MEMENTO: CONTINUAL LEARNING OF LLM AGENTS VIA STATEFUL REFLECTIVE MEMORY

Anonymous authors
Paper under double-blind review

ABSTRACT

We present a unified method for continual learning of large language model (LLM) agents that fully eliminates the need to fine-tune the underlying model. Existing paradigms are either rigid (handcrafted reflection workflows) or computationally costly (gradient updates of LLM parameters), and neither scales to long-horizon tasks where environments drift faster than models can be retrained. We argue that the right primitive is reflection: revisiting past experience to adjust future action selection. We formalise this primitive as a *Stateful Reflective Decision Process* (SRDP), in which the agent alternately writes new experiences to an episodic memory and reads relevant cases to guide decisions. By augmenting the state space with the memory, we lift the SRDP to a *Reflected Markov Decision Process* amenable to classical control analysis. We then adapt KL-regularised policy iteration to this memory setting as *Read–Write Reflective Learning*, integrating Parzen-window retrieval with closed-form soft updates, and prove (i) convergence under bounded rewards and a fixed memory, (ii) two-time-scale convergence when memory evolves slowly, and (iii) asymptotic optimality as the memory densely covers the state space. We instantiate this method as a planner–executor Deep Research agent, *Memento*, with both non-parametric and parametric memory variants. Empirically, *Memento* statistically ties the strongest RL-trained baseline on the seven-dataset DeepResearcher suite without finetuning LLMs, and wins the average on five multimodal benchmarks. Furthermore, in end-to-end system-level comparison, *Memento* reaches Top-1 on the GAIA validation leaderboard, and surpasses prior agents on SimpleQA, HLE, and UIS-QA.

1 INTRODUCTION

Large language model (LLM) agents augment LLMs with tool use, planning, and persistent state, and have rapidly become a leading framework for long-horizon problem solving across deep research, code generation, and embodied control (Wei et al., 2022; Yao et al., 2023; Christianos et al., 2023; Huang et al., 2025). Unlike static prompt–completion systems, LLM agents act, observe, and revise over long horizons, often interacting with hundreds of heterogeneous tools per task. This creates a structural tension intrinsic to the agentic setting: tasks arrive online and environments drift between deployments, yet updating LLM parameters remains prohibitively expensive on the timescale at which operational experience accrues. The central question is how an agent can incorporate the experience it accrues after deployment, when retraining is too costly to keep pace.

Current LLM agents respond along one of two paradigms, each structurally limited. Static-workflow approaches hard-code reasoning steps in advance (Wu et al., 2023; ByteDance, 2025; Camel-AI, 2025), yielding systems that are predictable but rigid; once deployed, they cannot adapt to environmental drift or absorb new skills. Parameter-tuning approaches instead train the LLM via supervised fine-tuning or reinforcement learning (Schick et al., 2023; Jin et al., 2025; Zheng et al., 2025a; Song et al., 2025), gaining flexibility at prohibitive compute, data, and catastrophic-forgetting cost. Both treat learning as something that happens before deployment rather than during it, separated from the agent's actual operational experience.

The dominant deployed regime is the frozen-LLM agent: parameters are fixed after pre-training, and all adaptation must come from the input. Several mechanisms exploit this input channel for short-horizon adaptation: in-context learning (Brown et al., 2020), ReAct-style feedback loops (Yao

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.

et al., 2023), and chain-of-thought reasoning (Wei et al., 2022) all change behaviour without changing parameters. Yet none of these mechanisms persists across episodes: every new task starts from the same context window, with no carry-over of skills or mistakes. A scalable form of input-side learning therefore requires an external, growing state that the agent both reads from and writes to over its lifetime, namely an episodic memory sitting between the prompt and the parameters.

Cognitive science suggests this is exactly how humans cope with long-horizon tasks. Each experience is encoded as an episodic trace (Tulving, 1983; Pritzel et al., 2017), selectively replayed when a similar situation recurs (Squire et al., 2015; Gershman & Daw, 2017), and progressively distilled into transferable knowledge through reflective consolidation (Anderson et al., 1997). This is the structure of case-based reasoning (Aamodt & Plaza, 1994; Guo et al., 2024; 2025; 2026): new problems are solved by retrieving and adapting past solutions, with credit assigned by outcome rather than by gradient. The result is sample-efficient, interpretable, and inherently continual.

A growing body of empirical work hints that LLM agents equipped with episodic memory can adapt at deployment time (Shinn et al., 2023; Zhao et al., 2024; Fu et al., 2024; Chhikara et al., 2025; Xu et al., 2025; Wang et al., 2024; Tang et al., 2025; Qiu et al., 2025; Ouyang et al., 2025; Fang et al., 2025). These systems, however, treat memory as a useful engineering artefact: cases are stored, retrieved by similarity, and concatenated into the prompt. None casts retrieval as a learnable action inside a Markovian process, none derives a regularised retrieval policy, and none establishes any formal guarantee about improvement, convergence, or asymptotic optimality. The empirical promise of memory-augmented LLM agents is therefore not yet matched by the analytic backbone that gradient-based learning has long enjoyed.

Contributions. We answer this question through four tightly coupled contributions.

1. **Formalism.** We introduce the Stateful Reflective Decision Process (SRDP, Def. 1), a formal abstraction of an LLM agent that maintains episodic memory through a read–write cycle (§2). By augmenting the state with the memory, we lift the SRDP to a Reflected MDP (Theorem 2), making the entire reflective dynamics amenable to classical control analysis.
2. **Algorithm.** We adapt KL-regularised soft policy iteration (Haarnoja et al., 2017; 2018; Neu et al., 2017) with a Parzen-window retrieval prior (Parzen, 1962; Silverman, 1986) to LLM agent memory, yielding Read–Write Reflective Learning (§3), an instantiation over the Reflected MDP. *The component techniques are classical; the contribution is their composition into a learnable soft retrieval policy over an LLM agent's episodic memory, which to our knowledge is new.* For the dominant single-step planning setting that arises in deep research deployment, the algorithm admits a stable supervised binary classification instantiation, while the general formulation retains its multi-step Reflected MDP analysis.
3. **Theory.** We establish convergence and two-time-scale stability of soft policy iteration under bounded rewards and slowly evolving memory (Theorems 5–7), and our headline guarantee, asymptotic optimality: as memory grows to densely cover the state space, the composite retrieval–LLM policy converges to the optimal policy of the underlying environment MDP (Theorem 3). To our knowledge these are the first convergence guarantees for memory-driven LLM agents.
4. **System and empirical wins.** We instantiate the method as a planner–executor Deep Research agent, *Memento* (§4), which reaches Top-1 on the GAIA validation leaderboard, ties the strongest RL-trained baseline on the seven-dataset DeepResearcher suite at a matched Qwen2.5-7B backbone with no LLM fine-tuning, and wins the average against five memory baselines under a shared Qwen2.5-VL-7B Executor. Ablations confirm both predicted theoretical phenomena: monotone iterative improvement and OOD generalisation that grows with memory coverage.

2 Stateful Reflective Decision Processes

We formalise an intermediate reflection stage in which the agent retrieves relevant experiences from episodic memory M before generating the final output. At each step, the agent first *reads* relevant cases from M to condition its next action and then *writes* the resulting outcome back to M (Figure 1). The classical MDP captures agent–environment interaction in its simplest form but lacks mechanisms for reflection and memory. We extend the formulation by augmenting the agent with an episodic

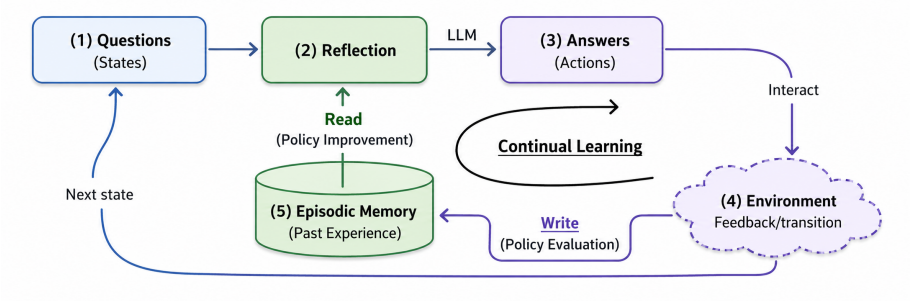


Figure 1: Learning by Stateful Reflection with Memory. The agent realises continual learning without fine-tuning the LLM by iteratively reading from (policy improvement) and writing to (policy evaluation) an evolving episodic memory. The same Read–Write loop instantiates classical case-based reasoning, retrieval-augmented generation, and (as we make precise) soft policy iteration on a Reflected MDP.

memory space M, a memory retrieval policy μ, and a stochastic LLM kernel p_{LLM} that generates context-dependent actions conditioned on both the current environment state and retrieved memory.

Definition 1 (Stateful Reflective Decision Process)**.** *An SRDP is a tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma, \mathcal{M}, p_{\mathrm{LLM}} \rangle$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ the action space, $\mathcal{P} : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ the environment transition kernel, $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ the reward function, $\gamma \in [0, 1)$ the discount, $\mathcal{M} = (\mathcal{S} \times \mathcal{A} \times \mathbb{R})^*$ the memory space, and $p_{\mathrm{LLM}}(a \mid s, c)$ a frozen LLM decision kernel that emits an action a conditioned on the current state s and a retrieved case $c \in \mathcal{M}$.*

Composite policy. A reflective agent factorises its overall policy into a retrieval step and an LLM action step:

$$\pi(a \mid s, M) \;=\; \sum_{c \in M} \mu(c \mid s, M)\, p_{\mathrm{LLM}}(a \mid s, c), \tag{1}$$

where the case-retrieval policy $\mu(\cdot \mid s, M)$ assigns a distribution over the cases in M. The trajectory factorises along five operations (RETRIEVE, REUSE&REVISE, EVALUATE, RETAIN, TRANSITION), formalised in Eq. (7) (Appendix C). Crucially, only μ is learnable; the LLM weights are frozen.

Reflected MDP: lifting to a Markovian process. Because the memory M_t evolves with each interaction, the conditional distribution of the next state given (s_t, a_t) alone is non-Markovian. We restore the Markov property by augmenting the state with the memory.

Theorem 2 (Markovian lifting)**.** *Define the augmented state $x_t := (s_t, M_t)$ and treat the retrieved case c_t as the action. Under deterministic memory rewriting, the process $\{(x_t, c_t)\}$ is a Markov decision process (the* Reflected MDP*) with transition kernel*

$$P_{\mathrm{LLM}}(x_{t+1} \mid x_t, c_t) = \mathop{\mathbb{E}}_{a \sim p_{\mathrm{LLM}}(\cdot \mid s_t, c_t)} \big[\mathcal{P}(s_{t+1} \mid s_t, a)\, \mathbb{1}\{M_{t+1} = M_t \cup (s_t, a, r_t)\} \big]$$

and reward $R_{\mathrm{LLM}}(x_t, c_t) = \mathbb{E}_{a \sim p_{\mathrm{LLM}}}[\mathcal{R}(s_t, a)]$.

The proof (Appendix D) is by direct verification of the Markov property in the augmented space and shows that the LLM kernel is absorbed into the effective transition. Theorem 2 is the bridge between practical memory-augmented agents and classical RL: any algorithmic guarantee for the Reflected MDP transfers back to the original SRDP via Eq. (1).

3 READ–WRITE REFLECTIVE LEARNING

We design a learning algorithm for the retrieval policy μ that (i) is stable for growing memory, (ii) preserves the LLM's exploratory capacity, and (iii) admits convergence and optimality guarantees.

Parzen-window retrieval prior. Memory consists of natural-language case descriptions, so structural assumptions are weak. We adopt a density-aware prior based on Parzen-window estimation (Parzen, 1962; Silverman, 1986): given a kernel k_θ over an embedded state representation, the prior probability of retrieving case $c = (s_c, a_c, r_c)$ at state s is

$$\mu_0(c \mid s, M) \;\propto\; k_\theta\big(\mathrm{enc}(s),\, \mathrm{enc}(s_c)\big), \tag{2}$$

augmented with a small probability mass on a special *void case* c_0 that triggers exploration directly from the LLM's prior. The void case preserves discovery in regions where memory coverage is sparse and is the mechanism behind the asymptotic-optimality result below.

