AGENT KB: A Hierarchical Memory Framework for Cross-Domain Agentic Problem Solving

Anonymous Authors[1]

Abstract

As language agents tackle increasingly complex tasks, they struggle with effective error correction and knowledge reuse across different domains. We present Agent KB, a hierarchical memory framework that enables cross-domain agent learning through a novel Reason-Retrieve-Refine pipeline. Our dual-phase approach combines workflow-level knowledge retrieval with targeted execution pattern refinement, allowing agents to break free from limited reasoning pathways by incorporating diverse problem-solving strategies. Evaluations on GAIA benchmark demonstrate substantial performance gains, with Agent KB improving success rates by up to **16.28** percentage points overall. Most notably, on challenging tasks, Claude-3.7 with Agent KB increased performance from 38.46% to 57.69%, while GPT-4.1 showed similar improvements on intermediate tasks (53.49% to 73.26%). For SWE-bench code repair tasks, our system significantly improved resolution rates, with Claude-3.7 achieving a **12.0** percentage point gain (41.33% to 53.33%). Agent KB provides a modular, agent-agnostic infrastructure that facilitates continuous improvement through knowledge sharing across task boundaries and agent architectures. Our code is publicly available at https://anonymous.4open.science/r/agent_kb-35C6/.

1. Introduction

As artificial intelligence advances, language agents are becoming increasingly vital for solving complex problems (Chan et al., 2023; Hong et al., 2023; Guo et al., 2024; Liu et al., 2025b). While these agents have shown impressive capabilities through supervised learning, they continue to struggle with complex, long-horizon tasks requiring sophisticated planning and tool use (Jimenez et al., 2023; Huang et al., 2024; Xiong et al., 2025). The integration of autonomous improvement modules has demonstrated performance gains (Zheng et al., 2023; Zhang et al., 2024b; Wang et al., 2024b; Hu et al., 2024; Shah et al., 2025; Xu et al., 2025), yet a critical bottleneck persists.

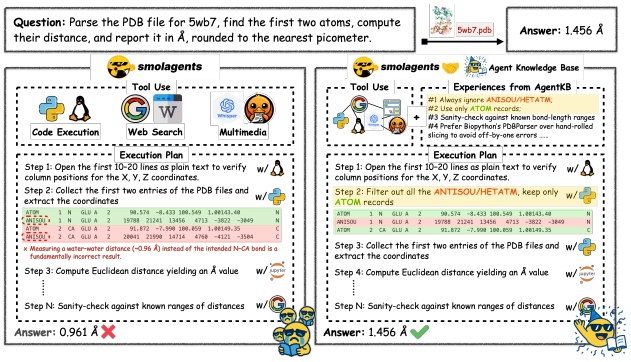

Figure 1. **Comparison of PDB distance-calculation workflows with and without AGENT KB. (A) Original pipeline:** indiscriminately reads the first two ATOM/HETATM/ANISOU lines, often selecting solvent records and yielding a spurious O–H distance (0.961 Å). **(B) AGENT KB-enhanced agent workflow:** applies memory-driven rules—filter out all ANISOU/HETATM, use only genuine ATOM entries in file order, and sanity-check against known N–CA bond-length ranges—to correctly extract the backbone N–CA pair and report the distance of 1.456 Å.

The fundamental limitation lies in error correction during complex problem-solving. When agents encounter difficulties, self-feedback proves insufficient—they lack access to the diverse problem-solving strategies and implicit reward signals that guide human experts. Recent work (Wang et al., 2024b) shows that learning reusable workflows from past experiences improves performance, yet current approaches remain limited to task-specific memories that operate in isolation. Agents cannot benefit from experiences across different tasks, domains, or frameworks, forcing them to repeatedly rediscover solutions to similar problems (Silver & Sutton, 2025).

To understand why current approaches fall short, we identify three critical design flaws in agent memory systems:

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

(1) Agent Isolation—agents cannot learn from others' successes or access diverse problem-solving strategies beyond their own experience. Single-framework experiences contain inherent reasoning biases constrained by their implementation design, leaving the cognitive diversity from different agent frameworks—each with unique reasoning patterns and optimization objectives—largely untapped. **(2) Undifferentiated Knowledge Organization**—retrieval mechanisms fail to distinguish between high-level workflow planning and precise execution details, preventing effective knowledge adaptation. **(3) Retrieval Without Reasoning**—systems attempt direct knowledge matching without first engaging in preliminary reasoning to determine relevant knowledge targets.

We propose the Agent Knowledge Base (AGENT KB), a framework that transforms how agents utilize cross-domain experiences through our novel *Reason-Retrieve-Refine* pipeline. Unlike existing systems, AGENT KB first engages agents in preliminary reasoning about the problem, directing subsequent knowledge retrieval toward relevant solution patterns rather than merely matching surface features. Our teacher-student dual-phase retrieval mechanism addresses the key challenge of knowledge application: student agents first retrieve workflow-level patterns to structure their approach, while teacher agents subsequently identify specific execution patterns to refine implementation details. This hierarchical process enables agents to break out of their limited reasoning pathways by incorporating diverse problem-solving strategies from external sources, providing implicit reward signals that guide refinement toward successful solutions.

Our experimental evaluations on the GAIA benchmark demonstrate substantial performance gains, with AGENT KB-enhanced models achieving improvements of up to 16.28 percentage points in overall success rates. Notably, on medium-difficulty GAIA tasks (Level 2), `GPT-4.1` with +AGENT KB ✓♡ (as defined in Section 4.1) shows remarkable improvement from 53.49% to 73.26% success rate. Even more impressive gains are observed on challenging Level 3 tasks, where `Claude-3.7` with +AGENT KB ✓♡ increases performance from 38.46% to 57.69%, demonstrating AGENT KB's effectiveness in bridging the capability gap for complex problem solving. For issue resolving tasks in SWEbench, our ablation studies reveal that the hybrid retrieval approach outperforming both pure text similarity and semantic similarity methods. Further analysis shows that automatically generated knowledge sometimes outperforms manually crafted examples, highlighting the value of our knowledge acquisition pipeline in capturing and structuring diverse agent experiences. Designed to be modular and agent-agnostic, AGENT KB retrieves experiences from other tasks to bootstrap decision making.

2. Related Work

2.1. Memory Systems in LLM Agents

Memory systems in LLM agents have evolved from simple storage mechanisms to sophisticated architectures supporting complex reasoning (Piao et al., 2025; Zeng et al., 2024; Liu et al., 2025b; Zhang et al., 2024a). Early implementations like MemoryLLM (Wang et al., 2024a) embedded knowledge in the latent space, while subsequent approaches introduced structured organization through Zettelkasten-style-graph-based systems (A-MEM (Xu et al., 2025), AriGraph (Anokhin et al., 2024)) and hierarchical frameworks (MemGPT (Packer et al., 2023), Unified Mind Model (Hu & Ying, 2025)). Knowledge integration approaches address planning capabilities and hallucination mitigation through frameworks like Agent Workflow Memory (Wang et al., 2024b), which enables automatic induction and reuse of sub-workflows, and KnowAgent (Zhu et al., 2024), which augments prompts with action-knowledge bases. More sophisticated approaches include parametric world-knowledge models (WKM) (Qiao et al., 2024) and multi-agent adaptation systems MARK (Ganguli et al., 2025). EcoAssistant (Zhang et al., 2024a) demonstrated the effectiveness of knowledge reuse and transfer across agents, establishing a foundation for collaborative reasoning ReAct (Yao et al., 2022) synergizes reasoning and acting by interleaving chain-of-thought with tool calls, allowing real-time plan adaptation, while Reflexion (Shinn et al., 2023) enables agents to learn from verbalized self-critiques. Toolformer (Schick et al., 2023) demonstrates that LLMs can learn to use external tools in an unsupervised manner, patching capability gaps mid-execution. Retrieval mechanisms for memory have progressed beyond basic RAG paradigms (Lewis et al., 2020), with innovations like HippoRAG's (Gutiérrez et al., 2024) hippocampal-inspired indexing, Echo's (Liu et al., 2025a) temporal cues, and HiAgent's (Hu et al., 2024) sub-goal chunking.

