_{\pm 0.58}}$ $\underline{86.76_{\pm 1.15}}$	$\mathbf{71.52}_{\pm 0.44}$ $\mathbf{88.67}_{\pm 1.00}$
AutomationBench Workflow Automation		$28.75_{\pm 0.42}$ –	$46.46_{\pm 1.30}$ $50.63_{\pm 0.81}$	$44.79_{\pm 1.77}$ $58.02_{\pm 1.15}$	$46.56_{\pm 0.64}$ $\underline{65.60_{\pm 1.83}}$	$58.96_{\pm 0.59}$ $63.33_{\pm 0.21}$	$\underline{61.01_{\pm 1.03}}$ $\underline{64.17_{\pm 0.93}}$	$\mathbf{65.83}_{\pm 1.53}$ $\mathbf{67.29}_{\pm 1.84}$
Average (Final SR)		50.47 [49.86, 51.09]	57.06 [55.30, 58.81]	56.35 [55.17, 57.52]	56.43 [55.67, 57.20]	63.73 [63.05, 64.41]	<u>67.27</u> <u>[66.84, 67.69]</u>	**71.50** **[70.59, 72.40]**
Average (CSR)		– –	65.93 [64.68, 67.17]	73.51 [72.57, 74.45]	72.25 [71.51, 72.98]	72.33 [71.46, 73.20]	<u>76.75</u> <u>[76.05, 77.46]</u>	**78.88** **[77.89, 79.88]**

DSBench, and 4.82 pp on AutomationBench. MemQ reaches a CSR of 88.67% on DSBench, compared with Self-RAG's 88.29%, and 67.29% on AutomationBench, compared with 65.60% for Mem0. The evolution of SR and CSR during runtime learning across all eight benchmarks is shown in Appendix D.

5.3 RQ2: How does provenance credit assignment affect performance?

We compare full MemQ with variants without bootstrapping or multi-depth propagation on LiveCodeBench, BFCL, and DSBench (Figure 5a). The discount γ weights future memory value in the bootstrap target, while λ controls the decay of observed TD errors; together they give the ancestor weight $(\gamma\lambda)^d$ at provenance depth d. Setting $\gamma = 0$ removes bootstrapping and all positive-depth contributions. Setting $\lambda = 0$ retains the one-step bootstrap but stops the observed TD error from reaching earlier provenance ancestors. We also sweep each parameter while holding the other fixed.

Full MemQ has the highest point estimates on all three metrics across the three benchmarks (Figure 5a). Setting $\gamma = 0$ lowers evaluation SR by 5.72 pp on LiveCodeBench and 2.43 pp on BFCL; $\lambda = 0$ lowers it by 8.57 and 4.41 pp, respectively. On DSBench, final/evaluation SR drops by 3.52/3.73 pp at $\gamma = 0$ and 5.71/4.88 pp at $\lambda = 0$, with lower CSR in both cases. Lower runtime and held-out scores at $\lambda = 0$, which retains bootstrapping, support propagating feedback beyond directly retrieved memories.

The benefit of a larger γ depends on the task. The discount γ shapes both the bootstrap $\gamma Q(m_{\text{new}})$ and structural weight $(\gamma\lambda)^d$ (Eqs. 5–6). Figure 4 (top) shows a reversal: LiveCodeBench peaks near $\gamma = 0.5$ and declines sharply at 0.9 (∼63% vs. ∼56%); BFCL favors $\gamma \in [0.8, 1.0]$ (∼76% vs. ∼73% at $\gamma = 0$). These contrasting responses suggest that the benefit of longer-range credit propagation depends on the task. On DSBench with $\lambda = 0.7$, $\gamma = 0.5$ and 0.9 both reach 68.86% final mean Train SR, versus 65.05% at 0.3: moderate and high discounts are effective, with no further final-SR gain from 0.5 to 0.9.

Moderate trace decay balances credit propagation and attenuation. Trace decay λ governs backward credit, not bootstrapping, weighting depth-d ancestors by $(\gamma\lambda)^d$ (Eq. 6). On LiveCodeBench at $\gamma = 0.3$ (Figure 4, bottom), $\lambda = 0.3$ gives the highest SR (∼65.8%) and 0.9 the lowest (∼59.5%). BFCL (Figures 4, bottom, and 12) favors $\lambda = 0.5$ over both 0.0 and 0.8. DSBench likewise favors intermediate decay: at $\gamma = 0.9$, $\lambda = 0.5$ reaches 70.29% final mean Train SR, versus 65.81% at 0 and 64.95% at 0.9. Memory evolution carries experience across exogenous tasks. The sweep results favor intermediate trace decay, suggesting that credit beyond direct parents is useful when longer-path contributions are attenuated.

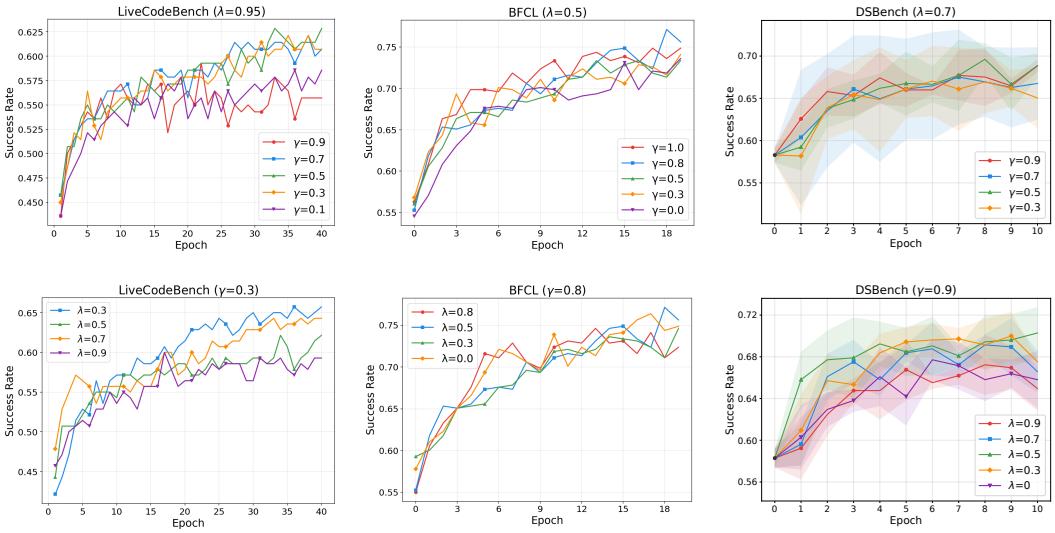


Figure 4: Ablations on γ (top) and λ (bottom): Success Rate on LiveCodeBench, BFCL, and DSBench.

5.4 RQ3: How does MemQ use memory during runtime learning?

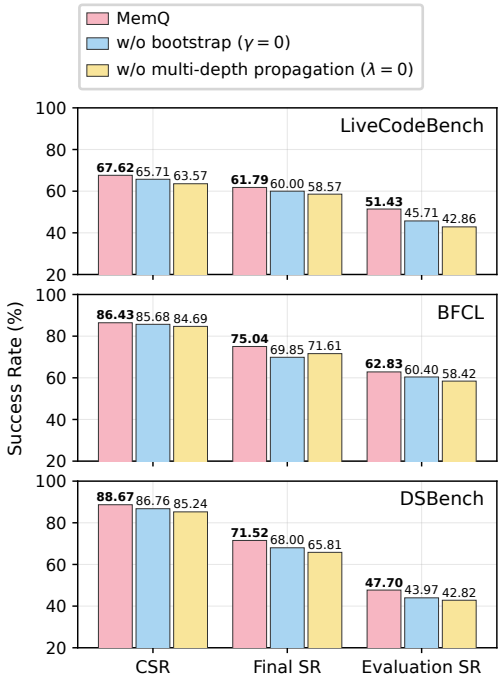
We examine how learned Q-values guide memory retrieval, how task coverage evolves during runtime learning, and how many LLM tokens this process consumes.

Learned Q-values are associated with subsequent task success. On AutomationBench, the top-ranked memory's normalized retrieval-time Q-value correlates with exact task success (Pearson $r = 0.640$, 480 rollouts). Success rises from 16.7% in the lowest Q range to 91.5% at 0.7–0.8 (Figure 5b); values recorded before execution thus distinguish contexts with different observed success rates. High-Q ranges contain a higher proportion of memories from successful trajectories but retain some failed trajectories; Appendix E gives the checkpoint distributions for MemQ and MemRL.

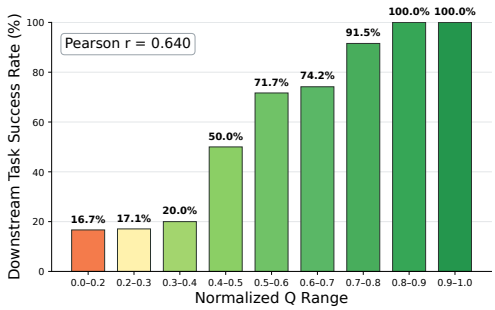
Q-values complement semantic similarity in retrieval. Within locality-filtered candidates, similarity-only and Q-only ranking yield 42.86% and 45.71% evaluation SR on LiveCodeBench, versus 51.43% for mixed ranking. On BFCL, several mixtures exceed both endpoints (56.77%, 59.41%), although the best weight differs between evaluation and runtime metrics (Tables 4 and 5). Learned value and semantic relevance provide complementary signals.

MemQ maintains more of its historical task coverage. CSR minus final SR measures the fraction of tasks solved previously but not in the final epoch, including stochastic failures. On BFCL, MemQ has the highest CSR and final SR, with a smaller gap than MemRL (11.39 versus 14.58 pp; Table 15, Appendix H). On AutomationBench (Figure 5c), MemP plateaus earlier while MemRL and MemQ continue improving; MemQ's final gap is 1.46 pp versus 3.16 for MemRL and 4.37 for MemP. MemQ thus combines broader coverage with better final retention. Appendix D provides runtime-evolving SR and CSR curves for all eight benchmarks.

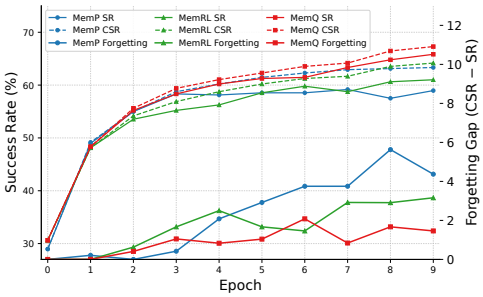
MemQ uses memory more effectively and efficiently. Among the proceduralization methods on DSBench, MemQ uses 16.2% fewer total LLM tokens than MemP and 11.2% fewer than MemRL over 10 training epochs (Table 16, Appendix H; evaluation and embedding excluded). Most savings are in input tokens, despite higher SR (Tables 1 and 2). More compact, informative memories may guide decisions more directly, reducing contextual overhead.



(a) Bootstrapping and multi-depth propagation.



(b) Q-value and downstream success.



(c) Runtime learning dynamics.