Soft policy iteration over the Reflected MDP. We optimise μ in the maximum-entropy / KL-regularised regime (Haarnoja et al., 2017; 2018; Neu et al., 2017). Define the soft state–action value

$$Q^\pi(x,c) = R_{\mathrm{LLM}}(x,c) + \gamma \mathop{\mathbb{E}}_{x' \sim P_{\mathrm{LLM}}}\Big[V^\pi(x')\Big], \;\; V^\pi(x) = \sum_c \mu(c \mid x)\big[Q^\pi(x,c) - \alpha \log \tfrac{\mu(c|x)}{\mu_0(c|x)}\big], \tag{3}$$

where $\alpha > 0$ controls the strength of the KL regulariser against the Parzen prior μ_0. The Read step performs KL-regularised greedy improvement, whose closed-form solution is

$$\mu'(c \mid x) \;=\; \frac{\mu_0(c \mid x)\, \exp(Q^\pi(x,c)/\alpha)}{Z(x)}, \qquad Z(x) = \sum_{c'} \mu_0(c' \mid x) \exp(Q^\pi(x,c')/\alpha). \tag{4}$$

The Write step simultaneously appends the new transition to M and re-fits Q via the soft Bellman recursion (full derivation and per-episode pseudocode in Appendix E, Algorithm 2).

Convergence under fixed and slowly evolving memory. Under bounded rewards and a fixed memory, alternating the soft policy improvement (4) with the soft Bellman evaluation (3) converges geometrically to the unique KL-regularised fixed point (Theorem 5, Appendix F). When memory evolves on a slower time scale than the (Q, μ) updates, the iterates produced by Algorithm 2 track the locally optimal $(Q^\star_{M_t}, \mu^\star_{M_t})$ and converge almost surely (Theorem 7, Appendix G). The headline result, however, is asymptotic optimality of the composite retrieval–LLM policy as memory coverage densifies.

Theorem 3 (Asymptotic optimality)**.** *Let $\delta_M = \sup_{s \in \mathcal{S}_{\mathrm{reach}}} \min_{c \in M}\, d(\mathrm{enc}(s), \mathrm{enc}(s_c))$ denote the* coverage radius *of memory M over the reachable state space, and let $\varepsilon_{\mathrm{LLM}}(\delta)$ be the local LLM-imitation modulus from Assumption 10 (Appendix H). Under bounded rewards and Assumption 10, the composite policy $\pi^\star_M(a \mid s) = \sum_c \mu^\star_M(c \mid s, M)\, p_{\mathrm{LLM}}(a \mid s, c)$ satisfies*

$$\big\| V^{\pi^\star_M} - V^\star \big\|_\infty \;\le\; \frac{2\, R_{\max}}{(1-\gamma)^2}\, \varepsilon_{\mathrm{LLM}}(\delta_M), \tag{5}$$

so that as memory coverage densifies ($\delta_M \to 0$), the composite policy converges to the optimal policy of the underlying environment MDP.

Interpretation. Theorem 3 is the formal version of the empirical observation that case banks improve agent behaviour: coverage (geometric density of stored states) and LLM local fidelity (how well the LLM's action distribution generalises around a stored case) are the two quantities controlling the gap to optimal. Both are measurable in practice (§5), and both are improvable through systematic Write operations.

4 Practical Instantiation: *Memento* Deep Research Agent

Memento instantiates the SRDP as a planner–executor Deep Research agent with frozen LLMs (Figure 2). The planner retrieves past plans from persistent Case Memory; the executor carries out subtasks through tools. Subtask Memory tracks progress within a task, and Tool Memory stores intermediate calls and outputs. Binary correctness feedback is observed at task completion.

Mapping SRDP objects to *Memento*. The state s contains the query, task metadata, and available subtask outcomes. A case $c = (s_c, a_c^{\mathrm{plan}}, r_c)$ records a prior state, plan, and binary reward. The retrieval policy $\mu(c \mid s, M)$ implements Read; the frozen planner samples $a^{\mathrm{plan}} \sim p_{\mathrm{plan}}(\cdot \mid s, c)$, and the executor realises that plan through tool calls. Write stores the completed experience and, for parametric CBR, updates $Q(s, c)$.

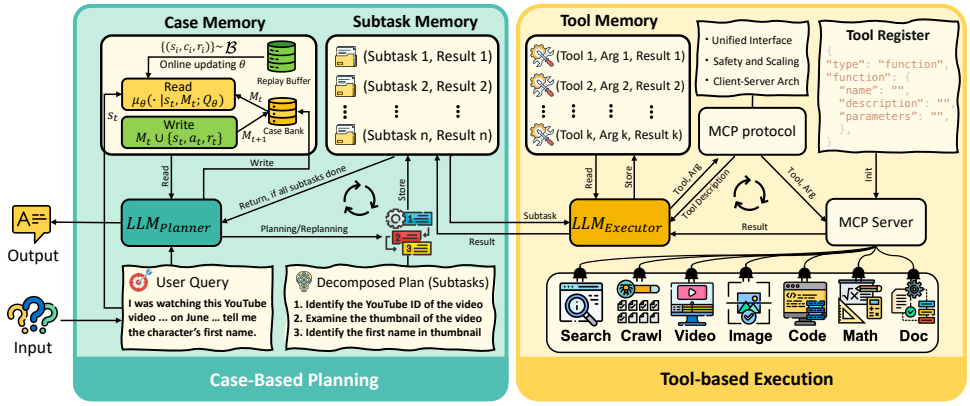


Figure 2: Architecture of *Memento* with parametric memory. The planner reads cases and writes completed experiences; the executor invokes tools through MCP (Anthropic, 2024). Case Memory persists across tasks, while Subtask and Tool Memories support execution within a task.

Planner–executor loop. Algorithm 1 makes the interaction sequence explicit. Retrieved cases condition planning; subtask results return to the planner for finalisation or replanning. At task completion, the plan and its outcome become a persistent case for future tasks. Parametric CBR additionally updates the retrieval scorer using this feedback, while the planner and executor weights remain frozen. The loop describes the full system, including optional replanning.

Algorithm 1 *Memento*: Planner–Executor Loop with Reflective Memory

Require: Task instruction s_0, planner LLM p_{plan}, executor LLM p_{exec}, tool set $\mathcal{T}$, case bank M, kernel parameters θ, budget $T_{\max}$.

Initialise subtask memory $\mathcal{U} \leftarrow \emptyset$, tool memory $\mathcal{V} \leftarrow \emptyset$, $t \leftarrow 0$.
while task not complete and $t < T_{\max}$ **do**
  [Read] retrieve top-K cases via Eq. (2)+(4) on memory M.
  [Plan] sample plan $a_t^{\text{plan}} \sim p_{\text{plan}}(\cdot \mid s_t, \{c_i\}_{i=1}^{K}, \mathcal{U})$.
  Decompose a_t^{plan} into subtasks $\{u_j\}$; push to $\mathcal{U}$.
  for each subtask u_j **do**
    [Execute] loop p_{exec} over MCP tools $\mathcal{T}$ with tool memory $\mathcal{V}_j$ until done.
    Append result and tool calls to $\mathcal{V}_j$ and $\mathcal{U}$.
  end for
  [Replan or finalise] if subtask outcomes fail to satisfy s_t, replan; else assemble final answer $\hat{y}_t$.
  Observe terminal reward r_t (binary correctness) at task completion.
end while
[Write] append $(s_0, a^{\text{plan}}, r)$ to M; if parametric, update θ via Eq. (6).

One-step learning instantiation. In the single-decision setting used to train the planner's retrieval scorer, terminal feedback provides the learning target directly: $V^{\pi}(x') = 0$ at the absorbing state, so Eq. (3) reduces to a Monte-Carlo update. For binary task reward r, we fit a two-layer MLP Q over frozen SimCSE encodings (Gao et al., 2021) using

$$\mathcal{L}_{\text{CE}}(\theta) = \mathop{\mathbb{E}}_{(s,c,r)} \Big[-r \log Q(s, c; \theta) - (1 - r) \log\big(1 - Q(s, c; \theta)\big) \Big]. \tag{6}$$

This supervised instantiation uses approximately 3k cases and retains the KL-regularised retrieval structure of §3; the general multi-step update is in Algorithm 2. For the non-parametric variant, Read selects top-K cases by SimCSE similarity under the Gaussian-kernel prior in Eq. (2); Write appends outcomes without training Q. This recovers Parzen-only retrieval as $\alpha \to \infty$.

Tools and memory curation. MCP exposes search and crawling, heterogeneous-document processing, and sandboxed code execution (Appendix J). We retain only the final-step (s, a, r) from each trajectory to limit redundant cases. Case Memory persists across tasks; Subtask and Tool Memories are cleared between tasks (Appendix I).

Table 1: Text-only main results on the DeepResearcher suite (F1; Qwen2.5-7B family). Baseline scores are from Zheng et al. (2025a). Avg. is the macro-average across all the datasets; *Memento* is reported with three-run mean ± s.d.

Method	In-Domain				Out-of-Domain			Avg.
	NQ	TQ	HotpotQA	2Wiki	MusiQue	Bamboogle	PopQA	
Prompt-based baselines								
CoT	19.8	45.6	24.4	26.4	8.5	22.1	17.0	23.4
CoT + RAG	**42.0**	68.9	37.1	24.4	10.0	25.4	46.9	36.4
Search-o1	32.4	58.9	33.0	30.9	14.7	46.6	38.3	36.4
RL-trained baselines								
Search-R1	33.1	44.7	45.7	43.4	26.5	45.0	43.0	40.2
R1-Searcher	35.4	73.1	44.8	59.4	22.8	64.8	42.7	49.0
DeepResearcher	39.6	**78.4**	**52.8**	**59.7**	27.1	71.0	48.5	**53.9**
Ours (No LLM fine-tuning)								
Memento	39.2±0.7	74.3±0.8	52.3±0.9	59.2±1.0	**29.6±1.1**	**71.6±1.8**	**48.7±1.0**	53.6±0.6

Table 2: Multimodal main results (accuracy, %; Qwen2.5-VL-7B). *Memento* is reported with three-run mean ± s.d. Bold marks column maxima, including ties.

Method	In-Domain	Out-of-Domain				Avg.
	FVQA-test	InfoSeek	SimpleVQA	LiveVQA	MMSearch	
Search Agent						
ReAct (Yao et al., 2023)	34.2	28.3	35.8	10.7	21.1	26.0
MMSearch-R1 (Wu et al., 2025)	58.0	49.0	55.3	28.3	43.9	46.9
WebWatcher (Geng et al., 2025)	–	–	54.3	–	55.3	–
DeepEyes2 (Hong et al., 2026)	60.6	51.1	59.4	–	**63.7**	–
Memory-Based Search Agent						
RAG (Lewis et al., 2020)	60.5	55.9	60.5	31.7	54.4	52.6
Mem0 (Chhikara et al., 2025)	55.6	48.2	56.7	24.5	43.3	45.7
A-MEM (Xu et al., 2025)	38.5	36.0	51.6	22.6	40.9	37.9
ReasoningBank (Ouyang et al., 2025)	64.7	**59.5**	60.4	34.2	57.3	55.2
ExpeL (Zhao et al., 2024)	64.2	58.6	**62.5**	34.1	61.4	56.2
Ours (No LLM fine-tuning)						
Memento	**67.4±0.5**	57.7±0.6	**62.5±0.5**	**37.5±0.5**	61.9±0.9	**57.4±0.2**

5 EXPERIMENTS

We evaluate complete-system performance and the contribution of Case Memory under controlled configurations. The main benchmarks are the seven-dataset DeepResearcher suite (Zheng et al., 2025a) and the five-dataset multimodal suite of Wu et al. (2025). Additional SimpleQA (Wei et al., 2024), HLE (Phan et al., 2025), and UIS-QA (Liu et al., 2026) results test broader transfer, while GAIA (Mialon et al., 2023) results evaluate end-to-end system competitiveness. We report each benchmark's standard metric: Exact Match (GAIA), F1 / Partial Match (DeepResearcher, HLE), and accuracy (SimpleQA, UIS-QA).

The controlled DeepResearcher experiments use a Qwen2.5-7B planner and executor under the Local RAG setting (Zheng et al., 2025a). The multimodal experiments use Qwen2.5-VL-7B with a shared executor and tool environment. For DeepResearcher, the bank is populated from NQ, TQ, HotpotQA, and 2Wiki; MusiQue, Bamboogle, and PopQA are out-of-distribution (OOD). For the multimodal suite, the bank uses FVQA-train; FVQA-test is in-domain, and InfoSeek, SimpleVQA, LiveVQA, and MMSearch are OOD. Within each benchmark, all controlled configurations use identical models, tools, budgets, and inference settings to ensure fair comparisons.

Table 3: Ablation studies on DeepResearcher benchmarks (F1). All stages use Qwen2.5-7B; tool-enabled stages share the Local RAG setting. We focus on five components: the base LLM (L), tool use (T), planner-executor architecture (P), similarity-based case memory (M), and the learned retrieval policy (Q). The last three rows are reported with mean ± s.d. over three runs.