2.2. Multi-Agent Collaboration and Shared Memory

Most existing memory systems remain agent-specific, designed for recalling interaction history (Lu et al., 2023), modeling user preferences (Zhong et al., 2024), and etc. Memory-augmented embodied agents (Glocker et al., 2025) have begun to explore collaborative architectures, where specialized agents (routing, planning, knowledge base) work together, leveraging in-context learning and RAG to retrieve context from past interactions. However, these systems typically maintain separate memory structures rather than a unified knowledge ecosystem. Limited work exists on cross-agent knowledge sharing and adaptation. Synapse (Zheng et al., 2023) introduces exemplar memory for trajectory storage but primarily focuses on single-agent contexts. EventWeave (Zhao et al., 2025) addresses incomplete context tracking by identifying both core and supporting events in a dynamic event graph but doesn't fully extend to multi-agent

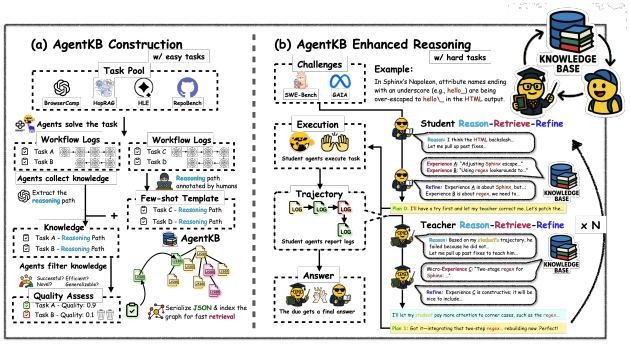

Figure 2. System architecture of AGENT KB, showing the integration of knowledge abstraction, dual-phase retrieval, and adaptive refinement into a unified framework. The dual-phase retrieval framework of AGENT KB. The student agent retrieves workflow-level patterns for structuring the approach, while the teacher agent retrieves step-level patterns for execution precision. Cross-agent and cross-domain knowledge transfer through adaptive refinement. Knowledge is dynamically adapted rather than directly copied, enabling effective transfer even between dissimilar domains

scenarios. Some researchers have explored pre-conditions for memory-learning agents (Shah et al., 2025), revealing that memory induction quality significantly impacts performance. This suggests that creating high-quality shared memory structures could benefit multiple agents simultaneously, particularly if stronger agents can induce memories that weaker agents can later leverage. Case-Based Reasoning (CBR) approaches (Hatalis et al., 2025) provide promising directions for multi-agent knowledge sharing, as they enable solving new problems by referencing past experiences.

3. Methodology

As shown in Figure 2, AGENT KB consists of two interconnected components: *knowledge base construction* and *dual-phase inference*. These innovations are achieved through our novel pipeline `Reason-Retrieve-Refine` that both the student and the teacher agents implement during different phases of problem solving.

3.1. AGENT KB Construction

During construction, we transform successful agent workflows from diverse tasks into abstracted knowledge patterns through systematic generalization operations. Our fundamental hypothesis is that experience gained from simpler tasks provides substantial benefits when addressing novel, more complex challenges.

The process begins by collecting execution logs from previously completed tasks across various domains. These logs undergo quality assessment to select the most valuable experiences based on success rates, efficiency, and generalizability. We incorporate human expert annotations as few-shot examples to help agents better synthesize and abstract experiential knowledge. These patterns are organized in a hierarchical graph structure for efficient retrieval.

Formally, each source *experience* is structured as a tuple $\mathcal{E} = \langle \pi, \gamma, \mathcal{S}, \mathcal{C}, \mathcal{R} \rangle$, where π represents the problem; γ denotes the goal or objective; $\mathcal{S} = \{s_1, s_2, \ldots, s_n\}$ is an abstracted solution trajectory with reasoning templates, optionally with observed failure modes; $\mathcal{C}$ captures problem characteristics such as domain and difficulty level; $\mathcal{R}$ contains relational links to other knowledge patterns in the hierarchical structure of AGENT KB. Rather than storing raw experiences, our knowledge base maintains abstracted reasoning patterns, creating a more generalizable and efficient knowledge structure.

3.2. Teacher-Student Dual-Phase Inference

We implement a hierarchical teacher-student framework, where both agents operate using complementary **Reason-Retrieve-Refine** (RRR) cycles to solve complex tasks. The teacher supervises the student by detecting and correcting errors to enhance overall performance.

In the **student phase**, the agent first analyzes query Q to identify the problem ($\hat{\pi}$) and goal ($\hat{\gamma}$), generating initial thoughts $\mathcal{T}$ about potential solutions. Next, it retrieves relevant workflow patterns from the knowledge base:

$$\mathcal{E}_w = \underset{\mathcal{E}_i \in \mathcal{K}}{\text{top-k}} \big[\alpha \cdot \phi_r(\mathcal{E}_i, \mathcal{T}, \hat{\pi}, \hat{\gamma}) + \beta \cdot \phi_s(\mathcal{E}_i)\big],$$

where $\mathcal{K}$ is the knowledge base, ϕ_r measures relevance, ϕ_s assesses historical success, and α, β are weights. The student then refines these workflows by integrating them with initial reasoning to create and execute a structured plan, resulting in a series of reasoning steps.

In the **teacher phase**, the agent evaluates the student's reasoning steps by summarizing them and identifying errors along with their types and causes. It retrieves targeted step-level experiences from the knowledge base to address these execution issues:

$$\mathcal{E}_s = \underset{\mathcal{E}_j \in \mathcal{K}}{\text{top-m}} \sum_{s_i \in \mathcal{Z}} \big[\alpha \cdot \phi_r(s_i, \mathcal{S}_j) + \beta \cdot \phi_p(\mathcal{E}_j)\big],$$

where ϕ_r measures similarity and ϕ_p evaluates precision quality. The teacher refines these step-level patterns into precise guidance, providing targeted interventions. This iterative feedback loop progressively enhances the student's performance.

4. Experiment

4.1. Setup

Datasets Our evaluation employs two representative benchmarks that assess diverse agent capabilities. The GAIA benchmark (Mialon et al., 2023) provides a comprehensive evaluation framework for general AI assistants, containing 165 evaluation instances carefully stratified across three difficulty levels: 53 tasks in Level 1 (basic), 86 tasks in Level 2 (intermediate), and 26 tasks in Level 3 (advanced). These tasks span information retrieval, multi-step reasoning,

and complex problem-solving scenarios. The SWE-Bench (Jimenez et al., 2023) serves as our second benchmark, focusing on realistic software engineering challenges extracted from GitHub issues, requiring agents to understand existing codebases and implement appropriate fixes.

The knowledge base for AGENT KB draws from diverse sources. For general assistant tasks, we aggregate experiences from four complementary datasets: BrowseComp (Wei et al., 2025) (1,266 tasks), HopRAG (Liu et al., 2025c) (2,556 tasks), a text-based subset of HLE (Phan et al., 2025) (3,000 tasks), and WebWalkerQA (Wu et al., 2025) (680 tasks). For software engineering knowledge, we incorporate structured experiences from three major repositories: RepoClassBench (Deshpande et al., 2024), SWE-Gym-Raw (Pan et al., 2024), and RepoEval (Zhang et al., 2023), collectively comprising approximately 3,000 structured problem-solving traces.