Figure 5: Credit-assignment ablations and memory analysis. (a) Removing bootstrapping or multi-depth propagation. (b) Downstream success by the top-ranked memory's normalized retrieval-time Q-value on AutomationBench (480 rollouts). (c) Runtime SR and CSR on AutomationBench.

5.5 RQ4: When are MemQ's gains limited?

After the challenging agentic tasks, we probe the limits of MemQ's gains on MMMU Pro, ERQA, and GPQA Diamond. MemQ has the highest held-out mean on each, but leads the strongest baseline by only 0.19, 0.83, and 0.84 pp, respectively. Table 17 and Appendix H give the full results and runtime–held-out comparisons.

These benchmarks present largely self-contained questions: most task context is provided in the question, and answering it relies mainly on the model's knowledge and reasoning. Without extended multi-turn interaction with an environment, procedural experience from earlier tasks has fewer opportunities to guide subsequent actions. This may explain the smaller additional gains from MemQ's accumulated experience.

6 Conclusion

In this paper, we studied multi-step credit assignment across episodic-memory creation events in LLM agents. We formalized the setting as an Exogenous-Context MDP and developed MemQ, replacing temporal distance in TD(λ)-inspired credit with provenance-path depth. Our analysis establishes fixed-DAG convergence and full-buffer fixed-point preservation. Experiments across eight benchmarks show improved held-out and runtime performance, with the largest held-out gains over each task's strongest baseline on DSBench and AutomationBench and smaller gains on multiple-choice reasoning. Ablations support bootstrapping and multi-depth propagation; learned Q-values associate with subsequent success, and learning dynamics show better retention of historical coverage. MemQ also uses fewer DSBench training tokens than MemP and MemRL. These findings demonstrate the value of memory-creation dependencies when learning to retrieve and reuse experience.

AI USE STATEMENT

We used generative AI tools to assist with writing and editing, literature search, theoretical formulation and proofs, experimental code implementation, and result analysis. The authors reviewed and verified all AI-assisted work and take full responsibility for the paper.

REPRODUCIBILITY STATEMENT

Section 4 specifies memory construction, retrieval, and provenance-based updates; Appendix A provides the complete algorithm. Appendix B states the theoretical assumptions and proofs. Appendices C and G document baseline implementations, benchmarks, data splits, model backbones, and hyperparameters. Additional learning curves and ablations appear in Appendices D and F.

REFERENCES

Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. Self-RAG: Learning to retrieve, generate, and critique through self-reflection. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=hSyW5go0v8.

Charles Blundell, Benigno Uria, Alexander Pritzel, Yazhe Li, Avraham Ruderman, Joel Z Leibo, Jack Rae, Daan Wierstra, and Demis Hassabis. Model-free episodic control, 2016. URL https://arxiv.org/abs/1606.04460.

V. S. Borkar and S. P. Meyn. The o.d. e. method for convergence of stochastic approximation and reinforcement learning. *SIAM J. Control Optim.*, 38(2):447–469, January 2000. ISSN 0363-0129. doi: 10.1137/S0363012997331639. URL https://doi.org/10.1137/S0363012997331639.

Zouying Cao, Jiaji Deng, Li Yu, Weikang Zhou, Zhaoyang Liu, Bolin Ding, and Hai Zhao. Remember me, refine me: A dynamic procedural memory framework for experience-driven agent evolution, 2026. URL https://arxiv.org/abs/2512.10696.

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

DeepSeek-AI. DeepSeek-V4: Towards highly efficient million-token context intelligence, 2026. URL https://arxiv.org/abs/2606.19348.

Lasse Espeholt, Hubert Soyer, Rémi Munos, Karen Simonyan, Volodymyr Mnih, Tom Ward, Yotam Doron, Vlad Firoiu, Tim Harley, Iain Dunning, Shane Legg, and Koray Kavukcuoglu. IMPALA: Scalable distributed deep-RL with importance weighted actor-learner architectures. In *International Conference on Machine Learning*, 2018.

Runnan Fang, Yuan Liang, Xiaobin Wang, Jialong Wu, Shuofei Qiao, Pengjun Xie, Fei Huang, Huajun Chen, and Ningyu Zhang. Memp: Exploring agent procedural memory, 2026. URL https://arxiv.org/abs/2508.06433.

Google Gemma Team. Gemma 4 e4b-it. https://huggingface.co/google/gemma-4-E4B-it, 2026. Vertex AI / Hugging Face Model Hub.

Alex Graves, Greg Wayne, and Ivo Danihelka. Neural turing machines, 2014. URL https://arxiv.org/abs/1410.5401.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. Realm: retrieval-augmented language model pre-training. In *Proceedings of the 37th International Conference on Machine Learning*, ICML'20. JMLR.org, 2020.

Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. Livecodebench: Holistic and contamination free evaluation of large language models for code. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=chfJJYC3iL.

Liqiang Jing, Zhehui Huang, Xiaoyang Wang, Wenlin Yao, Wenhao Yu, Kaixin Ma, Hongming Zhang, Xinya Du, and Dong Yu. Dsbench: How far are data science agents to becoming data science experts?, 2024. URL https://arxiv.org/abs/2409.07703.

Brandon Kynoch, Hugo Latapie, and Dwane van der Sluis. Recallm: An adaptable memory mechanism with temporal understanding for large language models, 2023. URL https://arxiv.org/abs/2307.02738.

Zichuan Lin, Tianqi Zhao, Guangwen Yang, and Lintao Zhang. Episodic memory deep q-networks. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, IJCAI'18, pp. 2433–2439. AAAI Press, 2018. ISBN 9780999241127.

Weitao Ma, Xiaocheng Feng, Lei Huang, Xiachong Feng, Zhanyu Ma, Jun Xu, Jiuchong Gao, Jinghua Hao, Renqing He, and Bing Qin. Fine-mem: Fine-grained feedback alignment for long-horizon memory management, 2026a. URL https://arxiv.org/abs/2601.08435.

Wenquan Ma, Jiayan Nan, Wenlong Wu, and Yize Chen. What deserves memory: Adaptive memory distillation for llm agents, 2026b. URL https://arxiv.org/abs/2508.03341.

Rémi Munos, Tom Stepleton, Anna Harutyunyan, and Marc G Bellemare. Safe and efficient off-policy reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 29, 2016.

Siru Ouyang, Jun Yan, I-Hung Hsu, Yanfei Chen, Ke Jiang, Zifeng Wang, Rujun Han, Long Le, Samira Daruki, Xiangru Tang, Vishy Tirumalashetty, George Lee, Mahsan Rofouei, Hangfei Lin, Jiawei Han, Chen-Yu Lee, and Tomas Pfister. Reasoningbank: Scaling agent self-evolving with reasoning memory. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=jL7fwchScm.

Charles Packer, Sarah Wooders, Kevin Lin, Vivian Fang, Shishir G. Patil, Ion Stoica, and Joseph E. Gonzalez. Memgpt: Towards llms as operating systems, 2024. URL https://arxiv.org/abs/2310.08560.

Joon Sung Park, Joseph O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*, UIST '23, New York, NY, USA, 2023. Association for Computing Machinery. ISBN 9798400701320. doi: 10.1145/3586183.3606763. URL https://doi.org/10.1145/3586183.3606763.

Shishir G Patil, Huanzhi Mao, Fanjia Yan, Charlie Cheng-Jie Ji, Vishnu Suresh, Ion Stoica, and Joseph E. Gonzalez. The berkeley function calling leaderboard (BFCL): From tool use to agentic evaluation of large language models. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=2GmDdhBdDk.

Jing Peng and Ronald J Williams. Incremental multi-step Q-learning. *Machine Learning*, 22(1–3): 283–290, 1996.

Alexander Pritzel, Benigno Uria, Sriram Srinivasan, Adrià Puigdomènech Badia, Oriol Vinyals, Demis Hassabis, Daan Wierstra, and Charles Blundell. Neural episodic control. In Doina Precup and Yee Whye Teh (eds.), *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 2827–2836. PMLR, 06–11 Aug 2017. URL https://proceedings.mlr.press/v70/pritzel17a.html.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026. URL https://qwen.ai/blog?id=qwen3.5.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. GPQA: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=Ti67584b98.

Herbert Robbins and David Siegmund. A convergence theorem for non negative almost supermartingales and some applications. In *Optimizing Methods in Statistics*, pp. 233–257. Academic Press, 1971. doi: 10.1016/B978-0-12-604550-5.50015-8. URL https://www.sciencedirect.com/science/article/pii/B9780126045505500158.

Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. Prioritized experience replay. In *International Conference on Learning Representations*, 2016.

John Schulman, Philipp Moritz, Sergey Levine, Michael I Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. In *International Conference on Learning Representations*, 2016.

Harm Seijen and Rich Sutton. True online td(lambda). In Eric P. Xing and Tony Jebara (eds.), *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pp. 692–700, Bejing, China, 22–24 Jun 2014. PMLR. URL https://proceedings.mlr.press/v32/seijen14.html.

Zhiyu Shen, Ziming Wu, Fuming Lai, Shaobing Lian, and Yanghui Rao. Membuilder: Reinforcing llms for long-term memory construction via attributed dense rewards, 2026. URL https://arxiv.org/abs/2601.05488.

Daniel Shepard and Robin Salimans. Automationbench, 2026. URL https://arxiv.org/abs/2604.18934.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik R Narasimhan, and Shunyu Yao. Reflexion: language agents with verbal reinforcement learning. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=vAElhFcKW6.

Satinder P. Singh and Richard S. Sutton. Reinforcement learning with replacing eligibility traces. *Mach. Learn.*, 22(1–3):123–158, January 1996. ISSN 0885-6125. doi: 10.1007/BF00114726. URL https://doi.org/10.1007/BF00114726.

Theodore Sumers, Shunyu Yao, Karthik R Narasimhan, and Thomas L. Griffiths. Cognitive architectures for language agents. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=1i6ZCvflQJ. Survey Certification, Featured Certification.

Richard S Sutton. Learning to predict by the methods of temporal differences. *Machine Learning*, 3 (1):9–44, 1988.

Richard S Sutton and Andrew G Barto. *Reinforcement Learning: An Introduction*. MIT Press, 2nd edition, 2018.