L	T	P	M	Q	Ablation Variants	In-Domain				Out-of-Domain			Avg.
						NQ	TQ	HotpotQA	2Wiki	MusiQue	Bamboogle	PopQA	
✓	×	×	×	×	Direct Answer	19.8	45.6	24.4	26.4	8.5	22.1	17.0	23.4
✓	✓	×	×	×	Tool-Based Agent	34.5	52.6	31.6	28.6	16.8	35.8	36.9	33.8
✓	✓	✓	×	×	Planner-Executor Agent	36.9±0.7	71.8±0.8	48.6±1.2	55.2±1.0	26.1±1.3	68.2±2.3	45.9±0.8	50.4±0.6
✓	✓	✓	✓	×	Non-Parametric CBR	38.4±0.8	73.6±0.7	50.8±1.0	57.6±1.1	28.3±1.2	70.5±2.0	47.8±0.9	52.4±0.6
✓	✓	✓	✓	✓	**Full Memento**	**39.2±0.7**	**74.3±0.8**	**52.3±0.9**	**59.2±1.0**	**29.6±1.1**	**71.6±1.8**	**48.7±1.0**	**53.6±0.6**

Table 4: Ablation studies on multi-modal benchmarks (accuracy, %; Qwen2.5-VL-7B). Component switches and reporting conventions follow Table 3.

L	T	P	M	Q	Ablation Variants	In-Domain	Out-of-Domain				Avg.
						FVQA-test	InfoSeek	SimpleVQA	LiveVQA	MMSearch	
✓	×	×	×	×	Direct Answer	20.9	23.9	30.4	8.3	7.2	18.1
✓	✓	×	×	×	Tool-Based Agent	34.2	28.3	35.8	10.7	21.1	26.0
✓	✓	✓	×	×	Planner-Executor Agent	61.8±0.5	56.9±0.7	**63.0±0.4**	33.6±1.3	54.7±0.7	54.0±0.5
✓	✓	✓	✓	×	Non-Parametric CBR	66.1±0.5	57.5±0.3	61.6±0.5	36.7±0.2	60.8±0.5	56.6±0.1
✓	✓	✓	✓	✓	**Full Memento**	**67.4±0.5**	**57.7±0.6**	62.5±0.5	**37.5±0.5**	**61.9±0.9**	**57.4±0.2**

5.1 Main results

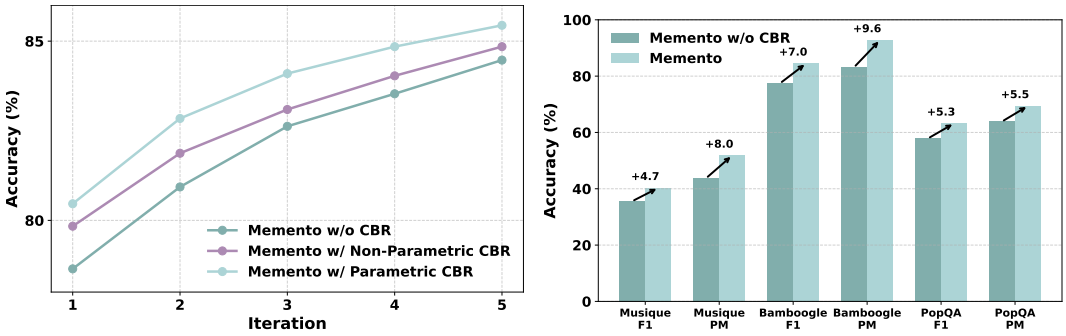
Text-Only Benchmark Results on DeepResearcher. Table 1 compares full *Memento* with prompt-based and RL-trained agents on the DeepResearcher datasets. Memento attains a seven-dataset average of 53.6 ± 0.6 F1, statistically tying the strongest RL-trained baseline DeepResearcher (53.9 F1) and clearing the prompt-based and RL-trained baselines by several F1 points. Read–write reflective memory recovers the gain RL fine-tuning provides, without ever updating LLM weights.

Multi-Modal Benchmark Results. Table 2 compares full *Memento* with memory methods sharing a Qwen2.5-VL-7B executor. We compare two families: contextual memory (RAG (Lewis et al., 2020), Mem0 (Chhikara et al., 2025), A-MEM (Xu et al., 2025)), which prepends retrieved memory directly to the Executor's context; and high-level-guidance memory (ReasoningBank (Ouyang et al., 2025), ExpeL (Zhao et al., 2024), Memento), which distils memory into a plan that conditions the Executor. *Memento* reaches 57.4 ± 0.2 average accuracy, compared with 56.2 for ExpeL and 55.2 for ReasoningBank, and leads on FVQA-test, LiveVQA, and MMSearch.

5.2 Ablation Studies

Tables 3 and 4 present two detailed component-wise ablation studies on text-only DeepResearcher and multi-modal benchmarks, respectively. Specifically, we isolate the effect of five components in *Memento*: the base LLM (L), tool use (T), planner-executor architecture (P), similarity-based case memory (M), and the learned retrieval policy (Q). Checkmarks denote enabled components and crosses denote absent ones. The last three rows hold the planner–executor configuration fixed and isolate memory and retrieval. The non-parametric variant reads cases by SimCSE similarity and writes outcomes without training a scorer. Parametric retrieval starts from the same bank and prior. We report three-run mean ± standard deviation.

On DeepResearcher, Non-Parametric CBR (i.e., similarity-based case memory) adds $+2.0$ F1 over the fixed Planner-Executor Agent, while *Memento* adds another $+1.2$ by training the retrieval policy. The corresponding multimodal gains are $+2.6$ and $+0.8$ accuracy points. Notably, *Memento* consistently



(a) Five-pass continual learning (PM).

(b) OOD transfer (F1 and PM).

Figure 3: Empirical validation of theory. (a) Five-iteration learning curve on DeepResearcher: parametric CBR (top), non-parametric CBR (middle), no-CBR (bottom). (b) Per-dataset OOD gains over the no-CBR baseline.

Table 5: GAIA validation leaderboard (reported scores, %). Systems differ in backbone, tools, and inference budget; this table measures end-to-end competitiveness.

System	Avg.	Level 1	Level 2	Level 3
Memento (Pass@3)	**87.88**	**96.23**	**90.70**	61.54
Alita	87.27	88.68	89.53	**76.92**
Skywork Super Agents v1.1	82.42	92.45	83.72	57.69
Langfun Agent (Gemini 2.5)	79.39	88.68	80.23	57.69
OpenAI DeepResearch (o3)	67.40	74.30	69.10	47.60
Open Deep Research (o1)	55.15	67.92	53.49	34.62

outperforms Non-Parametric CBR across all 12 datasets, despite the latter's strong baseline retriever, highlighting the importance of learning a retrieval policy that captures decision-relevant similarity. On SimpleVQA, even learned retrieval is 0.5 points below no memory. One possible explanation is that, for simpler single-hop questions, retrieved memories introduce distracting context that interferes with otherwise straightforward reasoning.

5.3 Empirical Validation of the Theory

Continual learning (Theorem 7). We treat each DeepResearcher pass as one slow-time-scale memory update, with the fast (Q, μ) dynamics converging within each pass. Figure 3a tracks five passes (PM, %). All three variants improve monotonically; both CBR variants stay above the no-CBR baseline at every pass; the parametric variant is consistently strongest, in line with the prediction that learnable retrieval tracks the optimal $\mu^\star$ more tightly than a fixed similarity prior. The fifth-pass gain is small ($+0.59$ PM for parametric CBR), indicating saturation.

Out-of-distribution generalisation (Theorem 3). Following the OOD protocol of Zheng et al. (2025a), we populate the case bank on NQ/TQ/HotpotQA/2Wiki and evaluate on MusiQue/Bamboogle/PopQA. Figure 3b shows that case-based memory yields $+4.7$ to $+9.6$ absolute F1 points on every OOD dataset over the no-CBR ablation. The magnitude scales with the test-time coverage radius: Bamboogle, the most distributionally similar to training, gains the most ($+9.6$ F1), directly visualising the δ_M dependence in Theorem 3.

5.4 End-To-End Results

In this section, we present additional end-to-end results on GAIA, SimpleQA, HLE, and UIS-QA, emphasising system-level comparisons rather than fully controlled evaluations.

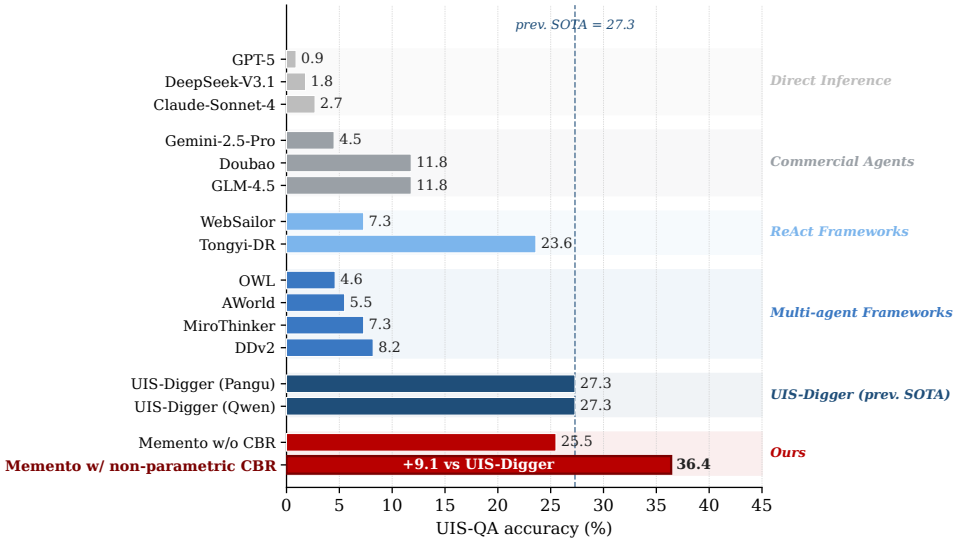


Figure 4: UIS-QA accuracy (%) across system families.

GAIA. Table 5 reports the GAIA validation leaderboard. *Memento* attains the top average score (87.88% Pass@3), ahead of Alita, Skywork Super Agents, and OpenAI's DeepResearch, and is the column-best on Levels 1 and 2; on Level 3 it remains competitive with the strongest reported entry.

SimpleQA and HLE. On SimpleQA, *Memento* reaches 95.0% accuracy, exceeding WebSailor (Li et al., 2025a) (93.5%) and prior web-agent baselines. On HLE, a 2,500-question exam covering eight academic domains, *Memento* obtains 24.4% PM, ranking second overall, within 0.9 points of GPT-5 (25.32%) and ahead of Gemini-2.5-Pro (21.64%), o3-high (20.32%), and o4-mini-high (18.08%). Per-category breakdowns and a visual summary are in Appendix L.

UIS-QA. On UIS-QA (Liu et al., 2026), a 110-question benchmark for unindexed information seeking (tasks whose answers live in web content unreachable by standard search-engine crawlers, including embedded files, dynamic pages, and deep-link subdirectories), even agents backed by o3 or GPT-5 score below 30%. Without its case bank, *Memento* reaches only 25.5%, since UIS is a novel task family that the original case bank was never populated for. Adding non-parametric CBR raises *Memento* to 36.36%, a +10.86-point gain over the no-CBR variant and +9.06 over the previously best result (UIS-Digger, 27.3%). Even sparse coverage of a new task distribution translates into measurable improvement, exactly as Theorem 3 predicts. A visual summary (Figure 4), action-space coverage, and backbone details for every system are in Appendix N.

6 Conclusion

We presented a method for continual learning of LLM agents that decouples agent adaptation from LLM parameter updates. Formalising reflection as a Stateful Reflective Decision Process and lifting it to a Reflected MDP recasts retrieval as a learnable action inside a Markovian process, on which we prove convergence under bounded rewards and a fixed memory, two-time-scale convergence when the memory evolves slowly, and asymptotic optimality as the case bank densely covers the state space. In its single-step deployment, the method reduces to a stable supervised binary classifier of retrieval utility, training with only $\sim$3k operational cases. Empirically, *Memento* statistically ties the strongest RL-fine-tuned baseline without any LLM weight update in DeepResearcher benchmarks, and attains the best average accuracy on five multimodal benchmarks.

AI USE STATEMENT

Generative AI tools assisted with language editing, LaTeX formatting, and template compliance. In the reported experiments, LLMs were used as components of the evaluated agents. The authors are responsible for the research design, experimental results, analysis, and final manuscript.

ETHICS STATEMENT

This work studies memory-based LLM agents using existing benchmarks. Persistent memory and external tool use may introduce privacy risks, retain biased information, or propagate errors. Deployment should therefore include appropriate data access controls and human oversight.

REPRODUCIBILITY STATEMENT

The main text and appendix describe the proposed algorithms, experimental settings, and theoretical proofs. We provide anonymized source code in the supplementary materials to support reproducibility and report mean and standard deviation over three runs for the controlled memory ablations.