Model Configurations We evaluate three distinct configurations across multiple foundation models to assess the effectiveness of AGENT KB. We use smolagents [1] without any knowledge integration to serve as our base agent framework. To enhance performance on complex tasks, we augment smolagents with audio-visual comprehension modules and a multi-source retrieval system, thereby improving multimodal input processing and facilitating more efficient access to diverse information sources. For SWE-Bench benchmark, we employ OpenHands framework [2] as our base agent framework. Default settings are used for all hyperparameters unless noted. The +AGENT KB configuration implements a two-round, teacher-student knowledge transfer process: first, the student agent attempts to solve the task; then the teacher agent reviews the student's work, searches the knowledge base for relevant experiences, and provides feedback without knowing whether the student's solution was correct (unsupervised). The student agent then makes a second attempt incorporating this feedback. The +AGENT KB ✓ configuration enhances this approach by providing supervision signals to the teacher agent, explicitly indicating whether the student's initial solution was correct. This allows the teacher to focus more precisely on understanding why the solution succeeded or failed and provide more targeted guidance. The teacher still must independently analyze the student's reasoning to identify specific errors or correct approaches before providing feedback for the student's second attempt. To ensure fair comparison with existing baselines that employ various performance-enhancing techniques, we incorporate equivalent optimization methods by +AGENT KB ✓♡ across all configurations, including optimized retrieval mechanisms, fine-grained knowledge extraction patterns, majority voting across multiple solution candidates, and consistent output formatting corrections.

[1] https://github.com/huggingface/smolagents

[2] https://github.com/All-Hands-AI/OpenHands

4.2. Main Results

In Table 1, our approach demonstrates significant improvements over baselines across all GAIA's difficulty levels. `GPT-4.1` with +AGENT KB ✓♡ shows an overall improvement of 18.79 percentage points, with the largest gains (19.77 points) observed in medium-difficulty tasks (Level 2). Claude models exhibit similar benefits from AGENT KB integration, with `Claude-3.7` with +AGENT KB ✓♡ improving from 58.79% to 75.15% in overall performance. Figure 3 also demonstrates consistent performance improvements across all six base LLMs tested. A 19.23 percentage point gain (`Claude-3.7` rising from 38.46% to 57.69%) in the most complex scenario category (level 3) validates our approach's effectiveness in supporting sophisticated multi-step reasoning and planning. Such improvements indicate that the bottleneck in handling complex tasks lies in their ability to effectively leverage relevant past experiences.

Notably, the +AGENT KB ✓♡ -enhanced `Claude-3.7` model achieves an average GAIA score of 75.15%, surpassing closed-source systems like h2oGPTe (63.64%) and open-source frameworks like OWL (69.09%). This performance is particularly impressive given that our approach builds upon a relatively straightforward agent framework (smolagents).

For the SWE-bench lite benchmark (Jimenez et al., 2023), we set the max limit for agent iterations to 50 and 100 and conduct experiments respectively. Table 2 shows similar patterns of improvement across different model types. `Claude-3.7` achieves the most substantial gains, with performance increasing from 30.00% to 51.00% at 50 iterations. Interestingly, we observe that the relative magnitude of improvement correlates with model sophistication, with larger and more capable models like `Claude-3.7` and `GPT-4.1` showing more substantial gains than smaller models like `Qwen-3 32B`. This suggests that more advanced models are better able to leverage the retrieved knowledge, potentially due to their enhanced reasoning capabilities.

Figure 3. Score improvements (%) across difficulty levels for multiple base LLMs enhanced with AGENT KB.

Table 1. Performance of various agent frameworks on GAIA benchmark

Method	Models	Average	Level 1	Level 2	Level 3
Single Model					
Search-o1-32B (Li et al., 2025a)	-	39.8	53.8	34.6	16.7
WebThinker-32B-RL (Li et al., 2025b)	-	48.5	56.4	50.0	16.7
Closed-source Agent Frameworks					
Langfun Agent (Peng, 2023)	Claude 3.7	71.52	83.02	68.60	**57.69**
TraseAgent (Trase, 2024)	Claude	70.30	83.02	69.77	46.15
Deep Research (OpenAI, 2024)	Unknown	67.36	74.29	69.06	47.60
h2oGPTe (H2O.ai, 2024)	Claude-3.5	63.64	67.92	67.44	42.31
Desearch (AI, 2024)	GPT-4o	56.97	71.70	58.14	23.08
Open-Source Agent Frameworks					
AWorld (at Ant Group, 2025)	DeepSeek V3	69.70	**86.79**	69.77	34.62
OWL (Hu et al., 2025)	Claude 3.7	69.09	84.91	67.44	42.31
TapeAgents (Bahdanau et al., 2024)	Claude 3.7	55.76	71.70	53.49	30.77
AutoAgent (Tang et al., 2025)	Claude 3.5	55.15	71.70	53.40	26.92
smolagents (LangChain, 2024)	OpenAI o1	55.15	67.92	53.49	34.62
Magnetic-1 (Fourney et al., 2024)	OpenAI o1	46.06	56.60	46.51	23.08
FRIDAY (Wu et al., 2024)	GPT-4 turbo	34.55	45.28	34.88	11.54
smolagents Baseline	GPT-4.1	55.15	67.92	53.49	34.62
smolagents +AGENT KB	GPT-4.1	61.21 ↑6.06	79.25 ↑11.33	58.14 ↑4.65	34.62
smolagents +AGENT KB ✓	GPT-4.1	67.27 ↑12.12	83.02 ↑15.07	67.44 ↑13.95	34.62
smolagents +AGENT KB ✓♡	GPT-4.1	73.94 ↑18.79	84.91 ↑16.99	73.26 ↑19.77	53.85 ↑19.23
smolagents Baseline	Claude 3.7	58.79	64.15	61.63	38.46
smolagents +AGENT KB	Claude 3.7	65.45 ↑6.66	75.47 ↑11.32	66.28 ↑4.65	38.46
smolagents +AGENT KB ✓	Claude 3.7	69.70 ↑10.91	79.25 ↑15.1	69.77 ↑8.14	50.00 ↑11.54
smolagents +AGENT KB ✓♡	Claude 3.7	**75.15** ↑16.36	84.91 ↑20.76	**74.42** ↑12.79	**57.69** ↑19.23

Table 2. Main results on the SWE-bench lite with maximum iteration limits of 50 and 100.