Gemini Robotics Team, Saminda Abeyruwan, Joshua Ainslie, Jean-Baptiste Alayrac, Montserrat Gonzalez Arenas, Travis Armstrong, Ashwin Balakrishna, Robert Baruch, Maria Bauza, Michiel Blokzijl, Steven Bohez, Konstantinos Bousmalis, Anthony Brohan, Thomas Buschmann, Arunkumar Byravan, Serkan Cabi, Ken Caluwaerts, Federico Casarini, Oscar Chang, Jose Enrique Chen, Xi Chen, Hao-Tien Lewis Chiang, Krzysztof Choromanski, David D'Ambrosio, Sudeep Dasari, Todor Davchev, Coline Devin, Norman Di Palo, Tianli Ding, Adil Dostmohamed, Danny Driess, Yilun Du, Debidatta Dwibedi, Michael Elabd, Claudio Fantacci, Cody Fong, Erik Frey, Chuyuan Fu, Marissa Giustina, Keerthana Gopalakrishnan, Laura Graesser, Leonard Hasenclever, Nicolas Heess, Brandon Hernaez, Alexander Herzog, R. Alex Hofer, Jan Humplik, Atil Iscen, Mithun George Jacob, Deepali Jain, Ryan Julian, Dmitry Kalashnikov, M. Emre Karagozler, Stefani Karp, Chase Kew, Jerad Kirkland, Sean Kirmani, Yuheng Kuang, Thomas Lampe, Antoine Laurens, Isabel Leal, Alex X. Lee, Tsang-Wei Edward Lee, Jacky Liang, Yixin Lin, Sharath Maddineni, Anirudha Majumdar, Assaf Hurwitz Michaely, Robert Moreno, Michael Neunert, Francesco Nori, Carolina Parada, Emilio Parisotto, Peter Pastor, Acorn Pooley, Kanishka Rao, Krista Reymann, Dorsa Sadigh, Stefano Saliceti, Pannag Sanketi, Pierre Sermanet, Dhruv Shah, Mohit Sharma, Kathryn Shea, Charles Shu, Vikas Sindhwani, Sumeet Singh, Radu Soricut, Jost Tobias Springenberg, Rachel Sterneck, Razvan Surdulescu, Jie Tan, Jonathan Tompson, Vincent Vanhoucke, Jake Varley, Grace Vesom, Giulia Vezzani, Oriol Vinyals, Ayzaan Wahid, Stefan Welker, Paul Wohlhart, Fei Xia, Ted Xiao, Annie Xie, Jinyu Xie, Peng Xu, Sichun Xu, Ying Xu, Zhuo Xu, Yuxiang Yang, Rui Yao, Sergey Yaroshenko, Wenhao Yu, Wentao Yuan, Jingwei Zhang, Tingnan Zhang, Allan Zhou, and Yuxiang Zhou. Gemini robotics: Bringing ai into the physical world, 2025. URL https://arxiv.org/abs/2503.20020.

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. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=ehfRiF0R3a.

Jun Wang. Memento 2: Learning by stateful reflective memory, 2026. URL https://arxiv.org/abs/2512.22716.

Christopher John Cornish Hellaby Watkins. *Learning from Delayed Rewards*. PhD thesis, King's College, Cambridge, 1989.

Tianxin Wei, Noveen Sachdeva, Benjamin Coleman, Zhankui He, Yuanchen Bei, Xuying Ning, Mengting Ai, Yunzhe Li, Jingrui He, Ed H. Chi, Chi Wang, Shuo Chen, Fernando Pereira, Wang-Cheng Kang, and Derek Zhiyuan Cheng. Evo-memory: Benchmarking llm agent test-time learning with self-evolving memory, 2025. URL https://arxiv.org/abs/2511.20857.

Sikuan Yan, Xiufeng Yang, Zuchao Huang, Ercong Nie, Zifeng Ding, Zonggen Li, Xiaowen Ma, Jinhe Bi, Kristian Kersting, Jeff Z. Pan, Hinrich Schütze, Volker Tresp, and Yunpu Ma. Memory-R1: Enhancing large language model agents to manage and utilize memories via reinforcement learning, 2026. URL https://arxiv.org/abs/2508.19828.

Qianhao Yuan, Jie Lou, Zichao Li, Jiawei Chen, Yaojie Lu, Hongyu Lin, Le Sun, Debing Zhang, and Xianpei Han. Memsearcher: Training LLMs to reason, search and manage memory via end-to-end reinforcement learning, 2026. URL https://openreview.net/forum?id=EWIAx3NgvA.

Xiang Yue, Tianyu Zheng, Yuansheng Ni, Yubo Wang, Kai Zhang, Shengbang Tong, Yuxuan Sun, Botao Yu, Ge Zhang, Huan Sun, Yu Su, Wenhu Chen, and Graham Neubig. MMMU-pro: A more robust multi-discipline multimodal understanding benchmark. 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. 15134–15186, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.736. URL https://aclanthology.org/2025.acl-long.736/.

Yanwei Yue, Boci Peng, Xuanbo Fan, Jiaxin Guo, Qiankun Li, and Yan Zhang. Mem-t: Densifying rewards for long-horizon memory agents, 2026. URL https://arxiv.org/abs/2601.23014.

Danyang Zhang, Lu Chen, Situo Zhang, Hongshen Xu, Zihan Zhao, and Kai Yu. Large language models are semi-parametric reinforcement learning agents. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=ZcJa1R6j3v.

Guibin Zhang, Haotian Ren, Chong Zhan, Zhenhong Zhou, Junhao Wang, He Zhu, Wangchunshu Zhou, and Shuicheng Yan. Memevolve: Meta-evolution of agent memory systems, 2025. URL https://arxiv.org/abs/2512.18746.

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, 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.

Xiaoying Zhang, Zichen Liu, Yipeng Zhang, Xia Hu, and Wenqi Shao. Retroagent: From solving to evolving via retrospective dual intrinsic feedback, 2026c. URL https://arxiv.org/abs/2603.08561.

Yaolun Zhang, Yiran Wu, Yijiong Yu, Qingyun Wu, and Huazheng Wang. Live-evo: Online evolution of agentic memory from continuous feedback, 2026d. URL https://arxiv.org/abs/2602.02369.

Yuxiang Zhang, Jiangming Shu, Ye Ma, Xueyuan Lin, Shangxi Wu, and Jitao Sang. Memory as action: Autonomous context curation for long-horizon agentic tasks, 2026e. URL https://arxiv.org/abs/2510.12635.

Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. Expel: Llm agents are experiential learners. In *Proceedings of the Thirty-Eighth AAAI Conference on Artificial Intelligence and Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence and Fourteenth Symposium on Educational Advances in Artificial Intelligence*, AAAI'24/IAAI'24/EAAI'24. AAAI Press, 2024. ISBN 978-1-57735-887-9. doi: 10.1609/aaai.v38i17.29936. URL https://doi.org/10.1609/aaai.v38i17.29936.

Junhao Zheng, Xidi Cai, Qiuke Li, Duzhen Zhang, ZhongZhi Li, Yingying Zhang, Le Song, and Qianli Ma. Lifelongagentbench: Evaluating llm agents as lifelong learners, 2025. URL https://arxiv.org/abs/2505.11942.

Wanjun Zhong, Lianghong Guo, Qiqi Gao, He Ye, and Yanlin Wang. Memorybank: enhancing large language models with long-term memory. In *Proceedings of the Thirty-Eighth AAAI Conference on Artificial Intelligence and Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence and Fourteenth Symposium on Educational Advances in Artificial Intelligence*, AAAI'24/IAAI'24/EAAI'24. AAAI Press, 2024. ISBN 978-1-57735-887-9. doi: 10.1609/aaai.v38i17.29946. URL https://doi.org/10.1609/aaai.v38i17.29946.

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, 2025a. URL https://arxiv.org/abs/2508.16153.

Zijian Zhou, Ao Qu, Zhaoxuan Wu, Sunghwan Kim, Alok Prakash, Daniela Rus, Jinhua Zhao, Bryan Kian Hsiang Low, and Paul Pu Liang. Mem1: Learning to synergize memory and reasoning for efficient long-horizon agents, 2025b. URL https://arxiv.org/abs/2506.15841.

Jinchang Zhu, Jindong Li, Cheng Zhang, Jiahong Liu, and Menglin Yang. Hela-mem: Hebbian learning and associative memory for llm agents, 2026. URL https://arxiv.org/abs/2604.16839.

A MEMQ ALGORITHM

Algorithm 1 gives the complete procedure described in Section 4.

Algorithm 1 MemQ with batch-parallel memory construction and every-path provenance credit propagation

Require: Task set $\mathcal{D}$, memory store $\mathcal{M}$, frozen agent π_{LLM}
Require: Hyperparameters: $\alpha, \gamma, \lambda, \varepsilon, k, \theta_{\text{sim}}, w_s, w_q, \epsilon_{\text{clip}}$, empty-parent value Q_0
Initialize DAG $\leftarrow \emptyset$; $\Delta Q_i \leftarrow 0$; $N_i \leftarrow 0$ for all m_i
for each epoch $\ell = 1, 2, \ldots$ **do**
  Shuffle $\mathcal{D}$ into mini-batches $\mathcal{B}_1, \ldots, \mathcal{B}_L$ of size Batch size
  $\mathcal{T} \leftarrow \emptyset$ ▷ transition buffer
  for each mini-batch $\mathcal{B}_b = \{s_1, \ldots, s_B\}$ **do**
    // Trajectory sampling
    for each task $s_j \in \mathcal{B}_b$ **do** ▷ in parallel
      $\mathcal{C}_j \leftarrow \{m \in \mathcal{M} : \text{sim}(\phi(s_j), \mathbf{e}_m) \geq \theta_{\text{sim}}\}$ ▷ locality filter
      $\text{score}(m) \leftarrow w_s \cdot \text{sim}(s_j, m) + w_q \cdot Q(m)$ for all $m \in \mathcal{C}_j$ ▷ Q-guided scoring
      $A_j \leftarrow \varepsilon$-greedy top-$k$ from $\mathcal{C}_j$
      for each step $t = 1, 2, \ldots$ **do** ▷ agent-environment interaction
        $a_t \leftarrow \pi_{\text{LLM}}(s_j, A_j, h_{<t})$; $o_t \leftarrow \text{Environment}(a_t)$
      end for
      $R_j \leftarrow \text{Environment.reward}()$
    end for
    for each completed record (s_j, τ_j, A_j, R_j) **do** ▷ batch-parallel build
      $c(m_{\text{new},j}) \leftarrow \text{Build}(s_j, \tau_j)$
      $Q(m_{\text{new},j}) \leftarrow |A_j|^{-1} \sum_{m_i \in A_j} Q(m_i)$ if $|A_j| > 0$, else Q_0
      $\mathcal{M} \leftarrow \mathcal{M} \cup \{m_{\text{new},j}\}$; $\mathcal{T} \leftarrow \mathcal{T} \cup \{(A_j, m_{\text{new},j}, R_j)\}$
    end for
  end for
  Record $\text{parents}(m_{\text{new},j}) \leftarrow A_j$ for each transition in $\mathcal{T}$ ▷ construct DAG
  // Value update using MemQ
  Compute $\delta(m_0, j) \leftarrow R_j + \gamma\, Q(m_{\text{new},j}) - Q(m_0)$ for each $m_0 \in A_j$ ▷ Eq. 5
  Enumerate every backward path p from each root m_0, up to depth D: ▷ path-wise DAG traversal
  for each path p ending at ancestor m **do**
    $\Delta Q(m) \leftarrow \Delta Q(m) + \alpha \cdot (\gamma\lambda)^{|p|} \cdot \delta(m_0, j)$; $N_m \leftarrow N_m + 1$ ▷ one contribution per path
  end for
  $Q(m_i) \leftarrow Q(m_i) + \text{clip}\Big(\frac{\Delta Q_i}{N_i},\, -\epsilon_{\text{clip}},\, \epsilon_{\text{clip}}\Big)$ for all m_i; $\Delta Q_i \leftarrow 0$; $N_i \leftarrow 0$ ▷ flush
end for

B THEORETICAL ANALYSIS OF MEMQ

We study the value recursion in Section 4.3 for a fixed set of memory-construction events in the EC-MDP of Section 3.