REFERENCES

Agnar Aamodt and Enric Plaza. Case-based reasoning: Foundational issues, methodological variations, and system approaches. *AI Communications*, 7(1):39–59, 1994.

Abbas Abdolmaleki, Jost Tobias Springenberg, Yuval Tassa, Rémi Munos, Nicolas Heess, and Martin Riedmiller. Maximum a posteriori policy optimisation. *arXiv preprint arXiv:1806.06920*, 2018.

John R. Anderson, Michael Matessa, and Christian Lebiere. ACT-R: A theory of higher level cognition and its relation to visual attention. *Human–Computer Interaction*, 12(4):439–462, 1997.

Anthropic. Introducing the model context protocol. `https://www.anthropic.com/news/model-context-protocol`, 2024.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-VL technical report, 2025.

Dimitri P. Bertsekas and John N. Tsitsiklis. *Neuro-Dynamic Programming*. Athena Scientific, Belmont, MA, 1996.

Charles Blundell, Benigno Uria, Alexander Pritzel, Yazhe Li, Avraham Ruderman, Joel Z. Leibo, Jack W. Rae, Daan Wierstra, and Demis Hassabis. Model-free episodic control. *arXiv preprint arXiv:1606.04460*, 2016.

Vivek S. Borkar. *Stochastic Approximation: A Dynamical Systems Viewpoint*. Cambridge University Press, 2009.

Vivek S. Borkar and Sean P. Meyn. The ODE method for convergence of stochastic approximation and reinforcement learning. *SIAM Journal on Control and Optimization*, 38(2):447–469, 2000.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

ByteDance. DeerFlow: Deep exploration and efficient research flow. `https://deerflow.tech/`, 2025.

Camel-AI. OWL: Optimized workforce learning for general multi-agent assistance in real-world task automation. `https://github.com/camel-ai/owl`, 2025.

Yang Chen, Hexiang Hu, Yi Luan, Haitian Sun, Soravit Changpinyo, Alan Ritter, and Ming-Wei Chang. Can pre-trained vision and language models answer visual information-seeking questions? In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 14948–14968, 2023.

Xianfu Cheng, Wei Zhang, Shiwei Zhang, Jian Yang, Xiangyuan Guan, Xianjie Wu, Xiang Li, Ge Zhang, Jiaheng Liu, Yuying Mai, et al. SimpleVQA: Multimodal factuality evaluation for multimodal large language models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 4637–4646, 2025.

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.

Filippos Christianos, Georgios Papoudakis, Matthieu Zimmer, Thomas Coste, Zhihao Wu, Jingxuan Chen, Khyati Khandelwal, James Doran, Xidong Feng, Jiacheng Liu, Kun Shao, Jun Wang, et al. Pangu-agent: A fine-tunable generalist agent with structured reasoning. *arXiv preprint arXiv:2312.14878*, 2023.

Runnan Fang, Yuan Liang, Xiaobin Wang, Jialong Wu, Shuofei Qiao, Pengjun Xie, Fei Huang, Huajun Chen, and Ningyu Zhang. Memp: Exploring agent procedural memory. *arXiv preprint arXiv:2508.06433*, 2025.

Mingyang Fu, Yuyang Peng, Benlin Liu, Yao Wan, and Dongping Chen. LiveVQA: Live visual knowledge seeking. *arXiv preprint arXiv:2504.05288*, 2025.

Yao Fu, Dong-Ki Kim, Jaekyeom Kim, Sungryull Sohn, Lajanugen Logeswaran, Kyunghoon Bae, and Honglak Lee. Autoguide: Automated generation and selection of context-aware guidelines for large language model agents. *Advances in Neural Information Processing Systems*, 37:119919–119948, 2024.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. SimCSE: Simple contrastive learning of sentence embeddings. *arXiv preprint arXiv:2104.08821*, 2021.

Xinyu Geng, Peng Xia, Zhen Zhang, Xinyu Wang, Qiuchen Wang, Ruixue Ding, Chenxi Wang, Jialong Wu, Yida Zhao, Kuan Li, Yong Jiang, Pengjun Xie, Fei Huang, and Jingren Zhou. WebWatcher: Breaking new frontier of vision-language deep research agent, 2025.

Samuel J. Gershman and Nathaniel D. Daw. Reinforcement learning and episodic memory in humans and animals: An integrative framework. *Annual Review of Psychology*, 68:101–128, 2017.

Siyuan Guo, Cheng Deng, Ying Wen, Hechang Chen, Yi Chang, and Jun Wang. DS-Agent: Automated data science by empowering large language models with case-based reasoning. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, pp. 16813–16848, 2024.

Siyuan Guo, Huiwu Liu, Xiaolong Chen, Yuming Xie, Liang Zhang, Tao Han, Hechang Chen, Yi Chang, and Jun Wang. Optimizing case-based reasoning system for functional test script generation with large language models. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 4487–4498, 2025.

Siyuan Guo, Yali Du, Hechang Chen, Yi Chang, and Jun Wang. Cascade: Case-based continual adaptation for large language models during deployment. *arXiv preprint arXiv:2605.06702*, 2026.

Tuomas Haarnoja, Haoran Tang, Pieter Abbeel, and Sergey Levine. Reinforcement learning with deep energy-based policies. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, pp. 1352–1361. PMLR, 2017.

Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, and Sergey Levine. Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905*, 2018.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. Constructing a multi-hop qa dataset for comprehensive evaluation of reasoning steps. In *Proceedings of the 28th International Conference on Computational Linguistics*, pp. 6609–6625, 2020.

Jack Hong, Chenxiao Zhao, ChengLin Zhu, Weiheng Lu, and Guohai Xu. Deepeyesv2: Toward agentic multimodal model. In *International Conference on Learning Representations*, volume 2026, pp. 114851–114872, 2026.

Wenyue Hua, Sripad Karne, Qian Xie, Armaan Agrawal, Nikos Pagonas, Kostis Kaffes, and Tianyi Peng. Agentopt v0. 1 technical report: Client-side optimization for llm-based agent. *arXiv preprint arXiv:2604.06296*, 2026.

Yuxuan Huang, Yihang Chen, Haozheng Zhang, Kang Li, Huichi Zhou, Meng Fang, Linyi Yang, Xiaoguang Li, Lifeng Shang, Songcen Xu, et al. Deep research agents: A systematic examination and roadmap. *arXiv preprint arXiv:2506.18096*, 2025.

Alibaba InclusionAI. AWorld: A unified agent playground for computer and phone use tasks. `https://github.com/inclusionAI/AWorld`, 2025.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Arik, Dong Wang, Hamed Zamani, and Jiawei Han. Search-R1: Training LLMs to reason and leverage search engines with reinforcement learning. *arXiv preprint arXiv:2503.09516*, 2025.

Sham Kakade and John Langford. Approximately optimal approximate reinforcement learning. In *Proceedings of the 19th International Conference on Machine Learning (ICML)*, pp. 267–274, 2002.

Vijay R. Konda and John N. Tsitsiklis. Convergence rate of linear two-time-scale stochastic approximation. *The Annals of Applied Probability*, 14(2):796–819, 2004.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33: 9459–9474, 2020.

Kuan Li, Zhongwang Zhang, Huifeng Yin, Liwen Zhang, Litu Ou, Jialong Wu, Wenbiao Yin, Baixuan Li, Zhengwei Tao, Xinyu Wang, et al. Websailor: Navigating super-human reasoning for web agent. *arXiv preprint arXiv:2507.02592*, 2025a.

Xiaoxi Li, Guanting Dong, Jiajie Jin, Yuyao Zhang, Yujia Zhou, Yutao Zhu, Peitian Zhang, and Zhicheng Dou. Search-o1: Agentic search-enhanced large reasoning models. *arXiv preprint arXiv:2501.05366*, 2025b.

Xinbin Liang, Jinyu Xiang, Zhaoyang Yu, Jiayi Zhang, Sirui Hong, Sheng Fan, and Xiao Tang. OpenManus: An open-source framework for building general AI agents, 2025.

Chang Liu, Chuqiao Kuang, Tianyi Zhuang, Yuxin Cheng, Huichi Zhou, Xiaoguang Li, and Lifeng Shang. Uis-digger: Towards comprehensive research agent systems for real-world unindexed information seeking. *arXiv preprint arXiv:2603.08117*, 2026.

Grégoire Mialon, Clémentine Fourrier, Thomas Wolf, Yann LeCun, and Thomas Scialom. Gaia: a benchmark for general ai assistants. In *The Twelfth International Conference on Learning Representations*, 2023.

Zhanfeng Mo, Xingxuan Li, Yuntao Chen, and Lidong Bing. Multi-agent tool-integrated policy optimization, 2025.

Rémi Munos. Performance bounds in L_p-norm for approximate value iteration. *SIAM Journal on Control and Optimization*, 46(2):541–561, 2007.

Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, et al. WebGPT: Browser-assisted question-answering with human feedback. *arXiv preprint arXiv:2112.09332*, 2021.

Kartik Narayan, Yang Xu, Tian Cao, Kavya Nerella, Vishal M. Patel, Navid Shiee, Peter Grasch, Chao Jia, Yinfei Yang, and Zhe Gan. DeepMMSearch-R1: Empowering multimodal LLMs in multimodal web search, 2025.

Gergely Neu, Anders Jonsson, and Vicenç Gómez. A unified view of entropy-regularized Markov decision processes. *arXiv preprint arXiv:1705.07798*, 2017.

OpenAI. Introducing deep research. `https://openai.com/index/introducing-deep-research/`, 2025.

Siru Ouyang, Jun Yan, I Hsu, Yanfei Chen, Ke Jiang, Zifeng Wang, Rujun Han, Long T Le, Samira Daruki, Xiangru Tang, et al. Reasoningbank: Scaling agent self-evolving with reasoning memory. *arXiv preprint arXiv:2509.25140*, 2025.

Emanuel Parzen. On estimation of a probability density function and mode. *The Annals of Mathematical Statistics*, 33(3):1065–1076, 1962.

Long Phan, Alice Gatti, Ziwen Han, Nathaniel Li, Josephina Hu, Hugh Zhang, Chen Bo Calvin Zhang, Mohamed Shaaban, John Ling, Sean Shi, et al. Humanity's last exam. *arXiv preprint arXiv:2501.14249*, 2025.

Alexander Pritzel, Benigno Uria, Sriram Srinivasan, Adrià Puigdomènech Badia, Oriol Vinyals, Demis Hassabis, Daan Wierstra, and Charles Blundell. Neural episodic control. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, pp. 2827–2836, 2017.

Jiahao Qiu, Xuan Qi, Tongcheng Zhang, Xinzhe Juan, Jiacheng Guo, Yifu Lu, Yimin Wang, Zixin Yao, Qihan Ren, Xun Jiang, et al. Alita: Generalist agent enabling scalable agentic reasoning with minimal predefinition and maximal self-evolution. *arXiv preprint arXiv:2505.20286*, 2025.

Bruno Scherrer, Mohammad Ghavamzadeh, Victor Gabillon, Boris Lesner, and Matthieu Geist. Approximate modified policy iteration and its application to the game of Tetris. *Journal of Machine Learning Research*, 16(49):1629–1676, 2015.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *Proceedings of the 32nd International Conference on Machine Learning (ICML)*, pp. 1889–1897, 2015.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Bernard W. Silverman. *Density Estimation for Statistics and Data Analysis*. Chapman and Hall, 1986.

Huatong Song, Jinhao Jiang, Yingqian Min, Jie Chen, Zhipeng Chen, Wayne Xin Zhao, Lei Fang, and Ji-Rong Wen. R1-Searcher: Incentivizing the search capability in LLMs via reinforcement learning. *arXiv preprint arXiv:2503.05592*, 2025.

Larry R. Squire, Lisa Genzel, John T. Wixted, and Richard G. Morris. Memory consolidation. *Cold Spring Harbor Perspectives in Biology*, 7(8):a021766, 2015.

Xiangru Tang, Tianrui Qin, Tianhao Peng, Ziyang Zhou, Daniel Shao, Tingting Du, Xinming Wei, Peng Xia, Fang Wu, He Zhu, et al. Agent KB: Leveraging cross-domain experience for agentic problem solving. *arXiv preprint arXiv:2507.06229*, 2025.

Endel Tulving. *Elements of Episodic Memory*. Oxford University Press, 1983.

Peng Wang, Qi Wu, Chunhua Shen, Anthony Dick, and Anton van den Hengel. FVQA: Fact-based visual question answering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40 (10):2413–2427, 2018.

Zora Zhiruo Wang, Jiayuan Mao, Daniel Fried, and Graham Neubig. Agent workflow memory. *arXiv preprint arXiv:2409.07429*, 2024.

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.

Jason Wei, Nguyen Karina, Hyung Won Chung, Yunxin Joy Jiao, Spencer Papay, Amelia Glaese, John Schulman, and William Fedus. Measuring short-form factuality in large language models. *arXiv preprint arXiv:2411.04368*, 2024.