Method	Models	Max Iter 50 Success Rate	Max Iter 100 Success Rate
OpenHands Baseline	GPT-4o	16.33	26.00
OpenHands +AGENT KB	GPT-4o	20.33 ↑+4.00	29.67 ↑+3.67
OpenHands +AGENT KB ✓	GPT-4o	29.33	35.67
OpenHands +AGENT KB ✓♡	GPT-4o	**31.33**	**39.33**
OpenHands Baseline	GPT-4.1	24.33	28.67
OpenHands +AGENT KB	GPT-4.1	28.33 ↑+4.00	31.67 ↑+3.00
OpenHands +AGENT KB ✓	GPT-4.1	37.33	42.33
OpenHands +AGENT KB ✓♡	GPT-4.1	**38.67**	**45.67**
OpenHands Baseline	o3-mini	23.00	29.33
OpenHands +AGENT KB	o3-mini	31.67 ↑+8.67	33.67 ↑+4.34
OpenHands +AGENT KB ✓	o3-mini	35.33	36.33
OpenHands +AGENT KB ✓♡	o3-mini	**37.00**	**40.00**
OpenHands Baseline	Claude-3.7	30.00	41.33
OpenHands +AGENT KB	Claude-3.7	46.67 ↑+16.67	48.33 ↑+7.00
OpenHands +AGENT KB ✓	Claude-3.7	49.67	51.67
OpenHands +AGENT KB ✓♡	Claude-3.7	**51.00**	**53.33**
OpenHands Baseline	DeepSeek-R1	24.33	30.00
OpenHands +AGENT KB	DeepSeek-R1	26.67 ↑+2.34	33.33 ↑+3.33
OpenHands +AGENT KB ✓	DeepSeek-R1	31.00	35.67
OpenHands +AGENT KB ✓♡	DeepSeek-R1	**32.67**	**37.33**
OpenHands Baseline	Qwen-3 32B	18.33	25.67
OpenHands +AGENT KB	Qwen-3 32B	20.67 ↑+2.34	28.67 ↑+3.00
OpenHands +AGENT KB ✓	Qwen-3 32B	28.67	34.33
OpenHands +AGENT KB ✓♡	Qwen-3 32B	**30.33**	**36.67**

4.3. Ablation Studies

To assess the contribution of each core component in AGENT KB, we conduct systematic ablation studies. The full system achieves an average score of 61.21% on GAIA.

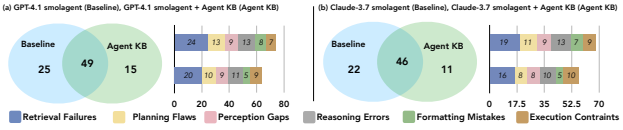

Figure 4. The frequency of errors with and without AGENT KB. The Venn diagrams quantify overlapping and unique failure cases, while the horizontal bar charts show category-specific error counts.

Removing either the student or teacher agent reduces performance to 59.39%, highlighting their complementary roles in the dual-phase architecture. Notably, the student agent is especially important for Level 1 tasks (a drop from 79.25% → 75.47%), suggesting its key role in planning simpler workflows. In contrast, removing the teacher agent leads to a sharper decline in Level 1 accuracy (79.25% → 73.58%), indicating its role in early-stage refinement. The most significant drop occurs when the `Refine` module is removed, decreasing overall accuracy by 6.06 percentage points (61.21% → 55.15%) and Level 3 performance by 3.85 points (34.62% → 30.77%), underscoring the necessity of fine-grained error correction. Ablating the *Retrieve* module also yields notable degradation (-3.63 points), demonstrating that knowledge grounding via retrieval is essential. In contrast, omitting the `Reason` module causes only a modest drop (-1.21), implying that retrieval and refinement can partially compensate for missing high-level planning. Finally, replacing structured experiences with raw workflow logs reduces performance to 58.18%, reaffirming the importance of knowledge abstraction and reuse beyond naive trajectory replay. These

results validate that reasoning, retrieval, and refinement each contribute distinct and synergistic improvements, with the refinement phase playing a particularly critical role in ensuring execution correctness on challenging tasks.

To better understand the factors contributing to AGENT KB's effectiveness, we conduct an in-depth analysis of different retrieval strategies across abstraction levels (Figure 5). Using `GPT-4.1` as our base model with top-k=3, we compare three retrieval approaches (text similarity, semantic similarity, and hybrid retrieval) across two complementary abstraction methods that we integrate in our full system. Our implemented retrieval system combines both summary-based and criticism-based approaches. The summary-based method transforms execution logs into concise summaries through refinement, while the criticism-based approach prompts teacher agents to reason about potential errors in execution logs. We then perform separate retrievals using each abstraction method before integrating the results. Figure 5 demonstrates their distinct contributions.

For summary-based retrieval (left panels), hybrid methods consistently outperform single-approach strategies, achieving 83% accuracy on Level 1 GAIA tasks and 37% on SWE-bench lite. The performance advantage is particularly pronounced for Level 1 and 2 tasks, where hybrid retrieval shows improvements of up to 9 percentage points over semantic-only approaches. Criticism-based retrieval (right panels) exhibits a different pattern, with text similarity performing competitively for Level 2 tasks (67%) and semantic similarity showing stronger results on SWE-bench (33%). Hybrid approaches maintain their edge in most scenarios, though with narrower margins.

4.4. Error Analysis

For `GPT-4.1` (Figure 4 a), we observe that 49 errors occur in both configurations, while 25 errors specific to the baseline were successfully corrected by AGENT KB. The enhanced model introduced only 15 new errors, yielding a net error reduction of 10 instances. Similarly, with `Claude-3.7` (Figure 4 b), 46 errors persist across both configurations, while AGENT KB corrects 22 baseline-specific errors and introduces just 11 new ones, resulting in a net improvement of 11 instances. The bar charts reveal the distribution of error types. The authors manually reviewed and categorized each error case through a systematic annotation process to ensure accurate classification across six distinct error categories. For `GPT-4.1`, retrieval errors decreased from 24 to 20 instances, and planning errors from 13 to 10. `Claude-3.7` demonstrates even more pronounced improvements in retrieval (19 to 16) and reasoning errors (13 to 8). This improvement stems from AGENT KB's knowledge base containing analogous search protocols and workflows, allowing agents to accumulate expertise through standardized pathways and successful planning precedents. Formatting errors also decreased significantly as agents adopt format requirements derived from similar experiences, contributing to more precise output specifications. While image and video comprehension tasks remain constrained by underlying tool capabilities, AGENT KB-enhanced agents still formulate more appropriate plans for visual tool utilization. Furthermore, the knowledge base helps reduce task hallucinations, resulting in more streamlined planning steps that minimize context length and information loss during complex reasoning processes. Interestingly, while both models show similar patterns of improvement, `Claude-3.7` experiences greater error reduction in reasoning tasks, whereas `GPT-4.1` benefits more in perception gap resolution, highlighting how AGENT KB's effectiveness complements each model's inherent strengths and weaknesses.

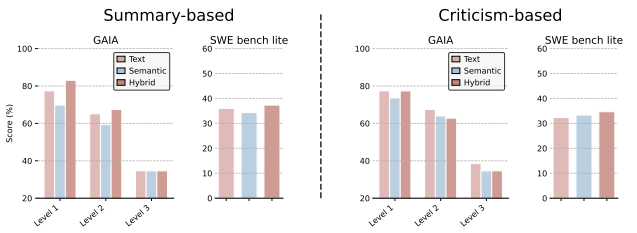

Figure 5. Performance comparison of text, semantic, and hybrid retrieval methods across two different abstraction levels. The left panels show results for summary-based retrieval, while the right panels show criticism-based retrieval.

Figure 4 illustrates the impact of AGENT KB on error patterns across different base LLMs configurations. The Venn diagrams provide a quantitative comparison of errors between smolagents framework and its AGENT KB-enhanced counterparts.

5. Conclusion

We introduce AGENT KB, a unified and scalable framework that enables LLM agents to continuously learn from experience across tasks, domains, and agent architectures. By structuring prior workflows into generalizable experience units and supporting their reuse through a dual-phase, teacher-student retrieval and refinement pipeline, AGENT KB moves beyond simple memory replay to realize adaptive, experience-driven reasoning. Our experiments across diverse settings—including GAIA and SWE-bench—demonstrate consistent performance improvements across difficulty levels, model families, and agent frameworks. Notably, AGENT KB 's structured knowledge abstraction and dual-phase inference enable not only effective reuse of past solutions but also the evolution of better workflows through agent collaboration. These results position AGENT KB as a general-purpose infrastructure for scalable, continual improvement in agent ecosystems, bridging the gap between episodic memory and cumulative agent intelligence.