B.1 PRELIMINARIES

Let $\mathcal{T} = \{(A_j, c_j, R_j)\}_j$ be a finite transition buffer, where $c_j = m_{\text{new},j}$ is the child of the retrieved set A_j, and let $\mathcal{G}$ be the provenance DAG consisting of its recorded parent–child edges. Assume that every edge joins an older parent to a strictly newer child and that every nonterminal vertex occurs as a retrieved root. Index the $n \geq 1$ nonterminal memories in chronological order and write $q_i = Q(m_i)$. The terminal scores g_c are fixed, with $|R_j| \leq R_{\max}$ and $|g_c| \leq G_{\max}$. Throughout, $0 \leq \gamma < 1$, $\lambda \in [0, 1]$, and $\beta = \gamma\lambda$. We consider unclipped value updates on this fixed graph.

The stored-child bootstrap in Eq. 5 is

$$v_c(q) = \begin{cases} q_c, & c \text{ is nonterminal,} \\ g_c, & c \text{ is terminal.} \end{cases}$$

Thus nonterminal children contribute their current values. Let $n_u = |\{j : u \in A_j\}| > 0$ and define

$$P_{uv} = \frac{1}{n_u} \sum_{j:u\in A_j} \mathbf{1}\{c_j = v\}, \qquad \bar{r}_u = \frac{1}{n_u} \sum_{j:u\in A_j} \left[R_j + \gamma \mathbf{1}\{c_j \text{ is terminal}\} g_{c_j}\right],$$

where the terminal summand is zero otherwise. The matrix P is nonnegative, substochastic, and strictly upper triangular. Define the memory-lineage Bellman operator by

$$\mathcal{T}_{\mathcal{G}} q = \bar{r} + \gamma P q.$$

For finite D, fix $\mathcal{L} \subseteq \{0, \dots, D\}$ with $0 \in \mathcal{L}$. Let $\mathcal{P}_{\mathcal{L}}(u \rightsquigarrow i)$ denote the set of all backward paths from u to i whose lengths belong to $\mathcal{L}$, including the zero-length path for $i = u$. Distinct paths between the same pair of vertices are counted separately. The unpruned case has $\mathcal{L} = \{0, \dots, D\}$; fixed geometric-weight pruning amounts to restricting $\mathcal{L}$.

Let $B_{iu} = 1$ if i is an immediate parent of u, and zero otherwise. Define

$$W_{iu} = \sum_{p\in\mathcal{P}_{\mathcal{L}}(u\rightsquigarrow i)} \beta^{|p|}, \qquad W = \sum_{d\in\mathcal{L}} \beta^d B^d, \tag{7}$$

with the convention $\beta^0 = 1$ also at $\beta = 0$. Put

$$N_i = \sum_j \sum_{u\in A_j} |\mathcal{P}_{\mathcal{L}}(u \rightsquigarrow i)|, \qquad C = \operatorname{diag}(N_1, \dots, N_n), \qquad D_n = \operatorname{diag}(n_1, \dots, n_n).$$

Then $N_i \geq n_i > 0$. The normalized update corresponding to Eqs. 5 and 6 is

$$q_{t+1} = q_t + \alpha_t h(q_t), \qquad h_i(q) = \frac{1}{N_i} \sum_j \sum_{u\in A_j} \sum_{p\in\mathcal{P}_{\mathcal{L}}(u\rightsquigarrow i)} \beta^{|p|} \big(R_j + \gamma v_{c_j}(q) - q_u\big). \tag{8}$$

The residual in each summand is evaluated at the retrieved root u.

B.2 Convergence

Theorem B.1 (Convergence and fixed-point preservation)**.** *Under the preceding assumptions, the following hold.*

(i) The full-buffer update satisfies $h(q) = b - Hq$, where

$$H = C^{-1} W D_n (I - \gamma P), \qquad b = C^{-1} W D_n \bar{r}. \tag{9}$$

Its unique fixed point is

$$q^{\dagger}_{\mathcal{G}} = (I - \gamma P)^{-1} \bar{r} = \sum_{k=0}^{n-1} \gamma^k P^k \bar{r}. \tag{10}$$

This is also the fixed point of the depth-zero update. With $\mathcal{T}$ and the terminal scores fixed, it is independent of λ, $\mathcal{L}$, and the choice of positive diagonal normalizer.

(ii) For the path-count normalizer C defined above, H is upper triangular with eigenvalues $n_i/N_i \in (0, 1]$. From any finite q_0, the recursion converges to $q^{\dagger}_{\mathcal{G}}$ for either a constant step

$$0 < \alpha < \frac{2}{\max_i (n_i/N_i)}$$

(in particular, any $0 < \alpha < 2$), or deterministic positive steps with $\sum_t \alpha_t = \infty$ and $\sum_t \alpha_t^2 < \infty$.

(iii) Consider random epochs of records from $\mathcal{G}$, with the same stored-child bootstrap and path-count normalization, and write $q_{t+1} = q_t + \alpha_t(b_t - H_t q_t)$. Then

$$\|H_t\|_\infty \le 1 + \gamma, \qquad \|b_t\|_\infty \le R_{\max} + \gamma G_{\max}. \tag{11}$$

Let $\mathcal{F}_t$ be the history before epoch t, so that q_t is $\mathcal{F}_t$-measurable. Assume there exist deterministic, time-independent $\bar{H}, \bar{b}$ such that

$$\mathbb{E}[H_t \mid \mathcal{F}_t] = \bar{H}, \qquad \mathbb{E}[b_t \mid \mathcal{F}_t] = \bar{b}, \tag{12}$$

and suppose

$$d_i = \mathbb{E}\left[\frac{n_{t,i}}{N_{t,i}} \,\middle|\, \mathcal{F}_t\right] > 0.$$

Here $n_{t,i}$ and $N_{t,i}$ are the epoch's direct-root and path counts, respectively, with $n_{t,i}/N_{t,i} = 0$ when $N_{t,i} = 0$. If q_0 is finite and deterministic and the deterministic steps satisfy $\alpha_t > 0$, $\sum_t \alpha_t = \infty$, and $\sum_t \alpha_t^2 < \infty$, then

$$q_t \longrightarrow \bar{H}^{-1}\bar{b} \qquad \textit{almost surely}.$$

In particular, $\sup_t \|q_t\| < \infty$ almost surely. If $\bar{H} = H$ and $\bar{b} = b$, the limit is $q^\dagger_{\mathcal{G}}$.

Proof. For each root u,

$$\sum_{j:u\in A_j} \big(R_j + \gamma v_{c_j}(q) - q_u\big) = n_u\big(\bar{r}_u + \gamma(Pq)_u - q_u\big).$$

Since $(B^d)_{iu}$ counts length-d paths from i to u, Eqs. 7 and 8 imply

$$h(q) = C^{-1}WD_n(\bar{r} + \gamma Pq - q).$$

This gives Eq. 9.

The matrices B and P are strictly upper triangular, while W is upper triangular with unit diagonal. Since C and D_n have positive diagonals, $C^{-1}WD_n$ is invertible. Hence

$$h(q) = 0 \quad \Longleftrightarrow \quad \bar{r} + \gamma Pq - q = 0.$$

As $P^n = 0$, we have $(I - \gamma P)^{-1} = \sum_{k=0}^{n-1} \gamma^k P^k$, which proves Eq. 10. The depth-zero update has $W = I$ and $C = D_n$, and therefore equals $\mathcal{T}_{\mathcal{G}} q - q$. Replacing C by any positive diagonal matrix leaves the zero of h unchanged. This proves (i).

All factors in Eq. 9 are upper triangular, and W and $I - \gamma P$ have unit diagonal. Thus

$$H_{ii} = \frac{n_i}{N_i} > 0, \qquad \operatorname{spec}(H) = \{n_i/N_i : 1 \le i \le n\} \subset (0, 1].$$

In particular, H is positive stable. For a constant step,

$$q_{t+1} - q^\dagger_{\mathcal{G}} = (I - \alpha H)(q_t - q^\dagger_{\mathcal{G}}).$$

The eigenvalues of $I - \alpha H$ are $1 - \alpha n_i/N_i$. Under the stated restriction on α, their moduli are strictly less than one, so $(I - \alpha H)^t \to 0$. The diminishing-step case of (ii) is a special case of (iii), proved below.

For the stochastic recursion, define

$$z_{t,j,u}(i) = \begin{cases} \dfrac{1}{N_{t,i}} \displaystyle\sum_{p\in\mathcal{P}_{\mathcal{L}}(u\rightsquigarrow i)} \beta^{|p|}, & N_{t,i} > 0, \\ 0, & N_{t,i} = 0. \end{cases} \tag{13}$$

Each term $\beta^{|p|}$ in Eq. 13 lies in $[0, 1]$, whereas $N_{t,i}$ counts paths without weighting. Consequently,

$$0 \le \sum_j \sum_{u\in A_{t,j}} z_{t,j,u}(i) \le 1.$$

Let e_u denote the uth coordinate vector and write $v_{c_j}(q) = \psi_{t,j}^\top q + \kappa_{t,j}$, with $(\psi_{t,j}, \kappa_{t,j}) = (e_{c_j}, 0)$ for a nonterminal child and $(0, g_{c_j})$ otherwise. Then $\|\psi_{t,j}\|_1 \le 1$, $|\kappa_{t,j}| \le G_{\max}$, and

$$H_t = \sum_j \sum_{u \in A_{t,j}} z_{t,j,u}(e_u - \gamma\psi_{t,j})^\top,$$
$$b_t = \sum_j \sum_{u \in A_{t,j}} z_{t,j,u}(R_{t,j} + \gamma\kappa_{t,j}). \tag{14}$$

The absolute sum of row i of H_t is bounded by

$$\sum_{j,u} z_{t,j,u}(i)\|e_u - \gamma\psi_{t,j}\|_1 \le (1+\gamma)\sum_{j,u} z_{t,j,u}(i) \le 1 + \gamma.$$

The analogous bound for b_t proves Eq. 11. It also follows that

$$\|b_t - H_t q\|_\infty \le R_{\max} + \gamma G_{\max} + (1+\gamma)\|q\|_\infty.$$

In Eq. 14, $z_{t,j,u}(i) > 0$ only if $i = u$ or i is an ancestor of u. Nonzero columns of the corresponding summand can occur only at u and, if nonterminal, its strictly newer child. Thus H_t is upper triangular, and its diagonal receives only the zero-length contribution:

$$(H_t)_{ii} = \frac{n_{t,i}}{N_{t,i}}, \qquad \bar{H}_{ii} = d_i > 0.$$

Hence $\bar{H}$ is upper triangular and positive stable.