Jinming Wu, Zihao Deng, Wei Li, Yiding Liu, Bo You, Bo Li, Zejun Ma, and Ziwei Liu. MMSearch-R1: Incentivizing LMMs to search. *arXiv preprint arXiv:2506.20670*, 2025.

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, Ahmed Hassan Awadallah, Ryen W. White, Doug Burger, and Chi Wang. AutoGen: Enabling next-gen LLM applications via multi-agent conversation. *arXiv preprint arXiv:2308.08155*, 2023.

Wujiang Xu, Kai Mei, Hang Gao, Juntao Tan, Zujie Liang, and Yongfeng Zhang. A-MEM: Agentic memory for LLM agents. *arXiv preprint arXiv:2502.12110*, 2025.

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. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2018.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*, 2023.

Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. ExpeL: LLM agents are experiential learners. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 19632–19642, 2024.

Yuxiang Zheng, Dayuan Fu, Xiangkun Hu, Xiaojie Cai, Lyumanshan Ye, Pengrui Lu, and Pengfei Liu. DeepResearcher: Scaling deep research via reinforcement learning in real-world environments. *arXiv preprint arXiv:2504.03160*, 2025a.

Ziwei Zheng, Michael Yang, Jack Hong, Chenxiao Zhao, Guohai Xu, Le Yang, Chao Shen, and Xing Yu. DeepEyes: Incentivizing "thinking with images" via reinforcement learning, 2025b.

A ADDITIONAL RETRIEVAL ANALYSIS

The main text presents the controlled component results (Tables 3 and 4) and the earlier learning and OOD curves (Figure 3). This appendix provides a qualitative retrieval trace that complements those aggregate measurements.

A.1 WHY DECISION-RELEVANT RETRIEVAL MATTERS

The theory assumes a decision-relevant distance: nearby states are those for which the same stored strategy helps the LLM choose an appropriate action. This need not agree with the lexical or semantic distance of two questions. SimCSE provides an operational prior for retrieval; the outcome-trained scorer can reweight cases according to observed utility, but does not guarantee that the underlying embedding captures the required decision geometry. Table 6 presents a qualitative example that makes this distinction concrete.

Table 6: Two retrieved cases for "What award did the actor who played Peter Pan in the movie *Hook* receive?" The textually closer case induces an incomplete one-hop plan; a different two-hop pattern succeeds. This is an illustrative trace, not an estimate of failure frequency.

Outcome	Retrieved case	Stored plan and resulting behavior
Failure	Who played Peter Pan in Spielberg's *Hook*?	Find the actor and stop; the agent returns the actor's name without identifying the award.
Success	Who played the spoilt character in *Charlie and the Chocolate Factory*?	Identify the character, then find the actor across the two adaptations; the two-hop pattern yields the correct answer.

This case suggests a mechanism for the average gain from learned retrieval: a scorer trained on state–case–outcome examples can suppress cases repeatedly associated with failure and favor cases that support the right decomposition, even when they are less similar in text. The example alone does not establish that retraining always chooses the successful case, and unsuccessful cases can still be useful as counterexamples when their outcomes are presented to the planner.

B RELATED WORK

Continual learning for LLM agents. Two camps dominate. Parametric approaches update LLM weights via SFT or RL, e.g., Toolformer (Schick et al., 2023), Search-R1 (Jin et al., 2025), R1-Searcher (Song et al., 2025), DeepResearcher (Zheng et al., 2025a), and WebSailor (Li et al., 2025a); they achieve high task fidelity but require costly trajectory rollouts, large annotated datasets, and risk catastrophic forgetting on long-horizon tasks. Non-parametric approaches freeze the LLM and attach external memory, including ReAct (Yao et al., 2023), Reflexion (Shinn et al., 2023), ExpeL (Zhao et al., 2024), AutoGuide (Fu et al., 2024), Mem0 (Chhikara et al., 2025), A-MEM (Xu et al., 2025), Agent-KB (Tang et al., 2025), Alita (Qiu et al., 2025), Agent Workflow Memory (Wang et al., 2024), and the two closest concurrent works to ours: ReasoningBank (Ouyang et al., 2025), which distils self-judged successes and failures into compact strategy items retrieved at test time by query similarity, and Memp (Fang et al., 2025), which converts trajectories into procedural scripts exposed through an explicit build/retrieve/update API. They differ from *Memento* on three axes: they store distilled summaries (strategies or scripts) rather than raw (s, a, r) cases; their retrieval is a fixed similarity rule rather than a learnable action inside a Markovian process; and neither establishes formal guarantees. Across the entire non-parametric line, memory is treated as an engineering artefact rather than a control object. No prior system formulates retrieval as an action in a Markov decision process, and consequently none can invoke the standard analytical machinery of RL to obtain improvement or optimality guarantees. Our SRDP construction (§2–3) supplies precisely this missing analytic interface, and the head-to-head in §5.1 and Appendix O (Tables 2 and 13) tests *Memento* against ReasoningBank, ExpeL, A-MEM, Mem0, and RAG under a shared Qwen2.5-VL-7B Executor.

Analytical foundations from classical RL. Our convergence and optimality results assemble three well-established analytical frameworks from classical RL and stochastic approximation, and

our contribution is to identify the right object on which to apply them rather than to refine the tools themselves. *(i) Soft / KL-regularised policy iteration*, developed in the maximum-entropy line (Haarnoja et al., 2017; 2018; Neu et al., 2017) and connected to trust-region and EM-style updates by TRPO (Schulman et al., 2015) and MPO (Abdolmaleki et al., 2018), supplies the contraction-mapping argument behind Theorem 5. *(ii) Two-time-scale stochastic approximation* (Borkar, 2009; Konda & Tsitsiklis, 2004) underpins Theorem 7, where the case bank evolves on a slower scale than the (Q, μ) updates. *(iii) Bellman-operator perturbation analysis*, in the spirit of approximate dynamic programming (Bertsekas & Tsitsiklis, 1996; Munos, 2007; Scherrer et al., 2015) and the performance-difference lemma (Kakade & Langford, 2002), gives the value-function gap in Theorem 3. The closest surface-level analogues to our framework are Episodic Control (Pritzel et al., 2017; Blundell et al., 2016), which approximates value functions via nearest-neighbour lookup over an episodic buffer, and classical RAG (Lewis et al., 2020), which retrieves over a static document corpus; both share the read-then-act structure but neither casts retrieval as a stochastic action with a learnable distribution, and neither contends with the non-stationarity of a memory that grows during deployment, which is precisely why the tools above were never available to them. The non-trivial step is the Markovian lift of §2, which absorbs the frozen LLM into an effective transition kernel and turns retrieval into a learnable action; our distinctive technical move is then to apply KL regularisation to the retrieval policy itself, with a Parzen-window prior (Parzen, 1962; Silverman, 1986) that exploits the latent similarity structure of memory and admits the closed-form soft-policy update of §3. The lift is not free: each tool's regularity assumptions must be re-verified in the augmented space, in particular a Lipschitz LLM-imitation modulus that controls how the value-function gap scales with memory coverage (Appendices F–H).

Tool-augmented LLM agents and Deep Research. A rapidly growing line of work composes specialised LLM agents and external tools to solve long-horizon information-seeking and reasoning tasks beyond what a single forward pass can deliver, and our practical instantiation *Memento* (§4) is evaluated head-to-head against representatives of every branch of this literature in §5. General-purpose agent frameworks (OpenManus (Liang et al., 2025), AWorld (InclusionAI, 2025), OWL (Camel-AI, 2025), and DeerFlow (ByteDance, 2025)) orchestrate planner, executor, and verifier roles around a shared workspace, typically exposing browser, code-interpreter, file, and search tools through a uniform protocol such as MCP (Anthropic, 2024). Web-search and browser agents, from the seminal WebGPT (Nakano et al., 2021) to reasoning-augmented systems like Search-o1 (Li et al., 2025b) and WebSailor (Li et al., 2025a), couple browsing and retrieval to a frozen or finetuned LLM and report large gains on factual and multi-hop QA; their multimodal counterparts (MMSearch-R1 (Wu et al., 2025), DeepMMSearch-R1 (Narayan et al., 2025), WebWatcher (Geng et al., 2025), and DeepEyes (Zheng et al., 2025b)) extend the same loop with image-grounded queries and tool calls. End-to-end Deep Research systems, including OpenAI's DeepResearch (OpenAI, 2025) and the multi-agent tool-integrated policy MATPO (Mo et al., 2025), package these capabilities behind a single user-facing pipeline, while specialist agents such as UIS-Digger (Liu et al., 2026) target unindexed information seeking unreachable by standard search engines. We share the planner–executor architecture and MCP toolbox of these systems, with *Memento*'s executor invoking search, crawl, document parsing, code, and math tools through the same protocol layer (§4, App. J), but we differ structurally rather than tactically: prior systems either hard-code their reflective workflow (planning prompts, voting, self-criticism) or finetune the LLM end-to-end on tool trajectories, leaving the underlying decision process opaque and the resulting agent either rigid or expensive to retrain. *Memento* instead exposes the read–write reflective dynamics as an analysable Reflected MDP (§2), turning case retrieval into a learnable action with closed-form soft-policy updates (§3). To our knowledge, this is the first Deep Research agent to come with provable convergence and asymptotic-optimality guarantees, and the first whose long-run improvement is attributable to a measurable, monotone quantity (case-bank coverage) rather than to weight updates or hand-tuned prompts.

C Detailed SRDP Construction and Trajectory Factorisation

C.1 Vocabulary-Indexed State and Action Spaces

For an LLM agent operating over natural language, both states and actions are finite-length sequences over a fixed token vocabulary $\mathcal{V}$. Formally, $\mathcal{S} \subseteq \mathcal{V}^{\leq L_s}$ and $\mathcal{A} \subseteq \mathcal{V}^{\leq L_a}$, where L_s and L_a are maximum-length budgets. Each case $c = (s, a, r) \in \mathcal{S} \times \mathcal{A} \times \mathbb{R}$ is a single experience triple, and the

memory space $\mathcal{M} = \bigcup_{n\geq 0}(\mathcal{S}\times\mathcal{A}\times\mathbb{R})^n$ is the set of all finite case sequences. The state encoder $\mathrm{enc}:\mathcal{S}\to\mathbb{R}^d$ embeds states into a continuous latent space (we use SimCSE (Gao et al., 2021) with $d = 768$ throughout).

C.2 Trajectory Probability

Given the composite policy π in Eq. (1), the distribution over trajectories $\tau = (M_0, s_0, c_0, a_0, r_0, M_1, s_1, c_1, a_1, r_1, \dots)$ factorises as

$$p(\tau) \;=\; \prod_{t=0}^{T-1} \underbrace{\mu(c_t \,|\, s_t, M_t)}_{\text{(1) Retrieve}} \cdot \underbrace{p_{\mathrm{LLM}}(a_t \,|\, s_t, c_t)}_{\text{(2) Reuse \& Revise}} \cdot \underbrace{\mathbb{1}\{r_t = \mathcal{R}(s_t, a_t)\}}_{\text{(3) Evaluate}} \cdot \underbrace{\mathbb{1}\{M_{t+1} = M_t \cup (s_t, a_t, r_t)\}}_{\text{(4) Retain}} \cdot \underbrace{\mathcal{P}(s_{t+1} \,|\, s_t, a_t)}_{\text{(5) Transition}}. \tag{7}$$

Components (1), (2), (4) describe agent behaviour; (3) and (5) describe environment dynamics. In the deterministic-reward, append-only setting, the indicator factors reduce to constants, leaving the three operational distributions $(\mu, p_{\mathrm{LLM}}, \mathcal{P})$ as the only sources of randomness.