References

AI, D. Desearch, 2024. URL https://desearch.ai/.

Anokhin, P., Semenov, N., Sorokin, A., Evseev, D., Burtsev, M., and Burnaev, E. Arigraph: Learning knowledge graph world models with episodic memory for llm agents. *arXiv preprint arXiv:2407.04363*, 2024.

at Ant Group, A. T. Aworld: A unified agent playground for computer and phone use tasks, 2025. URL https://github.com/inclusionAI/AWorld. Version 0.1.0. GitHub. Contact: chenyi.zcy@antgroup.com.

Bahdanau, D., Gontier, N., Huang, G., Kamalloo, E., Pardinas, R., Piché, A., Scholak, T., Shliazhko, O., Tremblay, J. P., Ghanem, K., Parikh, S., Tiwari, M., and Vohra, Q. Tapeagents: a holistic framework for agent development and optimization, 2024. URL https://arxiv.org/abs/2412.08445.

Chan, C.-M., Chen, W., Su, Y., Yu, J., Xue, W., Zhang, S., Fu, J., and Liu, Z. Chateval: Towards better llm-based evaluators through multi-agent debate. *arXiv preprint arXiv:2308.07201*, 2023.

Deshpande, A., Agarwal, A., Shet, S., Iyer, A., Kanade, A., Bairi, R., and Parthasarathy, S. Class-level code generation from natural language using iterative, tool-enhanced reasoning over repository. *arXiv preprint arXiv:2405.01573*, 2024.

Fourney, A., Bansal, G., Mozannar, H., Tan, C., Salinas, E., Niedtner, F., Proebsting, G., Bassman, G., Gerrits, J., Alber, J., et al. Magentic-one: A generalist multi-agent system for solving complex tasks. *arXiv preprint arXiv:2411.04468*, 2024.

Ganguli, A., Deb, P., and Banerjee, D. Mark: Memory augmented refinement of knowledge. *arXiv preprint arXiv:2505.05177*, 2025.

Glocker, M., Hönig, P., Hirschmanner, M., and Vincze, M. Llm-empowered embodied agent for memory-augmented task planning in household robotics. *arXiv preprint arXiv:2504.21716*, 2025.

Guo, T., Chen, X., Wang, Y., Chang, R., Pei, S., Chawla, N. V., Wiest, O., and Zhang, X. Large language model based multi-agents: A survey of progress and challenges. *arXiv preprint arXiv:2402.01680*, 2024.

Gutiérrez, B. J., Shu, Y., Gu, Y., Yasunaga, M., and Su, Y. Hipporag: Neurobiologically inspired long-term memory for large language models. *arXiv preprint arXiv:2405.14831*, 2024.

H2O.ai. Autonomous agentic ai: execute multi-step workflows autonomously. [Online], 2024. https://h2o.ai/platform/enterprise-h2ogpte/#AgenticAI.

Hatalis, K., Christou, D., and Kondapalli, V. Review of case-based reasoning for llm agents: Theoretical foundations, architectural components, and cognitive integration. *arXiv preprint arXiv:2504.06943*, 2025.

Hong, S., Zheng, X., Chen, J., Cheng, Y., Wang, J., Zhang, C., Wang, Z., Yau, S. K. S., Lin, Z., Zhou, L., et al. Metagpt: Meta programming for multi-agent collaborative framework. *arXiv preprint arXiv:2308.00352*, 3(4): 6, 2023.

Hu, M., Chen, T., Chen, Q., Mu, Y., Shao, W., and Luo, P. Hiagent: Hierarchical working memory management for solving long-horizon agent tasks with large language model. *arXiv preprint arXiv:2408.09559*, 2024.

Hu, M., Zhou, Y., Fan, W., Nie, Y., Xia, B., Sun, T., Ye, Z., Jin, Z., Li, Y., Zhang, Z., Wang, Y., Ye, Q., Luo, P., and Li, G. Owl: Optimized workforce learning for general multi-agent assistance in real-world task automation, 2025. URL https://github.com/camel-ai/owl.

Hu, P. and Ying, X. Unified mind model: Reimagining autonomous agents in the llm era. *arXiv preprint arXiv:2503.03459*, 2025.

Huang, X., Liu, W., Chen, X., Wang, X., Wang, H., Lian, D., Wang, Y., Tang, R., and Chen, E. Understanding the planning of llm agents: A survey. *arXiv preprint arXiv:2402.02716*, 2024.

Jimenez, C. E., Yang, J., Wettig, A., Yao, S., Pei, K., Press, O., and Narasimhan, K. Swe-bench: Can language models resolve real-world github issues? *arXiv preprint arXiv:2310.06770*, 2023.

LangChain. Open deep research. [Online], 2024. https://github.com/langchain-ai/open_deep_research.

Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W., Rocktäschel, T., Riedel, S., and Kiela, D. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Proc. NeurIPS*, 2020.

Li, X., Dong, G., Jin, J., Zhang, Y., Zhou, Y., Zhu, Y., Zhang, P., and Dou, Z. Search-o1: Agentic search-enhanced large reasoning models. *arXiv preprint arXiv:2501.05366*, 2025a.

Li, X., Jin, J., Dong, G., Qian, H., Zhu, Y., Wu, Y., Wen, J.-R., and Dou, Z. Webthinker: Empowering large reasoning

models with deep research capability. *arXiv preprint arXiv:2504.21776*, 2025b.

Liu, B., Li, C., Tan, M., Liu, W., and Yang, Y. Echo: A large language model with temporal episodic memory. *arXiv preprint arXiv:2502.16090*, 2025a.

Liu, B., Li, X., Zhang, J., Wang, J., He, T., Hong, S., Liu, H., Zhang, S., Song, K., Zhu, K., et al. Advances and challenges in foundation agents: From brain-inspired intelligence to evolutionary, collaborative, and safe systems. *arXiv preprint arXiv:2504.01990*, 2025b.

Liu, H., Wang, Z., Chen, X., Li, Z., Xiong, F., Yu, Q., and Zhang, W. Hoprag: Multi-hop reasoning for logic-aware retrieval-augmented generation. *arXiv preprint arXiv:2502.12442*, 2025c.

Lu, J., An, S., Lin, M., Pergola, G., He, Y., Yin, D., Sun, X., and Wu, Y. Memochat: Tuning llms to use memos for consistent long-range open-domain conversation. *arXiv preprint arXiv:2308.08239*, 2023.

Mialon, G., Fourrier, C., Wolf, T., LeCun, Y., and Scialom, T. Gaia: a benchmark for general ai assistants. In *The Twelfth International Conference on Learning Representations*, 2023.

OpenAI. deepresearch, 2024. URL https://openai.com/index/introducing-deep-research/.

Packer, C., Wooders, S., Lin, K., Fang, V., Patil, S. G., Stoica, I., and Gonzalez, J. E. Memgpt: Towards LLMs as operating systems. *arXiv preprint arXiv:2310.08560*, 2023.

Pan, J., Wang, X., Neubig, G., Jaitly, N., Ji, H., Suhr, A., and Zhang, Y. Training software engineering agents and verifiers with swe-gym. *arXiv preprint arXiv:2412.21139*, 2024.

Peng, D. Langfun, September 2023. URL https://github.com/google/langfun. Version 0.0.1, Apache-2.0 License.