It remains to prove almost-sure convergence. We use a standard stochastic-approximation argument (Borkar & Meyn, 2000). Set $q^\dagger = \bar{H}^{-1}\bar{b}$, $e_t = q_t - q^\dagger$, and

$$M_{t+1} = (b_t - H_t q_t) - (\bar{b} - \bar{H} q_t).$$

By Eq. 12, $\mathbb{E}[M_{t+1} \mid \mathcal{F}_t] = 0$. Equation 11 and equivalence of finite-dimensional norms give, for a deterministic $K_0 < \infty$,

$$\mathbb{E}[\|M_{t+1}\|_2^2 \mid \mathcal{F}_t] \le K_0(1 + \|q_t\|_2^2). \tag{15}$$

Positive stability of $\bar{H}$ implies that there exists a symmetric $S \succ 0$ satisfying

$$\bar{H}^\top S + S\bar{H} = I. \tag{16}$$

With $L_t = e_t^\top S e_t$ and $e_{t+1} = e_t - \alpha_t \bar{H} e_t + \alpha_t M_{t+1}$, Eqs. 15 and 16 yield

$$\begin{aligned}\mathbb{E}[L_{t+1} \mid \mathcal{F}_t] &= L_t - \alpha_t\|e_t\|_2^2 + \alpha_t^2 e_t^\top \bar{H}^\top S\bar{H} e_t + \alpha_t^2 \mathbb{E}[M_{t+1}^\top S M_{t+1} \mid \mathcal{F}_t] \\ &\le (1 + K\alpha_t^2)L_t - \alpha_t\|e_t\|_2^2 + K\alpha_t^2 \end{aligned} \tag{17}$$

for a deterministic $K < \infty$. Bounded coefficients and finite deterministic q_0 imply finite second moments at every finite time. Since $\sum_t \alpha_t^2 < \infty$, the almost-supermartingale theorem of Robbins & Siegmund (1971), applied to Eq. 17, gives almost-sure convergence of L_t to a finite limit and

$$\sum_t \alpha_t \|e_t\|_2^2 < \infty \qquad \text{almost surely.} \tag{18}$$

On any sample path where $\lim_t L_t > 0$, the inequality $L_t \le \lambda_{\max}(S)\|e_t\|_2^2$ bounds $\|e_t\|_2^2$ away from zero for all sufficiently large t. Together with $\sum_t \alpha_t = \infty$, this contradicts Eq. 18. Thus $L_t \to 0$ and, since $S \succ 0$, $q_t \to q^\dagger$ almost surely. Boundedness follows.

When $\bar{H} = H$ and $\bar{b} = b$, Eq. 10 gives $q^\dagger = q_{\mathcal{G}}^\dagger$. Taking $H_t = H$ and $b_t = b$ deterministically also proves the remaining assertion in (ii). □

Remark. The matrix $C^{-1}WD_n$ is an invertible preconditioner of $\mathcal{T}_{\mathcal{G}}q - q$, so the full-buffer operator in part (i) preserves $q_{\mathcal{G}}^\dagger$. Independent epochs from a fixed replay law satisfy Eq. 12. With epoch-wise normalization, part (iii) gives convergence to $\bar{H}^{-1}\bar{b}$ under its stated assumptions; this limit equals $q_{\mathcal{G}}^\dagger$ when $\bar{H} = H$ and $\bar{b} = b$.

B.3 Set-Value Approximation Error

Fix a policy π and a task–memory state $(s, \mathcal{M})$. Let $\mathcal{A}_s$ be a finite, nonempty collection of candidate retrieval sets $A \subseteq \mathcal{M}$ satisfying $1 \le |A| \le k$. For $A \in \mathcal{A}_s$, write

$$Q(A) = Q^\pi(s, A; \mathcal{M}), \qquad \widetilde{Q}(A) = \frac{1}{|A|} \sum_{m \in A} q^\pi(m), \qquad \xi(A) = \xi^\pi(s, A; \mathcal{M}).$$

By Eq. 3, $Q(A) = \widetilde{Q}(A) + \xi(A)$. Define $\varepsilon_{\mathrm{int}} = \max_{A \in \mathcal{A}_s} |\xi(A)|$. A uniform bound on the residual over task–memory states and candidate sets can be used in its place.

Proposition B.2 (Set-value approximation)**.** *Let*

$$A^\star \in \arg\max_{A \in \mathcal{A}_s} Q(A), \qquad \widehat{A} \in \arg\max_{A \in \mathcal{A}_s} \widetilde{Q}(A).$$

Then

$$\max_{A \in \mathcal{A}_s} |Q(A) - \widetilde{Q}(A)| \le \varepsilon_{\mathrm{int}}, \qquad 0 \le Q(A^\star) - Q(\widehat{A}) \le 2\varepsilon_{\mathrm{int}}.$$

If

$$Q(A^\star) - Q(A) > 2\varepsilon_{\mathrm{int}} \qquad \textit{for every } A \in \mathcal{A}_s \setminus \{A^\star\},$$

then $\widehat{A} = A^\star$.

Proof. The approximation-error bound follows from the definition of $\varepsilon_{\mathrm{int}}$. Optimality of $A^\star$ gives nonnegative regret. By optimality of $\widehat{A}$ for $\widetilde{Q}$,

$$\begin{aligned} Q(A^\star) - Q(\widehat{A}) &= \xi(A^\star) + \widetilde{Q}(A^\star) - \widetilde{Q}(\widehat{A}) - \xi(\widehat{A}) \\ &\le |\xi(A^\star)| + |\xi(\widehat{A})| \le 2\varepsilon_{\mathrm{int}}. \end{aligned}$$

For every $A \neq A^\star$, the margin condition gives

$$\widetilde{Q}(A^\star) - \widetilde{Q}(A) = Q(A^\star) - Q(A) - \xi(A^\star) + \xi(A) \ge Q(A^\star) - Q(A) - 2\varepsilon_{\mathrm{int}} > 0.$$

Thus $A^\star$ is also the unique maximizer of $\widetilde{Q}$. □

When $\xi(A) = 0$ for every candidate set, the approximation is exact and the selection regret is zero. In particular, for fixed retrieval cardinality k, an additive value $Q(A) = \sum_{m \in A} u(m)$ is represented exactly by choosing $q^\pi(m) = k\, u(m)$.

B.4 Computational Complexity

For a bank of N memories, MemQ stores one Q-value and at most k parent IDs per entry, requiring $O(Nk)$ metadata, or linear space for fixed k. Relative to MemRL, which already maintains per-memory values, the graph edges are the additional persistent state. Update cost depends on the paths traversed: every-path propagation can accumulate $O(|A_j| \sum_{d=0}^{D} k^d)$ contributions per task. The implementation bounds this work through depth truncation and discount-floor pruning.

C Experimental Setup

Baseline implementations. We compare six baselines: **No Memory**, a frozen LLM without retrieved context; **RAG**, cosine-similarity top-k retrieval without quality re-ranking; **Self-RAG** (Asai et al., 2024), query generation and self-evaluation to filter unhelpful memories; **Mem0** (Chhikara et al., 2025), episodic-memory extraction, updates and deletion; **MemP** (Fang et al., 2026), trajectory distillation into procedural instructions and scripts with dynamic update, correction and deprecation; and **MemRL** (Zhang et al., 2026b), single-step EMA of per-memory utility without eligibility traces.

RAG: We use the official MemRL codebase for the RAG experiments because it has already integrated RAG by disabling the value-related hyperparameters and doing raw trajectory saving.

Self-RAG: We initially attempted to reproduce the original method using its released Llama-2 checkpoints, but they did not function reliably in our agentic evaluations. These checkpoints are

fine-tuned on instruction-following and knowledge-intensive data augmented with retrieved passages and reflection tokens (Asai et al., 2024), a training setting that differs from the interactive tasks considered here. We therefore implement its sequence of on-demand retrieval, relevance, support, utility critique, and critique-based selection with the shared frozen backbone, eliciting these decisions through prompting.

Mem0: We use the official mem0 repository for its experiments. All the hyperparameters shared with other baselines align with those baselines.

MemP: We use the official MemRL codebase for the MemP experiments because it has already integrated MemP.

MemRL: We use the official MemRL codebase for the MemRL experiments.

Benchmarks and data splits. Each benchmark has memory-accumulation and held-out splits shared by all methods, with one frozen backbone per benchmark. The interaction, code-generation, and agentic tasks comprise LifeLongAgentBench (LLAB; multi-step OS planning) (Zheng et al., 2025), BFCL v3 (multi-turn function calling) (Patil et al., 2025), LiveCodeBench v6 (code generation) (Jain et al., 2025), DSBench (tool-mediated data science) (Jing et al., 2024), and AutomationBench (workflow automation) (Shepard & Salimans, 2026). Multiple-choice tasks comprise MMMU Pro (multimodal reasoning) (Yue et al., 2025), ERQA (physically grounded reasoning) (Team et al., 2025), and GPQA (expert science questions) (Rein et al., 2024).

Model backbones. We select modality-appropriate backbones to explore variation in MemQ's gains with model scale: GPT-4o-mini for LLAB, Qwen3.5-35B-A3B (FP8) (Qwen Team, 2026) for BFCL, DeepSeek-v4-flash-0731 (284B) (DeepSeek-AI, 2026) for DSBench and AutomationBench, and Gemma-4-E4B-it (Gemma Team, 2026) for the remaining four.

D Additional Learning Curves

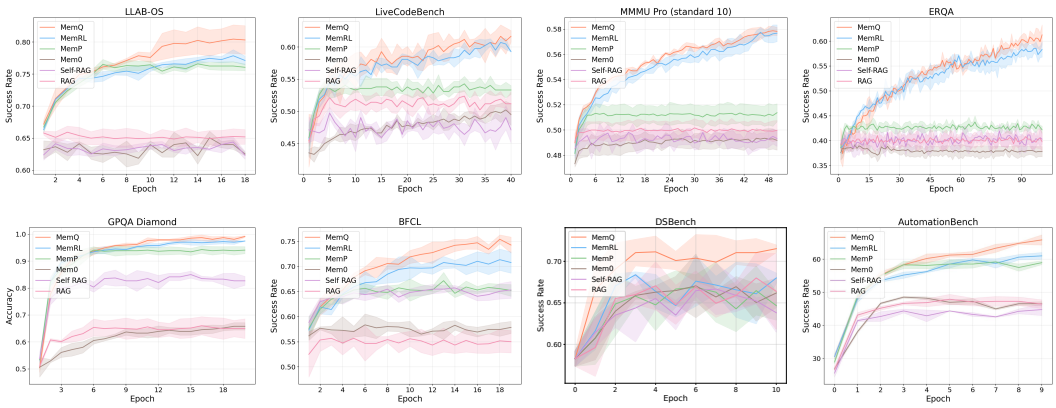


Figure 6: Runtime learning dynamics (success rate vs. epoch) across eight benchmarks. Top row: LLAB, LiveCodeBench, MMMU-Pro, and ERQA. Bottom row: GPQA, BFCL, DSBench, and AutomationBench. DSBench curves show means ± 1 sample SD.