D Proof of Theorem 2 (Markovian Lifting)

Proof of Theorem 2. Conditioned on $(x_t, c_t) = ((s_t, M_t), c_t)$, the action $a_t \sim p_{\mathrm{LLM}}(\cdot \mid s_t, c_t)$, the reward $r_t = \mathcal{R}(s_t, a_t)$, the next state $s_{t+1} \sim \mathcal{P}(\cdot \mid s_t, a_t)$, and the deterministic memory rewriting $M_{t+1} = M_t \cup \{(s_t, a_t, r_t)\}$ each depend only on (x_t, c_t) and not on the prior history. Marginalising over a_t,

$$P_{\mathrm{LLM}}(x_{t+1} \mid x_t, c_t) = \mathop{\mathbb{E}}_{a\sim p_{\mathrm{LLM}}(\cdot|s_t,c_t)}\big[\mathcal{P}(s_{t+1} \mid s_t, a)\,\mathbb{1}\{M_{t+1} = M_t \cup (s_t, a, r_t)\}\big], \tag{8}$$

which depends only on (x_t, c_t). The reward of the augmented MDP is $R_{\mathrm{LLM}}(x_t, c_t) = \mathbb{E}_{a\sim p_{\mathrm{LLM}}}[\mathcal{R}(s_t, a)]$, with $|R_{\mathrm{LLM}}| \leq R_{\max}$. □

E Soft-Q Optimal Retrieval Policy and TD Loss

Closed-form retrieval policy. For fixed Q^π, the KL-regularised improvement

$$\mu'(\cdot \mid x) \;=\; \arg\max_{\mu\in\Delta(M)} \mathop{\mathbb{E}}_{c\sim\mu}\big[Q^\pi(x, c)\big] - \alpha\ \mathrm{KL}\big(\mu(\cdot \mid x)\,\|\,\mu_0(\cdot \mid x)\big) \tag{9}$$

admits the unique solution

$$\mu'(c \mid x) \;=\; \mu_0(c \mid x)\ \exp(Q^\pi(x, c)/\alpha)\ /\ Z(x), \quad Z(x) = \sum_{c'} \mu_0(c' \mid x)\ \exp(Q^\pi(x, c')/\alpha), \tag{10}$$

obtained by Lagrangian stationarity of (9) (standard; cf. (Haarnoja et al., 2017; 2018)). When μ_0 is uniform, this recovers the maximum-entropy soft-Q solution.

Soft Bellman operator and TD loss. The soft Bellman optimality operator is

$$(\mathcal{T}^{\mathrm{soft}}Q)(x, c) = R_{\mathrm{LLM}}(x, c) + \gamma \mathop{\mathbb{E}}_{x'\sim P_{\mathrm{LLM}}}\Big[\alpha \log \sum_{c'\in M'} \mu_0(c' \mid x')\, e^{Q(x',c')/\alpha}\Big], \tag{11}$$

and the squared Bellman residual we minimise is

$$\mathcal{L}_{\mathrm{TD}}(\theta) = \mathop{\mathbb{E}}_{(s,c,r,s',M,M')\sim\mathcal{B}}\big[(Q_\theta(x, c) - y)^2\big], \quad y = r + \gamma\,\alpha \log \sum_{c'\in M'} \mu_0(c' \,|\, x')\, e^{Q_{\bar\theta}(x',c')/\alpha}. \tag{12}$$

Single-step instantiation. In the planner instantiation, the Reflected-MDP horizon is $T = 1$; the TD target reduces to $y = r$ and (12) becomes $\mathbb{E}_{(s,c,r)}[(Q_\theta(s, c) - r)^2]$. With binary $r \in \{0, 1\}$, MSE has vanishing gradients near $0/1$, so we use cross-entropy (Eq. 6) as in §4.

Algorithm 2 Read–Write Reflective Learning (one episode)

Require: Frozen LLM kernel p_{LLM}, kernel parameters θ, entropy weight α, discount γ, learning rate η, target update period K, polyak weight β, replay buffer $\mathcal{B}$, episodic store $\mathcal{D}$, case bank M.

1: **for** $t = 0, 1, 2, \ldots$ **do**
2: **Read:** compute $\mu_0(\cdot \mid s_t, M_t)$ via Eq. (2); form posterior via Eq. (10); sample $c_t \sim \mu(\cdot \mid s_t, M_t)$.
3: **Reuse & Revise:** sample $a_t \sim p_{\mathrm{LLM}}(\cdot \mid s_t, c_t)$; execute, observe r_t, s_{t+1}.
4: **Retain:** $M_{t+1} = M_t \cup \{(s_t, a_t, r_t)\}$; push $(s_t, c_t, r_t, s_{t+1}, M_{t+1})$ to $\mathcal{B}$; append (s_t, c_t, Q_t) to $\mathcal{D}$.
5: **Update:** $\theta \leftarrow \theta - \eta \nabla_\theta \mathcal{L}_{\mathrm{TD}}(\theta)$ where $\mathcal{L}_{\mathrm{TD}}$ is Eq. (12).
6: Every K steps: $\bar{\theta} \leftarrow \beta\bar{\theta} + (1-\beta)\theta$.
7: **end for**

Per-episode pseudocode. Algorithm 2 summarises one interaction episode of Read–Write Reflective Learning, combining the Read step from Eq. (10) with the soft Bellman update in Eq. (12).

F Theorem 5 and Proof

Assumption 4 (Bounded rewards, stationary memory)**.** *$|\mathcal{R}(s,a)| \le R_{\max}$, $\gamma \in [0,1)$, and the memory M is held fixed during a soft-policy-iteration phase.*

Theorem 5 (Convergence of Parzen–KL soft policy iteration)**.** *Under Assumption 4, alternating the soft policy improvement* (4) *with the soft Bellman evaluation* (3) *on the Reflected MDP induces a sequence $\{(Q^{(n)}, \mu^{(n)})\}$ that converges geometrically to a fixed point $(Q^\star, \mu^\star)$, and $\mu^\star$ is the optimal KL-regularised retrieval policy for the current memory.*

Proof sketch. Soft policy iteration on a finite MDP with bounded rewards and KL regularisation is a standard contraction argument (Haarnoja et al., 2017; Neu et al., 2017); we record the four steps for completeness.

(i) Policy evaluation contraction. For fixed μ, the soft Bellman operator $\mathcal{T}_\mu^{\mathrm{soft}} Q := R_{\mathrm{LLM}} + \gamma\, \mathbb{E}_{x' \sim P_{\mathrm{LLM}}}[V^\mu(x')]$ is a γ-contraction in $\|\cdot\|_\infty$, so $Q^{(n)} \to Q^\mu$ geometrically.

(ii) Policy improvement. The closed-form solution (10) is the unique maximiser of the KL-regularised objective at fixed $Q^{(n)}$, so $V^{\mu^{(n+1)}} \ge V^{\mu^{(n)}}$ pointwise.

(iii) Monotone bounded sequence. Soft values are uniformly bounded by $R_{\max}/(1-\gamma) + \alpha \log |M|$ and form a monotone non-decreasing sequence, hence converge to a limit $V^\star$.

(iv) Fixed point. At the limit, $\mu^\star$ satisfies (10) with $Q^\star$, identifying $(Q^\star, \mu^\star)$ as the unique fixed point. Geometric rate $\|Q^{(n)} - Q^\star\|_\infty \le \gamma^n \|Q^{(0)} - Q^\star\|_\infty$ inherits from (i). □

G Theorem 7 and Proof

Assumption 6 (Two-time-scale conditions)**.** *The step sizes $(\eta_t), (\nu_t)$ satisfy $\sum_t \eta_t = \sum_t \nu_t = \infty$, $\sum_t \eta_t^2, \sum_t \nu_t^2 < \infty$, and $\nu_t/\eta_t \to 0$. The noise sequences are martingale-difference with bounded conditional variance, the iterates $\{Q_t\}, \{\mu_t\}, \{M_t\}$ remain bounded a.s., and the memory process admits a compact attractor $\mathcal{M}_\infty$.*

Theorem 7 (Two-time-scale convergence)**.** *Under Assumptions 4 and 6, the iterates (Q_t, μ_t, M_t) produced by Algorithm 2 track the locally optimal $(Q^\star_{M_t}, \mu^\star_{M_t})$ for the current memory, and $(Q_t, \mu_t, M_t) \to \{(Q^\star_M, \mu^\star_M, M) : M \in \mathcal{M}_\infty\}$ almost surely. If $M_t \to M_\infty$ a.s., then $(Q_t, \mu_t) \to (Q^\star_{M_\infty}, \mu^\star_{M_\infty})$.*

The proof follows the ODE method for two-timescale stochastic approximation (Borkar, 2009; Borkar & Meyn, 2000); we record only the steps that require new content for the memory-augmented setting.

Setup. Let $Z_t = (Q_t, \mu_t)$ denote the fast variables. The memory update is

$$M_{t+1} = \begin{cases} M_t \cup \{(s_t, a_t, r_t)\} & \text{w.p. } \nu_t, \\ M_t & \text{otherwise.} \end{cases} \tag{13}$$

The sample-based recursions are

$$Z_{t+1} = Z_t + \eta_t \big[F_{M_t}(Z_t) - Z_t + \xi_t\big], \qquad M_{t+1} = M_t + \nu_t \big[h(M_t, Z_t) + \xi_t^{(M)}\big], \tag{14}$$

where F_M combines KL-regularised soft policy evaluation and Parzen–KL improvement, h encodes the experience-replay drift (13), and $\xi_t, \xi_t^{(M)}$ are martingale-difference noises.

Operator-level memory distance. Memory enters F_M only through the induced base measure $\mu_0^M(\cdot \mid x)$. We therefore measure memory at the operator level:

$$\|M' - M\| \;:=\; \sup_x \mathrm{TV}\big(\mu_0^{M'}(\cdot \mid x),\, \mu_0^M(\cdot \mid x)\big). \tag{15}$$

Lemma 8 (Lipschitz continuity of F_M in memory)**.** *Under Assumption 4 and bounded Q-range $\Delta Q = Q_{\max} - Q_{\min}$, there exists $L_F = O(e^{\Delta Q/\alpha})$ such that $\|F_{M'}(Z) - F_M(Z)\| \le L_F \, \|M' - M\|$ for any fixed Z.*

Proof sketch. F_M depends on memory only through the log-sum-exp $G_M(x; Q) = \alpha \log \sum_{c'} \mu_0^M(c' \mid x)\, e^{Q(x,c')/\alpha}$ and the softmax $\Phi_M(c \mid x; Q) = \mu_0^M(c \mid x)\, e^{Q(x,c)/\alpha} / Z_M(x)$. Both maps are Lipschitz in the base measure under (15): a direct calculation using $\|\nabla_b(\alpha \log b^\top w)\|_\infty \le \alpha \max_i w_i / \min_i w_i$ and $w_i \in [e^{Q_{\min}/\alpha}, e^{Q_{\max}/\alpha}]$ yields the constants $\alpha e^{\Delta Q/\alpha}$ and $2e^{\Delta Q/\alpha}$ respectively. Conditional expectations preserve Lipschitz constants. □

Fast-time-scale ODE. For fixed M, the limiting ODE $\mathrm{d}Z/\mathrm{d}\tau = F_M(Z) - Z$ inherits κ-contraction from Theorem 5 with $\kappa = \gamma$, hence admits a unique globally asymptotically stable equilibrium $Z_M^\star = (Q_M^\star, \mu_M^\star)$.

Lemma 9 (Lipschitz fixed point)**.** $\|Z_{M'}^\star - Z_M^\star\| \;\le\; \frac{L_F}{1-\kappa} \|M' - M\|$.

Proof. Using $Z_M^\star = F_M(Z_M^\star)$ and the triangle inequality with the contraction property of $F_{M'}$, $\|Z_{M'}^\star - Z_M^\star\| \le \kappa \|Z_{M'}^\star - Z_M^\star\| + L_F \|M' - M\|$, which gives the claim. □

Slow-time-scale dynamics. Each practical memory update (experience replay, sliding window, importance-weighted memory) induces $\|M_{t+1} - M_t\| = O(\nu_t)$ a.s. under (15); the associated slow ODE $\mathrm{d}M/\mathrm{d}\varsigma = h(M)$ admits the compact attractor $\mathcal{M}_\infty$ of Assumption 6.

Tracking and conclusion. Defining $\epsilon_t = \|Z_t - Z_{M_t}^\star\|$ and combining contraction of F_M, Lemma 9, and martingale-noise control,

$$\epsilon_{t+1} \;\le\; (1 - \eta_t \lambda)\epsilon_t + \eta_t L \, \|M_{t+1} - M_t\| + \eta_t \|\xi_t\|, \qquad \lambda = 1 - \kappa > 0. \tag{16}$$

Since $\|M_{t+1} - M_t\| = O(\nu_t)$ and $\nu_t = o(\eta_t)$, the drift term is negligible relative to the contraction. Standard two-timescale SA results (Borkar, 2009; Borkar & Meyn, 2000) then give $\epsilon_t \to 0$ a.s., and all limit points of (Z_t, M_t) lie in $\{(Z_M^\star, M) : M \in \mathcal{M}_\infty\}$. If $M_t \to M_\infty$, $(Q_t, \mu_t) \to (Q_{M_\infty}^\star, \mu_{M_\infty}^\star)$ a.s. □

H Proof of Theorem 3 (Asymptotic Optimality)

Assumption 10 (Lipschitz LLM imitation)**.** *There exists $\varepsilon_{\mathrm{LLM}} : \mathbb{R}_{\ge 0} \to \mathbb{R}_{\ge 0}$ with $\varepsilon_{\mathrm{LLM}}(0) = 0$ and $\lim_{\delta \to 0} \varepsilon_{\mathrm{LLM}}(\delta) = 0$, such that for every $s \in \mathcal{S}_{\mathrm{reach}}$ and every case $c = (s_c, a_c, r_c)$ with $d(\mathrm{enc}(s), \mathrm{enc}(s_c)) \le \delta$,*

$$\big\| \, p_{\mathrm{LLM}}(\cdot \mid s, c) - \pi^\star(\cdot \mid s) \, \big\|_{\mathrm{TV}} \;\le\; \varepsilon_{\mathrm{LLM}}(\delta), \tag{17}$$

where $\pi^\star$ is an optimal policy of the underlying environment MDP.