Phan, L., Gatti, A., Han, Z., Li, N., Hu, J., Zhang, H., Zhang, C. B. C., Shaaban, M., Ling, J., Shi, S., et al. Humanity's last exam. *arXiv preprint arXiv:2501.14249*, 2025.

Piao, J., Yan, Y., Zhang, J., Li, N., Yan, J., Lan, X., Lu, Z., Zheng, Z., Wang, J. Y., Zhou, D., et al. Agentsociety: Large-scale simulation of llm-driven generative agents advances understanding of human behaviors and society. *arXiv preprint arXiv:2502.08691*, 2025.

Qiao, S., Fang, R., Zhang, N., Zhu, Y., Chen, X., Deng, S., Jiang, Y., Xie, P., Huang, F., and Chen, H. Agent planning with world knowledge model. In *Proc. NeurIPS*, 2024.

Schick, T., Dwivedi-Yu, J., Dessì, R., Raileanu, R., Lomeli, M., Zettlemoyer, L., Cancedda, N., and Scialom, T. Toolformer: Language models can teach themselves to use tools. *arXiv preprint arXiv:2302.04761*, 2023.

Shah, V., Veerendranath, V., Neubig, G., Fried, D., and Wang, Z. Z. Exploring the pre-conditions for memory-learning agents. In *Scaling Self-Improving Foundation Models without Human Supervision*, 2025. URL https://openreview.net/forum?id=WZV7I3PT90.

Shinn, N., Cassano, F., Berman, E., Gopinath, A., Narasimhan, K., and Yao, S. Reflexion: Language agents with verbal reinforcement learning. *arXiv preprint arXiv:2303.11366*, 2023.

Silver, D. and Sutton, R. S. Welcome to the era of experience. *Google AI*, 2025.

Tang, J., Fan, T., and Huang, C. Autoagent: A fully-automated and zero-code framework for llm agents. *arXiv e-prints*, pp. arXiv–2502, 2025.

Trase. Meet trase systems. [Online], 2024. https://www.trasesystems.com/.

Wang, Y., Gao, Y., Chen, X., Jiang, H., Li, S., Yang, J., Yin, Q., Li, Z., Li, X., Yin, B., Shang, J., and McAuley, J. MEMORYLLM: Towards self-updatable large language models. *arXiv preprint arXiv:2402.04624*, 2024a.

Wang, Z. Z., Mao, J., Fried, D., and Neubig, G. Agent workflow memory. *arXiv preprint arXiv:2409.07429*, 2024b.

Wei, J., Sun, Z., Papay, S., McKinney, S., Han, J., Fulford, I., Chung, H. W., Passos, A. T., Fedus, W., and Glaese, A. Browsecomp: A simple yet challenging benchmark for browsing agents. *arXiv preprint arXiv:2504.12516*, 2025.

Wu, J., Yin, W., Jiang, Y., Wang, Z., Xi, Z., Fang, R., Zhang, L., He, Y., Zhou, D., Xie, P., et al. Webwalker: Benchmarking llms in web traversal. *arXiv preprint arXiv:2501.07572*, 2025.

Wu, Z., Han, C., Ding, Z., Weng, Z., Liu, Z., Yao, S., Yu, T., and Kong, L. Os-copilot: Towards generalist computer agents with self-improvement. *arXiv preprint arXiv:2402.07456*, 2024.

Xiong, W., Song, Y., Dong, Q., Zhao, B., Song, F., Wang, X., and Li, S. Mpo: Boosting llm agents with meta plan optimization. *arXiv preprint arXiv:2503.02682*, 2025.

Xu, W., Liang, Z., Mei, K., Gao, H., Tan, J., and Zhang, Y. A-mem: Agentic memory for llm agents. *arXiv preprint arXiv:2502.12110*, 2025.

Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K., and Cao, Y. ReAct: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*, 2022. URL https://arxiv.org/abs/2210.03629.

Zeng, R., Fang, J., Liu, S., and Meng, Z. On the structural memory of llm agents. *arXiv preprint arXiv:2412.15266*, 2024.

Zhang, F., Chen, B., Zhang, Y., Keung, J., Liu, J., Zan, D., Mao, Y., Lou, J.-G., and Chen, W. Repocoder: Repository-level code completion through iterative retrieval and generation. *arXiv preprint arXiv:2303.12570*, 2023.

Zhang, J., Krishna, R., Awadallah, A. H., and Wang, C. Ecoassistant: Using llm assistants more affordably and accurately. In *ICLR 2024 Workshop on Large Language Model (LLM) Agents*, 2024a.

Zhang, Z., Bo, X., Ma, C., Li, R., Chen, X., Dai, Q., Zhu, J., Dong, Z., and Wen, J.-R. A survey on the memory mechanism of large language model based agents. *arXiv preprint arXiv:2404.13501*, 2024b.

Zhao, Z., Zhang, S., Du, Y., Liang, B., Wang, B., Li, Z., Li, B., and Wong, K.-F. Eventweave: A dynamic framework for capturing core and supporting events in dialogue systems. *arXiv preprint arXiv:2503.23078*, 2025.

Zheng, L., Wang, R., Wang, X., and An, B. Synapse: Trajectory-as-exemplar prompting with memory for computer control. *arXiv preprint arXiv:2306.07863*, 2023.

Zhong, W., Guo, L., Gao, Q., Ye, H., and Wang, Y. Memorybank: Enhancing large language models with long-term memory. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 19724–19731, 2024.

Zhu, Y., Qiao, S., Ou, Y., Deng, S., Zhang, N., Lyu, S., Shen, Y., Liang, L., Gu, J., and Chen, H. Knowagent: Knowledge-augmented planning for LLM-based agents. In *Proc. NAACL Findings*, 2024.

Limitations

Despite the promising results demonstrated by AGENT KB, our approach faces inherent scalability challenges as the knowledge base grows. The current retrieval mechanism, while effective on our experimental scale, exhibits polynomial complexity growth with respect to the number of stored experiences. As the repository expands from thousands to millions of entries across diverse domains, maintaining sub-second retrieval latency becomes increasingly difficult, potentially limiting real-time applications that require immediate responses. Our experiments show that retrieval time increases by approximately 15% for every doubling of the knowledge base size, suggesting the need for more sophisticated indexing mechanisms beyond our current hierarchical structure.

The quality and reliability of automatically generated experiences represent another fundamental limitation. While our validation mechanisms filter out obvious failures, subtle errors in reasoning patterns or domain-specific nuances may propagate through the system undetected. Our analysis reveals that approximately 8% of automatically generated experiences contain minor inaccuracies that, while not immediately harmful, could compound when applied recursively. This is particularly problematic in safety-critical domains where even small errors can have significant consequences. The current system lacks mechanisms for experience deprecation or version control, meaning outdated or suboptimal strategies may persist indefinitely without systematic review.

Cross-domain knowledge transfer, while generally beneficial, shows diminishing returns when domains share minimal structural similarity. Our experiments indicate that experiences from programming tasks provide limited benefit for natural language generation tasks, with transfer effectiveness dropping below 20% for semantically distant domains. This suggests fundamental boundaries to the universality of our approach, requiring careful consideration of domain relationships when constructing the knowledge base. Additionally, our reliance on pre-trained language models for experience encoding and retrieval creates an implicit bias toward tasks well-represented in these models' training data, potentially disadvantaging novel or specialized domains.

Future Work

Advancing beyond retrieval-based knowledge reuse, we envision developing a causal reasoning framework that understands why certain strategies succeed in specific contexts. This framework would decompose experiences into causal chains, identifying prerequisite conditions, action-outcome relationships, and contextual dependencies. By modeling these causal structures explicitly, agents could synthesize novel solutions by recombining causal fragments rather than merely adapting complete experiences. Preliminary investigations suggest that causal decomposition could improve transfer effectiveness by 30-40% for cross-domain applications, particularly in scenarios requiring creative problem-solving rather than pattern matching.