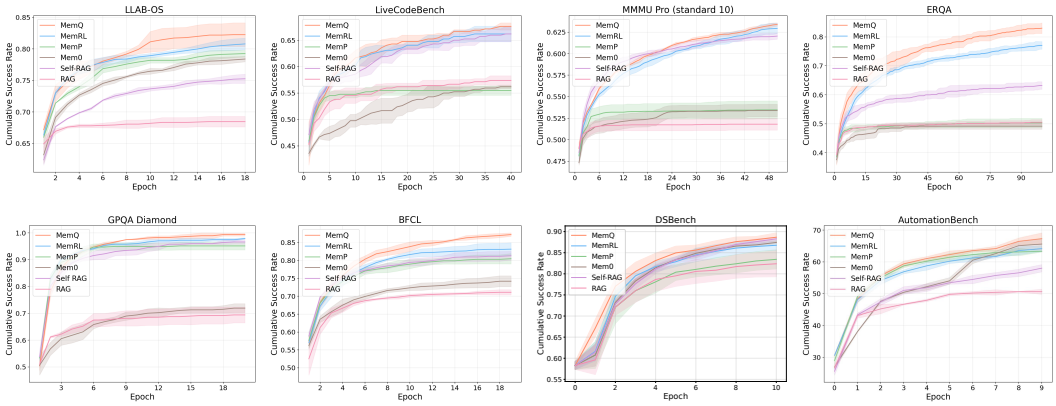


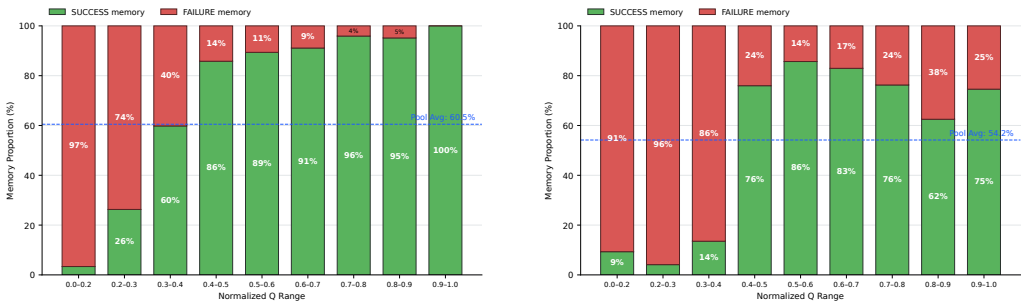
Figure 7: Cumulative success rate over epochs across eight benchmarks, in the same order as Figure 6. CSR counts tasks solved at least once through each epoch and is nondecreasing by construction. DSBench curves show means ± 1 sample SD.

E Additional AutomationBench Value Analysis

Complementing the retrieval-time analysis in Section 5.4, we examine checkpoint memory composition by Q range. Each memory is labeled by whether its originating trajectory succeeded.

In MemQ's 4,800-memory checkpoint, successful trajectories account for 60.5% of the bank. Their proportions increase to 95.9% and 95.2% in the normalized Q ranges 0.7–0.8 and 0.8–0.9 (Figure 8a), leaving 4.1% and 4.8% failed trajectories in those ranges. High-Q ranges thus include memories from both successful and failed trajectories. The highest range, 0.9–1.0, contains only eight memories, all successful.

Figure 8b reports the corresponding composition of MemRL's seed-43 checkpoint at E9, whose overall successful-memory proportion is 54.2%. The two checkpoint views show how trajectory outcomes are distributed across Q ranges in each memory bank.



(a) MemQ checkpoint composition.

(b) MemRL checkpoint composition.

Figure 8: Successful and failed memories by normalized Q range on AutomationBench. MemQ uses a 4,800-memory checkpoint; MemRL uses seed 43 at E9. Dashed lines indicate each checkpoint's overall successful-memory proportion, 60.5% and 54.2%, respectively.

F Additional Ablation Studies

F.1 Q-Value Initialization

Table 3 compares parent-mean initialization with fixed values of 0.5 and 0 under the same within-benchmark ablation pipeline. Parent mean gives the best evaluation SR in both benchmarks. On

LiveCodeBench, a fixed value of 0.5 instead gives higher final and cumulative SR. Parent-mean initialization therefore offers a useful cold-start rule, with an observed tradeoff between runtime and held-out performance. Results are point estimates from the initialization ablation runs.

Table 3: Q-value initialization ablation results (%).

Benchmark	Q initialization	CSR	Final SR	Evaluation SR
LiveCodeBench	Mean parent Q	67.62	61.79	**51.43**
	0.5	**69.29**	**63.57**	45.71
	0	57.86	52.86	34.29
BFCL	Mean parent Q	**86.43**	**75.04**	**62.83**
	0.5	83.92	70.60	60.73
	0	82.66	68.34	57.76

F.2 Trace-Decay Sensitivity on LiveCodeBench

Figure 9 reports TD diagnostics for the LiveCodeBench λ sweep at $\gamma = 0.3$. The corresponding Train SR curves appear in Figure 4 (bottom).

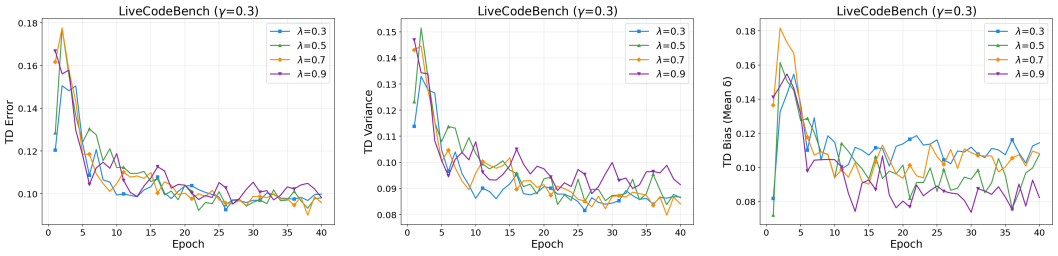


Figure 9: TD error, TD variance, and TD bias for the LiveCodeBench λ sweep at $\gamma = 0.3$.

F.3 Additional γ Sweeps and TD Diagnostics

Figure 10 complements Figure 4 (top) with $\lambda = 0.3$. LiveCodeBench no longer shows the marked late-stage decline at $\gamma = 0.9$ seen with $\lambda = 0.95$. On BFCL, intermediate discounts remain competitive with the largest tested discount. These additional sweeps show how the response to γ changes with trace decay.

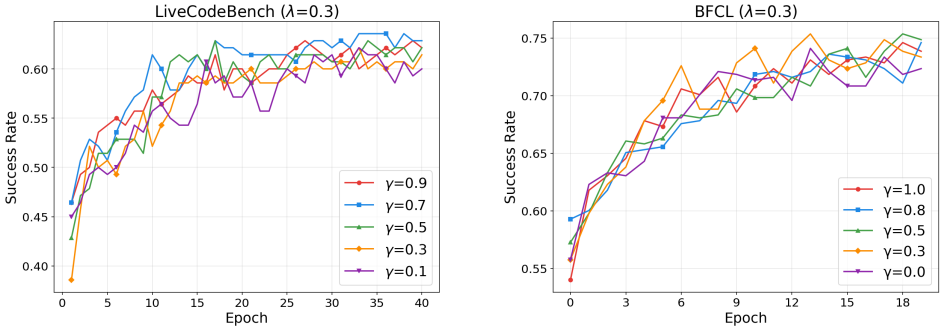


Figure 10: Additional Train SR curves under varying γ, with $\lambda = 0.3$ on LiveCodeBench (left) and BFCL (right).

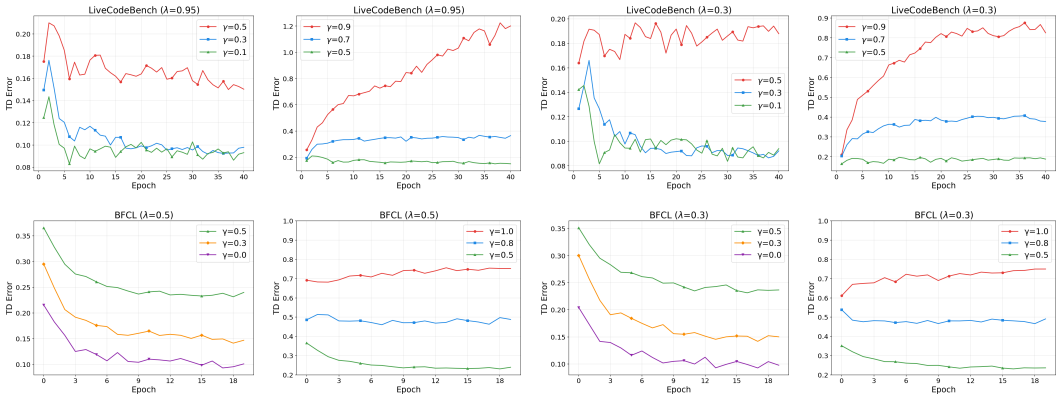


Figure 11: TD error under different γ on LiveCodeBench (top) and BFCL (bottom).

F.4 λ Ablation on BFCL

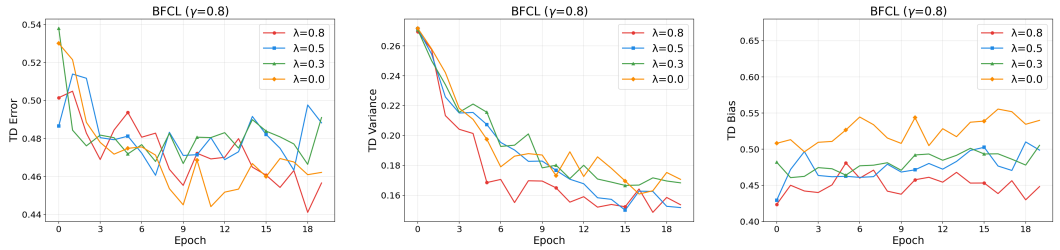


Figure 12: TD error, TD variance, and TD bias for the BFCL λ sweep at $\gamma = 0.8$. The corresponding Train SR curves appear in Figure 4 (bottom).

F.5 D Ablation on BFCL

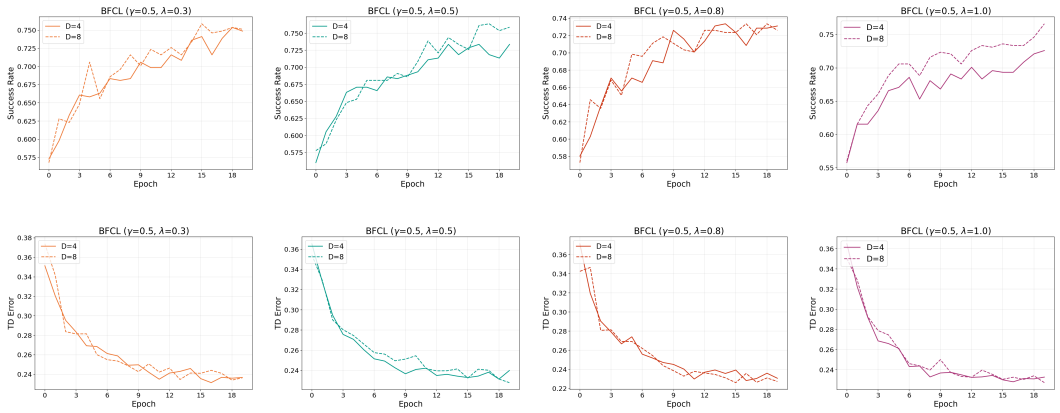


Figure 13: SR (top row) and TD error (bottom row) under different D on BFCL.