The function $\varepsilon_{\mathrm{LLM}}(\delta)$ is the LLM's local optimality gap and is empirically measurable; small $\varepsilon_{\mathrm{LLM}}$ at moderate δ indicates a wide "radius of competence" under coarse retrieval.

Proof of Theorem 3. The argument has three steps.

Step 1 (Retrieval capability). Let $c_M(s) \in \arg\min_{c \in M} d(\mathrm{enc}(s), \mathrm{enc}(s_c))$ and $\pi_M(a \mid s) := p_{\mathrm{LLM}}(a \mid s, c_M(s))$ be the deterministic NN-retrieval policy. By definition of δ_M, $d(\mathrm{enc}(s), \mathrm{enc}(s_{c_M(s)})) \le \delta_M$, so Assumption 10 gives

$$\big\|\pi_M(\cdot \mid s) - \pi^\star(\cdot \mid s)\big\|_{\mathrm{TV}} \;\le\; \varepsilon_{\mathrm{LLM}}(\delta_M) \quad \forall s \in \mathcal{S}_{\mathrm{reach}}. \tag{18}$$

Step 2 (Value-function gap). For any policy π on a discounted MDP with $|\mathcal{R}| \le R_{\max}$ and $\gamma \in [0, 1)$, the standard simulation lemma gives

$$\big\|V^{\pi^\star} - V^{\pi}\big\|_\infty \;\le\; \frac{2\,R_{\max}}{(1-\gamma)^2}\,\sup_s \big\|\pi(\cdot \mid s) - \pi^\star(\cdot \mid s)\big\|_{\mathrm{TV}}. \tag{19}$$

This follows from $V^{\pi^\star} - V^{\pi} = (T^{\pi^\star} - T^{\pi})V^{\pi^\star} + (T^{\pi}V^{\pi^\star} - T^{\pi}V^{\pi})$, γ-contraction of T^{π}, and bounding $\|(T^{\pi^\star} - T^{\pi})V^{\pi^\star}\|_\infty \le \frac{2R_{\max}}{1-\gamma} \sup_s \|\pi - \pi^\star\|_{\mathrm{TV}}$ via Hölder's inequality with $\|V^{\pi^\star}\|_\infty \le R_{\max}/(1-\gamma)$ (cf. (Kakade & Langford, 2002; Scherrer et al., 2015)).

Step 3 (Transfer to $\pi^\star_M$). Combining (18) and (19) for the NN policy π_M:

$$\big\|V^{\pi^\star} - V^{\pi_M}\big\|_\infty \;\le\; \frac{2R_{\max}}{(1-\gamma)^2}\,\varepsilon_{\mathrm{LLM}}(\delta_M). \tag{20}$$

The KL-regularised optimum $\mu^\star_M$ satisfies $J_\alpha(\mu^\star_M) \ge J_\alpha(\mu^{\mathrm{NN}}_M)$ where $\mu^{\mathrm{NN}}_M(c \mid s) = \mathbb{1}\{c = c_M(s)\}$[1]; the soft-policy-improvement lemma (Haarnoja et al., 2018) then gives $V^{\pi^\star_M} \ge V^{\pi_M} - \alpha H_{\max}$. Combined with $V^{\pi^\star_M} \le V^{\pi^\star}$ (optimality of $\pi^\star$) and (20), letting $\alpha \downarrow 0$ (or absorbing $\alpha H_{\max}$ into the asymptotic statement) yields

$$\big\|V^{\pi^\star} - V^{\pi^\star_M}\big\|_\infty \;\le\; \frac{2R_{\max}}{(1-\gamma)^2}\,\varepsilon_{\mathrm{LLM}}(\delta_M). \tag{21}$$

Assumption 10 gives $\varepsilon_{\mathrm{LLM}}(\delta_M) \to 0$ as $\delta_M \to 0$, hence $V^{\pi^\star_M} \to V^\star$ uniformly. □

Remark (stochastic retrieval). If retrieval may with probability $\eta_M(s)$ select a case outside the nearest-neighbour radius, Step 1 generalises to $\|\pi_M - \pi^\star\|_{\mathrm{TV}} \le \varepsilon_{\mathrm{LLM}}(\delta_M) + \eta_M$, and the value bound becomes $\frac{2R_{\max}}{(1-\gamma)^2}(\varepsilon_{\mathrm{LLM}}(\delta_M) + \eta_M)$. The conclusion holds whenever $\eta_M \to 0$, which is satisfied by Parzen-window retrieval as the kernel bandwidth shrinks.

I Full System Architecture

I.1 Planner–Executor Loop

Algorithm 1 in §4 gives the full planner–executor loop. The general Read–Write Reflective Learning procedure remains in Algorithm 2; the memory lifetimes and tool interfaces used by the system are detailed below.

I.2 Memory Modules

Memento maintains three distinct memory stores, each with different lifetimes:

Case Memory. Long-lived, vectorised storage of $(s, a^{\mathrm{plan}}, r)$ triples used by the planner. Persists across tasks. Encoded with frozen SimCSE; queried via Eq. (2).

[1] To accommodate degenerate ties between the deterministic indicator and the Parzen prior, α can be taken arbitrarily small or μ^{NN}_M slightly smoothed; the resulting bound differs by an arbitrarily small additive entropy term that does not affect the asymptotic conclusion.

Subtask Memory. Per-task ledger of active subtasks and their execution outcomes; cleared after task completion.

Tool Memory. Per-subtask, per-tool log of intermediate calls and returns; allows the executor to reason about partial progress without re-invoking tools.

J Complete Tool Inventory and MCP Protocol Details

We expose all tools via the Model Context Protocol (Anthropic, 2024). The executor is an MCP client; tool servers are independent processes communicating over JSON-RPC. This allows hot-swapping individual tools without rebuilding the agent.

J.1 External Information Acquisition

Search. Self-hosted searxng metasearch engine aggregating Google, Bing, DuckDuckGo, Brave. Top-50 candidates re-ranked by SimCSE similarity to the query.

Crawl. Crawl4AI for full-page parse and Markdown extraction. Triggered when the executor requires deeper context than search snippets provide.

J.2 Document Processing

Per-format handlers: PDF (Chunkr AI fallback for malformed files), images (VLM captioning via GPT-4o), audio (AssemblyAI ASR), PowerPoint (slide-by-slide with embedded image descriptions), spreadsheets (row-wise textualisation), archives (auto-unpack), Word (Pandoc to Markdown), JSON/XML (structured-object parsing), video (Gemini 2.5 Pro summarisation), code/text files (direct read).

J.3 Reasoning and Computation

Code. Sandboxed Python and shell, validated against a security whitelist, with persistent workspace state.

Math. Rational and floating-point arithmetic, plus a SymPy backend for symbolic simplification.

J.4 MCP Server Configuration

The full `mcp_servers.yaml` is included with our code release. Each tool exposes a typed JSON schema, enabling the executor to compose calls without parsing free-form documentation.

K Hyper-Parameter Sweeps

Table 7: Effect of retrieval count K on DeepResearcher (F1/PM, weighted average over 7 datasets). $K = 4$ is optimal.

K	0	1	2	4	8	16	32
NQ	39.5/67.8	41.1/74.4	41.3/72.7	**41.9/73.0**	41.7/73.8	42.1/73.2	42.2/75.4
TQ	81.1/89.1	86.1/93.8	86.2/93.9	**86.3/94.1**	85.8/94.1	85.9/94.3	85.5/93.9
HotpotQA	62.0/76.0	65.4/80.7	65.7/81.3	**67.4/84.2**	66.6/82.0	65.5/82.0	66.4/83.2
2Wiki	78.3/90.0	81.3/94.9	80.9/94.1	81.0/94.7	**82.0/94.5**	81.1/93.6	81.0/94.1
MusiQue	35.6/43.8	39.8/50.2	40.1/52.1	**41.6/51.0**	41.0/52.1	39.6/51.4	40.3/50.4
Bamboogle	77.5/83.2	**85.9/91.2**	84.1/91.2	84.7/90.4	84.9/91.2	85.2/92.0	83.0/87.2
PopQA	57.9/63.9	62.6/70.1	63.4/71.3	**63.6/70.9**	62.7/69.2	64.1/70.9	63.2/69.4
Average	59.9/72.2	63.6/77.9	63.7/78.1	**64.5/78.5**	64.1/78.2	63.9/78.1	63.9/78.1

K.1 Fast vs. Slow-Thinking Planners

Table 8 compares pairing the executor with a "fast" general-purpose planner (GPT-4.1 or Qwen3-32B-Fast) against a "slow" deliberative planner (o3 or Qwen3-32B-Slow). Across both planner families, the fast planner consistently outperforms its slow counterpart, by 7.9 points (GPT-4.1 vs. o3 over the same o3 executor, 70.91% vs. 63.03%) and by 13.3 points on Qwen3-32B over the same o4-mini executor.

Trace inspection reveals two failure modes for slow planners. *(a) Short-circuit of the planner–executor split.* On hard items, the slow planner sometimes responds with a one-line "plan" that is, in effect, a direct guess at the final answer (e.g., "`Plan: this is` X`. Final answer:` X`.`"). No subtasks are issued; the executor receives an empty work-queue plus a guessed answer, and either echoes the guess or burns the rest of its budget on cosmetic verification of a wrong claim, never invoking the search and code tools that the framework is designed to delegate. The cue from the case bank is overridden by the planner's own confidence in its parametric memory. This pattern is not specific to our setup: Hua et al. (2026) sweep 81 planner–solver combinations on HotpotQA and report that Claude Opus 4.6, the strongest standalone model in their pool, is the worst planner across all 81 combinations because it "answers directly from parametric knowledge and bypasses the solver's search tools."

(b) Plan compression into a single chain of thought. On long-horizon multi-hop items, the slow planner instead emits a 20–40-sentence reasoning trace that bundles search, arithmetic, and synthesis into one block of prose rather than separating them into discrete subtasks. The executor, taking that block as its work specification, treats it as monolithic: it either runs the plan as a single long generation (no tool calls at all) or fixates on the first sub-step and never reaches the rest of the chain.

In a two-stage architecture, the planner's job is structured decomposition, not deep within-step reasoning; over-deliberation introduces redundant context and erodes the very specialisation the architecture is designed to exploit. Strong standalone reasoning is precisely the wrong objective to optimise in the planner role; what the planner needs to produce is a delegable decomposition, not a private answer.

Table 8: Fast- vs. slow-thinking planners on GAIA validation (Pass@1). Fast planners consistently outperform their slow counterparts when paired with the same executor.

Planner	Executor	Level 1	Level 2	Level 3	Average
GPT-4.1 (fast)	o3	**77.36**	**69.77**	**61.54**	**70.91**
o3 (slow)	o3	73.58	63.95	38.46	63.03
Qwen3-32B (fast)	o4-mini	62.26	56.98	26.92	53.94
Qwen3-32B (slow)	o4-mini	56.60	36.05	23.08	40.61

L Per-Category Results on HLE and SimpleQA

Table 9: HLE per-category F1/PM. *Memento* attains the best overall PM with consistent improvement across categories.