The integration of continual learning mechanisms represents another crucial direction for AGENT KB's evolution. Rather than treating the knowledge base as a static repository, we propose implementing experience refinement loops that automatically update strategies based on deployment outcomes. This would involve tracking the success rates of retrieved experiences in novel contexts, identifying systematic failure patterns, and synthesizing improved versions through automated experimentation. Such a system would require careful balance between exploration of new strategies and exploitation of proven approaches, potentially leveraging multi-armed bandit algorithms or evolutionary optimization techniques to guide the refinement process.

Theoretical foundations for cross-agent knowledge transfer remain underdeveloped, presenting opportunities for fundamental research. We plan to investigate formal frameworks for characterizing experience transferability, potentially drawing from domain adaptation theory and meta-learning. Understanding the geometric properties of experience embeddings and their relationship to task similarity could enable more principled retrieval mechanisms. Furthermore, developing provable guarantees for retrieval quality and transfer effectiveness would enhance AGENT KB's applicability in high-stakes scenarios where performance bounds are critical.

Broad Impact

AGENT KB fundamentally transforms how AI systems accumulate and share knowledge, potentially accelerating the pace of AI development while reducing duplicated efforts across the research community. By enabling smaller organizations and individual researchers to leverage experiences accumulated by well-resourced institutions, our framework democratizes access to advanced problem-solving strategies. This democratization effect could be particularly transformative in developing countries and underfunded research areas, where limited computational resources currently constrain AI advancement. However, this concentration of knowledge also raises questions about intellectual property and competitive advantage, requiring careful consideration of contribution attribution and usage rights.

The transparency and interpretability afforded by AGENT KB's experience-based reasoning addresses growing concerns about AI accountability in critical applications. Unlike black-box neural systems, agents using AGENT KB can justify decisions by citing specific past experiences and the reasoning patterns derived from them. This traceability becomes invaluable in regulated industries such as healthcare and finance, where decision audit trails are legally mandated. Nevertheless, the system's reliance on historical experiences may inadvertently perpetuate past biases or outdated practices, particularly if the knowledge base lacks diversity in contributors or problem domains.

The societal implications of widespread AGENT KB adoption extend beyond technical considerations. In educational settings, students could access expert problem-solving strategies previously available only through direct mentorship, potentially revolutionizing how complex skills are taught and learned. In professional contexts, AGENT KB could serve as an intelligent assistant that captures and propagates organizational knowledge, preventing expertise loss due to employee turnover. However, this same capability raises concerns about job displacement and the commoditization of expert knowledge. Ensuring that AGENT KB enhances rather than replaces human expertise requires thoughtful deployment strategies and ongoing dialogue between technologists, domain experts, and affected communities.

A. Experience Representation and Storage

While the main paper focuses on our three key innovations (knowledge abstraction, dual-phase retrieval, and adaptive refinement), this appendix provides additional technical details on how experiences are represented and stored within AGENT KB.

A.1. Experience Representation

Each experience in AGENT KB is encoded as a structured tuple $E = \langle \pi, \gamma, S, \mathcal{C}, \mu, \mathcal{F}, \mathcal{R} \rangle$, where:

- π represents the problem pattern, including task type, input structure, and constraints
- γ denotes the goal or objective, including success criteria and expected outputs
- $S = \{s_1, s_2, \ldots, s_n\}$ is a workflow capturing a sequence of reasoning and execution steps
- $\mathcal{C}$ captures contextual features including domain D and difficulty level δ
- μ contains metadata such as success indicator, efficiency metrics, and generalizability scores
- $\mathcal{F}$ encodes failure modes and recovery patterns when applicable
- $\mathcal{R}$ represents relations to other experiences, including prerequisites and alternatives

This comprehensive representation enables AGENT KB to capture not only what worked but also contextual factors that influence success and alternative approaches that might be relevant in different scenarios.

Experience Representation and Organization Before detailing the retrieval process, we define how experiences are represented within AGENT KB. Each experience E is encoded with multi-faceted embeddings: $f(E) = \{f^{\pi}, f^{\gamma}, f^{S}, f^{\mathcal{C}}\}$, where f^{π} represents the problem pattern embedding, f^{γ} the goal embedding, f^{S} the solution steps embedding, and $f^{\mathcal{C}}$ the context embedding.

Experiences are organized in a hierarchical knowledge graph $\mathcal{KB} = (V, \mathcal{E})$ where vertices V are experiences and edges $\mathcal{E}$ represent relationships such as abstraction, composition, and adaptation. This structure enables efficient navigation across related experiences.

Student Agent: Query-based Workflow Retrieval When a query Q (e.g., a GAIA benchmark problem) is received, the student agent initiates the first retrieval phase. The student agent first *reasons* about how to approach the problem, identifying key requirements and potential solution strategies. Then, it performs *retrieval* from AGENT KB to find relevant experiences that might guide its planning process. Given the current agent state $\hat{S}$ with problem $\hat{\pi}$ and goal $\hat{\gamma}$, the student retrieves relevant experiences through: $\text{Retrieve}(\hat{S}, \hat{\pi}, \hat{\gamma}, k) = \arg\text{top}_k\Big(\text{sim}(E_i, \hat{S}) \cdot \text{Relevance}(E_i, \hat{\pi}, \hat{\gamma}) \cdot \text{Success}(E_i)\Big)$.

This query-based retrieval process operates through a sophisticated multi-stage approach that balances broad similarity matching with precise state alignment. First, we perform coarse retrieval based on problem-goal similarity, identifying experiences where $\mathcal{E}_{\text{coarse}} = \{E_i | \text{sim}_{\cos}(r_i, \hat{r}) > \theta_{\text{coarse}}\}$, where $r_i = f(\pi_i) + f(\gamma_i)$ and $\hat{r} = f(\hat{\pi}) + f(\hat{\gamma})$. This is complemented by fine-grained retrieval that uses the current agent state to find experiences with matching execution steps, where $\mathcal{E}_{\text{fine}} = \arg\text{top}_k(S_i^{\text{fine}})$ with $S_i^{\text{fine}} = \sum_{j=1}^{m} \max_{\ell=1,\ldots,L_i} \text{sim}_{\cos}(s_{i,\ell}, \hat{s}_j)$. Finally, these retrieval strategies are combined through an adaptive mechanism: $S_i(t) = \lambda(t) \cdot S_i^{\text{coarse}} + (1 - \lambda(t)) \cdot S_i^{\text{fine}}$, where $S_i^{\text{coarse}} = \text{sim}_{\cos}(r_i, \hat{r})$ and $\lambda(t) \in [0, 1]$ is a time-dependent weighting function that balances coarse and fine-grained retrieval based on the current stage of problem solving. This creates a context-sensitive retrieval approach that evolves throughout the problem-solving process, with final selection given by $\mathcal{R}(t) = \arg\text{top}_k S_i(t)$.

The retrieved experiences contain successful workflows from similar historical tasks, including critical elements such as complete planning structures (step sequences), appropriate tool selection for each step, and general reasoning patterns relevant to the query type. The student agent's primary focus at this stage is ensuring the overall workflow structure is appropriate for the task.