F.6 Similarity–Q Retrieval Weight Ablations

Tables 4 and 5 report the complete sweep used to test whether semantic similarity or learned Q-values alone explain MemQ's retrieval behavior. The best weight depends on the benchmark and metric, but mixed policies consistently provide stronger overall tradeoffs than either endpoint. These ablations were conducted separately from the main experiments. The best-performing weights identified here were not adopted in the main experiments, which use the configurations reported in Appendix G.

Table 4: Similarity/Q retrieval-weight ablation on LiveCodeBench (%).

w_s	w_q	CSR	Final SR	Evaluation SR
0.0	1.0	64.29	52.86	45.71
0.2	0.8	67.14	58.57	42.86
0.4	0.6	65.71	58.57	48.57
0.6	0.4	**67.62**	**61.79**	**51.43**
0.8	0.2	62.14	60.00	48.57
1.0	0.0	58.57	55.00	42.86

Table 5: Similarity/Q retrieval-weight ablation on BFCL (%).

w_s	w_q	CSR	Final SR	Evaluation SR
0.0	1.0	85.43	61.56	59.41
0.2	0.8	86.18	67.17	**63.70**
0.4	0.6	**86.93**	**73.60**	61.39
0.6	0.4	86.68	71.77	60.07
0.8	0.2	83.17	68.34	63.04
1.0	0.0	78.64	65.75	56.77

G HYPERPARAMETERS

Hyperparameter configurations. We report hyperparameters for each benchmark separately, since key parameters (γ, λ, w_s, w_q, Batch size, θ_{sim}, Initial Q, epochs) vary across benchmarks. Table 6 lists settings shared across all benchmarks. Tables 7–14 give per-benchmark configurations for each method.

Table 6: Settings shared across all benchmarks and methods.

Parameter	**Value**	**Description**
normalize_dq	true	Normalize Q-value deltas
success_reward	1.0	Reward for correct answer
failure_reward	0.0	Reward for wrong answer
normalize_sim	false	Raw similarity scores (already in $[0, 1]$)
normalize_q	true (value-based methods)	Normalize Q for retrieval scoring
temperature	0.0	Deterministic LLM generation
D	4	Maximum backward-path depth for DAG traversal
Discount floor	$(\gamma\lambda)^d < 10^{-12}$	Prune deep provenance paths

LifeLongAgentBench. LLM backbone: GPT-4o-mini. Embedding model: `text-embedding-3-large` ($d = 3072$). The artifact uses 350 training instances and 150 held-out evaluation instances; every method uses this split, and hyperparameters are not tuned on the held-out set. Max agent steps per task: 15.

Table 7: LifeLongAgentBench hyperparameters.

	MemQ	**MemRL**	**MemP**	**Mem0**	**Self-RAG**	**RAG**
α	0.3	0.3	N/A	N/A	N/A	N/A
γ	0.5	N/A	N/A	N/A	N/A	N/A
λ	0.7	N/A	N/A	N/A	N/A	N/A
ε	0.01	0.01	0.01	0.0	0.01	0.01
k_{ret}	10	10	10	5	10	10
k_{top}	5	5	5	5	5	5
θ_{sim}	0.5	0.5	0.5	0.0	0.5	0.5
w_s	0.5	0.5	N/A	N/A	N/A	N/A
w_q	0.5	0.5	N/A	N/A	N/A	N/A
Batch size	64	64	64	64	64	64
Max tokens	10240	10240	10240	10240	10240	10240
Initial Q	0.66	0.66	N/A	N/A	N/A	N/A
Build	proceduralization	proceduralization	proceduralization	trajectory	trajectory	trajectory
Update	adjustment	adjustment	adjustment	vanilla	vanilla	vanilla
Epochs	20	20	20	20	20	20

LiveCodeBench. LLM backbone: Gemma-4-E4B-it. Embedding model: Qwen3-Embedding-8B. 175 problems (140 train / 35 evaluation).

Table 8: LiveCodeBench hyperparameters.

	MemQ	**MemRL**	**MemP**	**Mem0**	**Self-RAG**	**RAG**
α	0.3	0.3	N/A	N/A	N/A	N/A
γ	0.3	N/A	N/A	N/A	N/A	N/A
λ	0.95	N/A	N/A	N/A	N/A	N/A
ε	0.0	0.0	0.0	0.0	0.0	0.0
k_{ret}	5	5	5	5	5	5
k_{top}	3	3	3	5	3	3
θ_{sim}	0.6	0.6	0.6	0.0	0.6	0.6
w_s	0.4	0.4	N/A	N/A	N/A	N/A
w_q	0.6	0.6	N/A	N/A	N/A	N/A
Batch size	32	32	32	32	32	32
Max tokens	10240	10240	10240	10240	10240	10240
Initial Q	0.37	0.37	N/A	N/A	N/A	N/A
Build	proceduralization	proceduralization	proceduralization	trajectory	trajectory	trajectory
Update	adjustment	adjustment	adjustment	vanilla	vanilla	vanilla
Epochs	40	40	40	40	40	40

MMMU Pro. LLM backbone: Gemma-4-E4B-it. Embedding model: Qwen3-Embedding-8B. 1730 problems (1384 train / 346 evaluation).

Table 9: MMMU Pro hyperparameters.

	MemQ	**MemRL**	**MemP**	**Mem0**	**Self-RAG**	**RAG**
α	0.3	0.3	N/A	N/A	N/A	N/A
γ	0.5	N/A	N/A	N/A	N/A	N/A
λ	0.7	N/A	N/A	N/A	N/A	N/A
ε	0.0	0.0	0.0	0.0	0.0	0.0
k_{ret}	5	5	5	5	5	5
k_{top}	3	3	3	5	3	3
θ_{sim}	0.55	0.55	0.55	0.0	0.50	0.55
w_s	0.5	0.5	N/A	N/A	N/A	N/A
w_q	0.5	0.5	N/A	N/A	N/A	N/A
Batch size	128	128	128	128	128	128
Max tokens	4096	4096	4096	4096	4096	4096
Initial Q	0.526	0.526	N/A	N/A	N/A	N/A
Build	proceduralization	proceduralization	proceduralization	trajectory	trajectory	trajectory
Update	adjustment	adjustment	adjustment	vanilla	vanilla	vanilla
Epochs	50	50	50	50	50	50

ERQA. LLM backbone: Gemma-4-E4B-it. Embedding model: Qwen3-Embedding-8B. 400 problems (320 train / 80 evaluation).

Table 10: ERQA hyperparameters.

	MemQ	**MemRL**	**MemP**	**Mem0**	**Self-RAG**	**RAG**
α	0.3	0.3	N/A	N/A	N/A	N/A
γ	0.5	N/A	N/A	N/A	N/A	N/A
λ	0.99	N/A	N/A	N/A	N/A	N/A
ε	0.0	0.0	0.0	0.0	0.0	0.0
k_{ret}	5	5	5	5	5	5
k_{top}	3	3	3	5	3	3
θ_{sim}	0.6	0.6	0.6	0.0	0.5	0.6
w_s	0.4	0.5	N/A	N/A	N/A	N/A
w_q	0.6	0.5	N/A	N/A	N/A	N/A
Batch size	64	64	64	64	64	64
Max tokens	4096	4096	4096	4096	4096	4096
Initial Q	0.37	0.37	N/A	N/A	N/A	N/A
Build	proceduralization	proceduralization	proceduralization	trajectory	trajectory	trajectory
Update	adjustment	adjustment	adjustment	vanilla	vanilla	vanilla
Epochs	100	100	100	100	100	100

BFCL. LLM backbone: Qwen3.5-35B-A3B with FP8 quantization. Embedding model: Qwen3-Embedding-8B. 500 problems (400 train / 100 test).

Table 11: BFCL hyperparameters.

	MemQ	**MemRL**	**MemP**	**Mem0**	**Self-RAG**	**RAG**
α	0.3	0.3	N/A	N/A	N/A	N/A
γ	0.5	N/A	N/A	N/A	N/A	N/A
λ	0.8	N/A	N/A	N/A	N/A	N/A
ε	0.01	0.01	0.01	0.0	0.01	0.01
k_{ret}	10	10	10	5	10	5
k_{top}	5	5	5	5	5	5
θ_{sim}	0.5	0.5	0.5	0.0	0.5	0.5
w_s	0.7	0.7	N/A	N/A	N/A	N/A
w_q	0.3	0.3	N/A	N/A	N/A	N/A
Batch size	100	100	100	100	100	100
Max tokens	4096	4096	4096	4096	4096	4096
Initial Q	0.5	0.5	N/A	N/A	N/A	N/A
Build	proceduralization	proceduralization	proceduralization	trajectory	trajectory	trajectory
Update	adjustment	adjustment	adjustment	vanilla	vanilla	vanilla
Epochs	20	20	20	20	20	20

GPQA Diamond. LLM backbone: Gemma-4-E4B-it. Embedding model: Qwen3-Embedding-8B. 198 problems (158 train / 40 test).

Table 12: GPQA Diamond hyperparameters.

	MemQ	**MemRL**	**MemP**	**Mem0**	**Self-RAG**	**RAG**
α	0.3	0.3	N/A	N/A	N/A	N/A
γ	0.5	N/A	N/A	N/A	N/A	N/A
λ	0.7	N/A	N/A	N/A	N/A	N/A
ε	0.01	0.01	0.01	0.0	0.01	0.01
k_{ret}	10	10	10	5	10	10
k_{top}	5	5	5	5	5	5
θ_{sim}	0.5	0.5	0.5	0.0	0.5	0.5
w_s	0.5	0.7	N/A	N/A	N/A	N/A
w_q	0.5	0.3	N/A	N/A	N/A	N/A
Batch size	40	100	100	100	100	100
Max tokens	16384	16384	16384	16384	16384	16384
Initial Q	0.5	0.5	N/A	N/A	N/A	N/A
Build	proceduralization	proceduralization	proceduralization	trajectory	trajectory	trajectory
Update	adjustment	adjustment	adjustment	vanilla	vanilla	vanilla
Epochs	20	20	20	20	20	20

DSBench. LLM backbone: DeepSeek-v4-flash-0731. Embedding model: Qwen3-Embedding-8B. Training: 350 tasks from 30 challenges; held-out evaluation: 116 tasks from 8 challenges.

Table 13: DSBench hyperparameters.