	Hum./SS	Math	Chem.	Other	Phys.	Eng.	Bio./Med.	CS/AI	Avg
Offline executor	5.2/9.6	7.1/5.8	2.3/7.9	2.9/11.4	7.6/4.6	12.8/14.0	5.2/17.7	6.5/11.9	6.4/8.7
Online executor	10.8/24.9	13.1/16.0	6.9/9.3	14.0/16.3	7.8/10.2	7.5/5.3	5.7/17.2	13.3/15.4	11.2/15.8
Memento w/o CBR	25.5/29.2	24.9/16.3	17.4/21.1	24.8/24.1	18.4/10.8	15.8/8.8	10.0/18.7	25.4/12.4	22.2/17.4
Memento (full)	**28.4/33.0**	**30.9/24.2**	**18.7/22.7**	**28.5/32.4**	**22.9/19.1**	**15.9/12.1**	**14.0/26.1**	**28.5/18.5**	**26.7/24.4**

M Multimodal / Text-Only Memory-Based Search-Agent Protocol

This appendix supplements the unified memory-based search-agent evaluation in §5.1 (Tables 2 and 13). The protocol is adapted from Wu et al. (2025): every memory-based agent shares the *same*

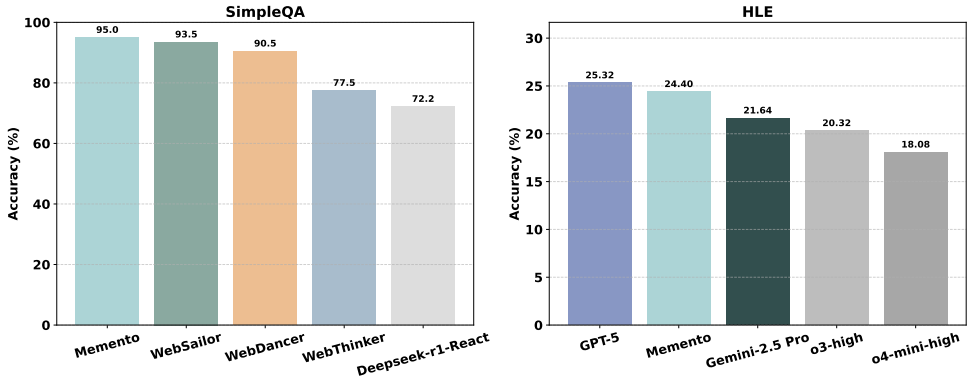


Figure 5: SimpleQA (left) and HLE (right). *Memento* sets a new state of the art on SimpleQA and ranks second on HLE, within 0.9 points of GPT-5 while outperforming all open and closed agentic baselines.

Table 10: SimpleQA per-category F1/PM. *Memento* consistently leads across all 10 categories.

	Art	Geo.	Sci./Tech	Pol.	Sport	Other	TV	Music	Hist.	Game	Avg
Offline	16.5/19.6	25.7/31.1	20.1/24.7	25.8/24.8	18.8/19.0	15.8/14.9	12.6/13.3	15.7/17.6	25.9/26.6	15.5/13.3	19.7/21.5
Online	49.7/82.9	45.1/82.5	59.4/87.7	51.6/86.9	46.6/90.1	48.7/86.4	39.5/84.3	43.1/88.5	49.7/83.8	34.7/78.6	48.5/84.8
Mementow/o CBR	83.8/92.2	71.8/84.9	87.1/94.1	83.7/90.8	77.4/86.4	81.2/90.7	70.0/81.6	80.7/89.1	79.1/86.1	81.2/88.9	81.0/89.7
Memento (full)	**86.9/96.4**	**76.6/91.7**	**89.3/96.9**	**87.1/95.6**	**82.2/93.5**	**84.2/95.4**	**77.3/90.8**	**83.3/95.0**	**85.7/94.8**	**86.3/95.6**	**84.7/95.0**

Qwen2.5-VL-7B (Bai et al., 2025) Executor and tool environment, and the only thing that differs across methods is the prompt template through which memory is injected.

Three prompt variants. All methods share the *same* off-the-shelf Qwen2.5-VL-7B model with no SFT/RL fine-tuning; they differ only in the extra prompt that is prepended to the Executor's context at inference time:

- **No extra prompt.** Nothing is prepended; the Executor is queried without any memory channel. This variant is used for the *No Memory* baseline.
- **Long-context memory prompt.** For methods that inject retrieved memory directly, we prepend

  `Here are some memories for your reference:\n{memory context}\n`

  where `{memory context}` is the retrieved memory. This variant is used for RAG, Mem0 and A-MEM.
- **Guideline (plan) prompt.** For methods that distil memory into a high-level plan, we prepend

  `Here is a guide for your reference:\n{plan}\nBegin your answer:\n`

  where `{plan}` is filled with the abstracted plan generated from memory. This variant is used for ReasoningBank, ExpeL and *Memento*.

At evaluation time, each method is paired with the prompt variant matching its memory-injection mode; the underlying Qwen2.5-VL-7B weights are identical across all rows.

Tools and retrievers. While running the Executor over FVQA-train to populate the case bank, the text retriever uses E5-base-v2 with a FAISS index over a wiki25 corpus (top-3 passages per query) and an offline image-to-image retriever based on cached Serper image-search results (top-3 images). For evaluation, we additionally enable the online Serper text-search tool (top-5 results); image search remains via Serper. Inference uses vLLM at temperature 0.

Dataset settings. Table 11 summarises the multimodal and text-only datasets used in Tables 2 and 13. FVQA-train is used only as the source pool from which memory-based methods accumulate in-domain (s, a, r) experience for their case bank, with no LLM weights updated; all other datasets

are evaluation-only. GAIA-Text is the 103-example text-only subset of GAIA used by Mo et al. (2025).

Table 11: Datasets used in the multimodal/text-only memory-based search-agent evaluation (Tables 2 and 13).

Dataset	**Modality**	**# Examples**	**Usage**
FVQA-train (Wang et al., 2018)	Image-text	4,856	Memory case-bank source
FVQA-test (Wang et al., 2018)	Image-text	1,800	In-domain evaluation
InfoSeek (Chen et al., 2023)	Image-text	2,000	OOD evaluation
LiveVQA (Fu et al., 2025)	Image-text	2,384	OOD evaluation
SimpleVQA (Cheng et al., 2025)	Image-text	1,013	OOD evaluation
MMSearch (visual subset) (Wu et al., 2025)	Image-text	171	OOD evaluation
SimpleQA (Wei et al., 2024)	Text-only	4,327	OOD evaluation
2Wiki (Ho et al., 2020)	Text-only	12,576	OOD evaluation
HotpotQA (Yang et al., 2018)	Text-only	7,405	OOD evaluation
GAIA-Text (Mo et al., 2025)	Text-only	103	OOD evaluation

N Full UIS-QA Leaderboard

Table 12: Full UIS-QA leaderboard. †: reasoning-oriented LLM. Action-space columns: Cr (crawl), Vi (visual), Fi (file), Br (browser).

Method	Action Space				Backbone	UIS-QA
	Cr	Vi	Fi	Br		
Direct Inference						
DeepSeek-V3.1	–	–	–	–	DeepSeek-V3.1	1.8
Claude-sonnet-4	–	–	–	–	Claude-S4	2.7
GPT-5	–	–	–	–	GPT-5	0.9
Commercial System						
GLM-4.5	✓	×	–	✓	GLM4.5†	11.8
Doubao	–	–	–	–	Doubao	11.8
Gemini-2.5-pro	–	–	–	–	Gemini-2.5-pro	4.5
ReAct-based Framework						
WebSailor	✓	×	×	×	WebSailor-32B + Qwen3-72B	7.3
Tongyi-DR	✓	×	×	×	TongyiDR-30B-A3B†+GPT-4o	23.6
Multi-agent Framework						
DDv2	✓	✓	✓	✓	Pangu-38B	8.2
OWL (Camel-AI, 2025)	✓	✓	✓	✓	O3-mini + 4o + Claude-S3.7	4.6
MiroThinker v0.1	✓	✓	✓	✓	MiroThinker-32B + GPT-4.1 + Claude-S3.7	7.3
AWorld	✓	✓	✓	✓	Gemini-2.5-pro + GPT-4o	5.5
UIS-Digger (Liu et al., 2026)						
(Pangu)	✓	✓	✓	✓	PanGu-38B	27.3
(Qwen)	✓	✓	✓	✓	Qwen3-32B	27.3
Ours						
Memento w/o CBR	✓	✓	✓	✓	O3 + GPT-4.1	25.5
Memento w/ non-parametric CBR	✓	✓	✓	✓	O3 + GPT-4.1	**36.4**

O Text-Only Memory-Based Search-Agent Results

Table 13 supplements the main seven-dataset results in Table 1 under a separate protocol. It evaluates a shared Qwen2.5-VL-7B executor and tool environment on SimpleQA, 2Wiki, HotpotQA, and GAIA-Text. All four are out-of-domain relative to the FVQA-train case bank; dataset sizes and the

Table 13: Text-only comparison (accuracy, %; shared Qwen2.5-VL-7B executor). Scores are point estimates. The *Memento* rows compare no Case Memory with non-parametric retrieval; GAIA-Text is the 103-example text-only subset of Mo et al. (2025).

Model	Out-of-Domain				Avg
	SimpleQA	2Wiki	HotpotQA	GAIA-Text	
RAG (Lewis et al., 2020)	38.3	56.3	47.5	14.6	39.2
Mem0 (Chhikara et al., 2025)	38.1	54.9	49.0	16.5	39.6
A-MEM (Xu et al., 2025)	38.8	56.2	47.5	12.6	38.8
ReasoningBank (Ouyang et al., 2025)	42.4	61.0	52.7	14.6	42.7
ExpeL (Zhao et al., 2024)	**43.0**	63.4	**55.5**	20.4	45.6
Memento without Case Memory	40.7	61.2	51.0	11.7	41.2
Memento non-parametric	42.4	**64.2**	55.2	**22.3**	**46.0**

common protocol are in Appendix M. Non-parametric *Memento* achieves the highest reported average, 46.0, versus 41.2 without Case Memory and 45.6 for ExpeL. It leads on 2Wiki and GAIA-Text, while ExpeL leads on SimpleQA and HotpotQA. These point estimates support the breadth of the memory comparison; the small gap to ExpeL is descriptive. Learned-retrieval results are not reported for this four-dataset setting.

P Operational Behaviour of *Memento*

We close with a brief look at how *Memento* actually operates at run time. Two observations stand out: (i) tool usage scales sub-linearly with task difficulty, with reasoning increasingly absorbed into the LLM rather than into extra tool calls; and (ii) the dominant cost component is input tokens devoted to integrating prior tool outputs, not output generation.

Tool-call distribution across difficulty levels. Figure 6 reports the average number of tool invocations per task on GAIA Levels 1–3. Code execution, web search, and crawling dominate at every level, and their usage grows monotonically with difficulty. The growth is, however, sub-linear: the hardest tasks rely relatively more on the LLM's internal aggregation of prior tool outputs than on issuing additional tool calls. This is consistent with the role our framework assigns to memory: once relevant evidence is in the case bank, further progress is gated by interpretation rather than acquisition.

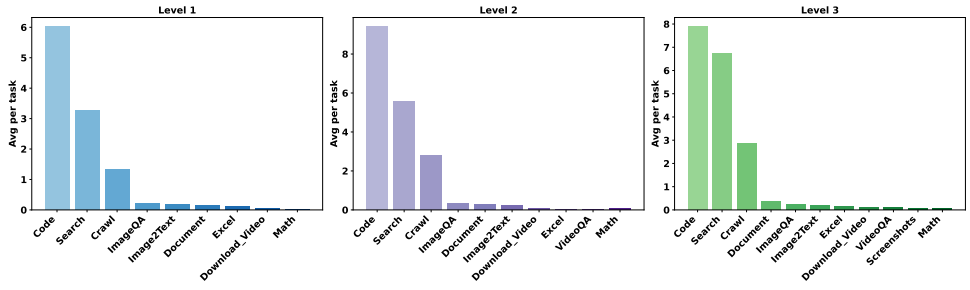


Figure 6: Average number of tool invocations per task on GAIA, broken down by tool family and difficulty level. Code, search, and crawl dominate, and their counts grow with difficulty, but sub-linearly, indicating that long-horizon tasks are increasingly bound by reasoning rather than by extra tool calls.

Token cost per task. On a random sample of 10 GAIA-validation tasks per level (Figure 7), input-token cost rises sharply with difficulty: ∼26k input / 4.7k output tokens at Level 1, ∼48k / 6.9k at Level 2, and ∼121k / 9.8k at Level 3. Output tokens stay roughly stable across levels because the final answer is short regardless of difficulty; the explosion is entirely on the input side,

where multi-step tool outputs, intermediate plans, and reflection traces accumulate. This isolates the dominant operational bottleneck for long-horizon Deep Research: not generation, but evidence integration.

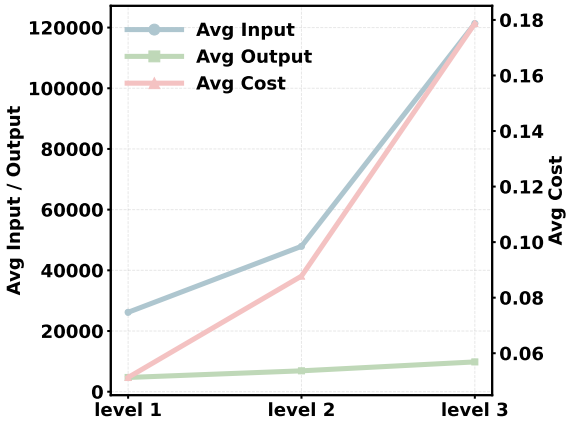


Figure 7: Token cost per task on GAIA validation, averaged over 10 random tasks per level. Input tokens scale roughly linearly with task complexity, while output tokens stay flat, confirming that the cost of long-horizon Deep Research is dominated by reading and integrating tool outputs, not by writing answers.