	MemQ	**MemRL**	**MemP**	**Mem0**	**Self-RAG**	**RAG**
α	0.3	0.3	N/A	N/A	N/A	N/A
γ	0.9	N/A	N/A	N/A	N/A	N/A
λ	0.5	N/A	N/A	N/A	N/A	N/A
ε	0.1	0.1	0.0	0.0	0.0	0.0
k_{ret}	3	3	3	3	3	3
k_{top}	5	5	5	5	5	5
θ_{sim}	0.0	0.0	0.0	0.0	0.0	0.0
w_s	0.5	0.5	N/A	N/A	N/A	N/A
w_q	0.5	0.5	N/A	N/A	N/A	N/A
Batch size	30	30	30	30	30	30
Max tokens	32768	32768	32768	32768	32768	32768
Initial Q	0.66	0	N/A	N/A	N/A	N/A
Build	proceduralization	proceduralization	proceduralization	trajectory	trajectory	trajectory
Update	adjustment	adjustment	adjustment	vanilla	vanilla	vanilla
Epochs	10	10	10	10	10	10

AutomationBench. LLM backbone: DeepSeek-v4-flash-0731 (284B). Embedding model and other unspecified settings follow DSBench. The split contains 480 training tasks and 120 held-out/test tasks.

Table 14: AutomationBench hyperparameters.

	MemQ	**MemRL**	**MemP**	**Mem0**	**Self-RAG**	**RAG**
α	0.3	0.3	N/A	N/A	N/A	N/A
γ	0.5	N/A	N/A	N/A	N/A	N/A
λ	0.8	N/A	N/A	N/A	N/A	N/A
ε	0.1	0.1	0.0	0.0	0.0	0.0
k_{ret}	3	3	3	3	3	3
k_{top}	5	5	5	5	5	5
θ_{sim}	0.0	0.0	0.0	0.0	0.0	0.0
w_s	0.7	0.5	N/A	N/A	N/A	N/A
w_q	0.3	0.5	N/A	N/A	N/A	N/A
Batch size	50	50	50	50	50	50
Max tokens	16384	16384	16384	16384	16384	16384
Initial Q	0.66	0	N/A	N/A	N/A	N/A
Build	proceduralization	proceduralization	proceduralization	trajectory	trajectory	trajectory
Update	adjustment	adjustment	adjustment	vanilla	vanilla	vanilla
Epochs	10	10	10	10	10	10

H Additional Experimental Results

Tables 15 and 16 give the coverage and token statistics discussed in Section 5.4. Table 17 reports the complete multiple-choice results discussed in Section 5.5.

Table 15: Historical coverage and final-epoch success (%). Values are taken from Table 2; the gap is CSR minus final SR, in percentage points.

Benchmark	Method	CSR	Final SR	Gap ↓
BFCL	MemP	80.15	65.91	<u>14.24</u>
	MemRL	<u>85.85</u>	<u>71.27</u>	14.58
	MemQ	**86.43**	**75.04**	**11.39**
AutomationBench	MemP	63.33	58.96	4.37
	MemRL	<u>64.17</u>	<u>61.01</u>	<u>3.16</u>
	MemQ	**67.29**	**65.83**	**1.46**

Table 16: DSBench LLM token consumption over 10 training epochs, excluding evaluation and embedding tokens (mean ± sample SD across three seeds). All values are in millions of tokens. Counts include agent/judge and memory LLM calls; Input and Output denote prompt and completion tokens.

Method	Input (w. memory) ↓	Output ↓	Total ↓
MemP	340.01 ± 25.82	33.34 ± 1.31	373.35 ± 25.78
MemRL	317.11 ± 28.37	35.27 ± 1.11	352.38 ± 27.65
MemQ	$\mathbf{280.68 \pm 37.85}$	$\mathbf{32.35 \pm 2.73}$	$\mathbf{313.03 \pm 40.01}$

Runtime and held-out comparisons on multiple-choice tasks. The relationship between runtime and held-out performance varies across these tasks. On GPQA, MemQ reaches 98.52% final SR and 98.95% CSR, compared with MemRL's 97.05% and 97.89%; their held-out scores are 61.67% and 60.83%. On MMMU Pro, MemQ and MemRL differ by less than one percentage point on all three metrics. ERQA shows larger improvements over MemRL: 4.58 pp in held-out SR, 4.22 pp in final SR, and 5.94 pp in CSR. Its smaller 0.83 pp lead over Self-RAG reflects the stronger held-out baseline on this task.

Table 17: Multiple-choice reasoning results with Gemma-4-E4B-it (%). For each benchmark, the three rows report Eval SR, Final SR, and CSR, respectively. Entries are mean ± sample SD. Approximate 95% CIs are shown below the averages. Best in **bold**; second-best underlined.

Benchmark	No Mem.	RAG	Self-RAG	Mem0	MemP	MemRL	*MemQ*
MMMU Pro Multimodal	$48.46_{\pm 0.76}$ $48.92_{\pm 0.50}$ –	$53.76_{\pm 1.04}$ $49.98_{\pm 0.79}$ $51.81_{\pm 0.68}$	$53.57_{\pm 0.84}$ $49.33_{\pm 0.62}$ $62.04_{\pm 0.36}$	$49.90_{\pm 1.37}$ $49.18_{\pm 0.77}$ $53.40_{\pm 0.72}$	$54.24_{\pm 1.42}$ $51.37_{\pm 0.72}$ $53.47_{\pm 1.04}$	$53.66_{\pm 1.01}$ $57.68_{\pm 0.66}$ $62.93_{\pm 0.61}$	$\mathbf{54.43}_{\pm 0.88}$ $\mathbf{57.83}_{\pm 0.32}$ $\mathbf{63.39}_{\pm 0.19}$
ERQA Embodied Reasoning	$41.67_{\pm 1.24}$ $39.38_{\pm 0.00}$ –	$44.58_{\pm 1.44}$ $39.61_{\pm 1.66}$ $50.78_{\pm 0.90}$	$50.42_{\pm 2.89}$ $40.31_{\pm 1.02}$ $62.42_{\pm 1.80}$	$48.33_{\pm 1.91}$ $37.81_{\pm 1.02}$ $49.17_{\pm 1.03}$	$45.00_{\pm 3.31}$ $41.80_{\pm 0.95}$ $50.31_{\pm 1.55}$	$46.67_{\pm 1.91}$ $57.11_{\pm 2.50}$ $77.08_{\pm 1.31}$	$\mathbf{51.25}_{\pm 2.17}$ $\mathbf{61.33}_{\pm 1.76}$ $\mathbf{83.02}_{\pm 1.85}$
GPQA Diamond Science QA	$47.50_{\pm 3.01}$ $52.53_{\pm 1.10}$ –	$58.33_{\pm 3.82}$ $65.19_{\pm 3.62}$ $68.99_{\pm 3.23}$	$58.33_{\pm 3.82}$ $85.02_{\pm 1.30}$ $95.99_{\pm 0.30}$	$58.33_{\pm 1.18}$ $64.86_{\pm 1.31}$ $71.94_{\pm 1.66}$	$59.17_{\pm 1.44}$ $93.46_{\pm 1.30}$ $95.99_{\pm 1.49}$	$60.83_{\pm 5.20}$ $97.05_{\pm 0.52}$ $97.89_{\pm 0.30}$	$\mathbf{61.67}_{\pm 2.89}$ $\mathbf{98.52}_{\pm 1.37}$ $\mathbf{98.95}_{\pm 0.60}$
Average (Eval SR)	45.88 [43.81, 47.95]	52.22 [49.58, 54.86]	54.11 [51.50, 56.72]	52.19 [50.89, 53.48]	52.80 [50.62, 54.99]	53.72 [50.04, 57.40]	**55.78** **[53.82, 57.75]**
Average (Final SR)	46.94 [46.17, 47.72]	51.59 [49.12, 54.07]	58.22 [57.33, 59.11]	50.62 [49.72, 51.52]	62.21 [61.33, 63.09]	70.61 [68.78, 72.45]	**72.56** **[71.35, 73.77]**
Average (CSR)	– –	57.19 [54.84, 59.55]	73.48 [72.11, 74.86]	58.17 [57.08, 59.26]	66.59 [65.44, 67.74]	79.30 [78.41, 80.19]	**81.79** **[80.43, 83.15]**

I Extended Related Work

I.1 Self-Evolving Memory Agents

Early memory-augmented agents rely on fixed heuristic retrieval—embedding similarity or hand-crafted scores (Park et al., 2023; Packer et al., 2024; Shinn et al., 2023; Zhao et al., 2024; Wang et al., 2024; Zhong et al., 2024; Kynoch et al., 2023; Sumers et al., 2024)—with no learning signal from task outcomes. More recent work makes memory *self-evolving* along two paradigms. *Parametric approaches* (Yan et al., 2026; Zhang et al., 2026e; Ma et al., 2026a; Shen et al., 2026; Zhang et al., 2026c; Zhou et al., 2025b; Yue et al., 2026) learn network parameters for memory operations, requiring gradient-based optimization. *Non-parametric approaches* avoid weight updates, attaching value estimates or update rules directly to memory entries: REMEMBERER (Zhang et al., 2023) equips a frozen LLM with experience memory updated via RL without weight modification; MemRL (Zhang et al., 2026b) further attaches Q-values with single-step EMA ($\gamma = 0$); Memento (Zhou et al., 2025a) learns a case-selection policy with memory rewriting; other systems employ rule-based curation (Chhikara et al., 2025), cognitive self-organization (Ma et al., 2026b), meta-evolution (Zhang et al., 2025), learnable skills (Zhang et al., 2026a), online experience weighting (Zhang et al., 2026d), Hebbian graphs (Zhu et al., 2026), utility-based pruning (Cao et al., 2026), or procedural memory distillation (Fang et al., 2026). Among the closest methods considered here, learning or selection remains local to a memory operation or current retrieval event. MemQ instead studies how delayed feedback can be propagated across recorded memory-creation exposures while retaining the non-parametric, frozen-LLM setting.

I.2 Reinforcement Learning for Memory

RL for memory spans from episodic control methods (Blundell et al., 2016; Pritzel et al., 2017; Lin et al., 2018) and differentiable memory architectures (Graves et al., 2014; Guu et al., 2020; Schaul et al., 2016) to recent LLM-agent-specific methods. The parametric methods in §I.1 train neural policies for memory operations. On the non-parametric side, MemRL uses TD(0) with $\gamma = 0$, and Memento 2 (Wang, 2026) optimizes retrieval via supervised learning within the Reflected MDP formalism—absorbing the frozen LLM into environment dynamics so that the retrieval policy becomes the sole decision variable and the memory store a valid MDP state. Other work applies RL to memory-augmented retrieval (Yuan et al., 2026; Ouyang et al., 2026; Wei et al., 2025), focusing on what to store and when to retrieve. Classical trace theory (Sutton, 1988; Singh & Sutton, 1996; Sutton & Barto, 2018; Peng & Williams, 1996; Seijen & Sutton, 2014; Schulman et al., 2016) propagates delayed credit over temporal steps. MemQ transfers this update principle to a structural domain: traces follow recorded provenance exposures and use DAG depth as the decay distance.