The student agent then adapts these experiences to the current context, applying operations such as parameter substitution, step expansion/contraction, and domain translation: $E_{\text{adapted}} = \text{Adapt}(E_{\text{retrieved}}, \hat{S}, \hat{\pi}, \hat{\gamma})$. These adapted experiences are synthesized to generate an initial execution plan: $\text{Plan}_{\text{initial}} = \text{Integrate}(\text{Plan}_{\text{empty}}, \{E_{\text{adapted}}\})$.

This plan includes a sequence of reasoning steps $S = \{s_1, s_2, ..., s_n\}$, each with specified tools and execution parameters. The student agent executes this plan, generating execution logs L that capture both successes and failures during the process.

Teacher Agent: Log-based Reasoning and Refinement After the initial execution, the student agent forwards both the query Q and execution logs L to the teacher agent. Unlike the student agent, which focuses on planning the overall workflow, the teacher agent performs critical reasoning functions on the execution itself. The teacher agent analyzes the logs through three main processes:

First, it performs error analysis to identify problematic steps: $\text{ErrorAnalysis}(L) = \{(s_i, \text{error}_i, \text{cause}_i) \mid s_i \in S, \text{HasError}(s_i) = \text{True}\}$. Next, it summarizes the execution log to extract key patterns: $\text{LogSummarization}(L) = \text{Summarize}(\{s_1, s_2, ..., s_n\})$. Finally, it evaluates the overall performance by comparing actual outcomes with expected results: $\text{PerformanceEvaluation}(L, Q) = \text{Evaluate}(\text{Outcome}(L), \text{ExpectedOutcome}(Q))$.

Based on this comprehensive analysis, the teacher agent identifies problematic steps that require refinement: $\text{ProblematicSteps} = \text{IdentifyIssues}(\text{ErrorAnalysis}(L), \text{PerformanceEvaluation}(L, Q))$.

For each problematic step, the teacher agent performs a targeted secondary retrieval from AGENT KB, focusing on fine-grained matching of step-level experiences: $E_{\text{refinement}} = \arg\text{top}_m \Big(\sum_{s_i \in \text{ProblematicSteps}} \max_l \text{sim}_{\cos}(s_i, E_j.S_l) \cdot \text{Precision}(E_j) \Big)$.

Unlike the first retrieval phase which focused on overall workflow structure, this log-based refinement retrieval targets specific execution details that affect precision and correctness. The teacher agent identifies granular aspects such as precise parameter configurations (e.g., maintaining three decimal places in calculations), error handling strategies for specific failure modes, tool usage refinements and constraints, and step-specific reasoning patterns that improve accuracy. These fine-grained execution details are critical for successfully completing tasks that require not just the right approach but also precise implementation.

The teacher agent then adapts these refinement experiences: $E_{\text{refined}} = \text{Adapt}(E_{\text{refinement}}, L, Q)$.

And generates specific refinement hints by reasoning over the adapted experiences: $\text{Hints} = \text{GenerateRefinements}(E_{\text{refined}}, \text{ProblematicSteps}, L, Q)$.

Benefits of the Dual-Phase Approach This two-phase approach significantly enhances performance by addressing both structural correctness and execution precision. The Query-based Retrieval ensures the overall workflow structure is appropriate for the task (correct sequence of steps and tool selection), while the Log-based Refinement focuses on execution details that impact success (precise calculations, error handling, parameter tuning).

Through this teacher-student collaboration, AGENT KB enables progressive refinement that mimics human expert-apprentice learning relationships. Both agents employ the `Reason-Retrieve-Refine` pipeline, but with different focuses: the student agent reasons about the problem structure and overall solution approach, while the teacher agent reasons about the execution quality and potential improvements. The teacher agent effectively transfers knowledge from past experiences to guide the student agent toward successful task completion, with each phase targeting a different aspect of performance improvement.

A.2. Vector Embedding Mechanisms

To support efficient retrieval, each experience is encoded with multi-faceted embeddings: $f(E) = \{f^{\pi}, f^{\gamma}, f^{S}, f^{\mathcal{C}}\}$, where:

- f^{π} represents the problem pattern embedding
- f^{γ} denotes the goal embedding
- f^{S} captures the solution steps embedding
- $f^{\mathcal{C}}$ encodes the context embedding

These embeddings are generated using specialized encoding models that are tailored to each aspect of the experience. Problem and goal embeddings prioritize semantic understanding, while step embeddings prioritize sequential patterns and tool usage. Context embeddings capture domain-specific features that influence solution strategies.

A.3. Storage and Indexing

Experiences are organized in a hierarchical knowledge graph $\mathcal{KB} = (V, \mathcal{E})$ where vertices V represent individual experiences and edges $\mathcal{E}$ denote meaningful relationships between them. These relationships include:

- **Abstraction**: connecting concrete experiences to their abstracted versions
- **Composition**: linking sub-workflows to composite workflows
- **Adaptation**: connecting experiences that have been successfully adapted across domains
- **Alternative**: connecting different approaches to solving similar problems

This graph structure facilitates efficient navigation across related experiences, enabling both breadth-first exploration of alternatives and depth-first exploration of hierarchical solution approaches.

To enable efficient retrieval over this structured repository, we employ a multi-indexing strategy. Specifically, two primary indexes form the basis of the retrieval mechanism:

- **Semantic index**: Encodes the semantic meaning of problems and goals to enable intent-driven retrieval, identifying experiences addressing conceptually similar tasks.
- **Structural index**: Captures workflow structure patterns to support retrieval based on similarities in process organization or control flow.

Together, these indexes underpin a dual-phase retrieval approach that efficiently identifies relevant experiences at both the workflow and component level, avoiding exhaustive traversal of the entire knowledge graph.

B. Detailed Retrieval Mechanisms

This appendix provides additional technical details on the retrieval mechanisms used in AGENT KB, focusing on the algorithms and scoring functions that drive the dual-phase retrieval process.

B.1. Coarse-Grained Workflow Retrieval

The student agent's workflow retrieval process combines multiple similarity metrics to identify relevant experiences. The primary retrieval function is:

$$\mathcal{E}_{\text{workflow}} = \arg\text{top}_k\Big(S_{\text{workflow}}(E_i, Q)\Big) \tag{1}$$

Where the workflow similarity score S_{workflow} is calculated as:

$$S_{\text{workflow}}(E_i, Q) = w_\pi \cdot \text{sim}_\pi(E_i.\pi, Q.\pi) + w_\gamma \cdot \text{sim}_\gamma(E_i.\gamma, Q.\gamma) + w_C \cdot \text{sim}_C(E_i.C, Q.C) \tag{2}$$

The similarity functions use cosine similarity between the corresponding embedding vectors:

$$\text{sim}_\pi(E_i.\pi, Q.\pi) = \frac{f^\pi(E_i.\pi) \cdot f^\pi(Q.\pi)}{||f^\pi(E_i.\pi)|| \cdot ||f^\pi(Q.\pi)||} \tag{3}$$

To ensure retrieval of experiences that can be effectively adapted, we incorporate a transferability score:

$$\text{trans}(E_i, Q) = \exp\left(-\frac{d_{\text{domain}}(E_i.D, Q.D)}{\tau}\right) \tag{4}$$

Where d_{domain} measures domain distance and τ is a temperature parameter that controls the sensitivity to domain differences. The final workflow retrieval score combines similarity and transferability:

$$S_{\text{final}}(E_i, Q) = S_{\text{workflow}}(E_i, Q) \cdot \text{trans}(E_i, Q) \cdot E_i.\mu.\text{success} \tag{5}$$

This approach ensures that retrieved workflows are not only similar to the current task but also likely to transfer successfully across domain boundaries.

B.2. Fine-Grained Step Retrieval

The teacher agent's step retrieval process focuses on identifying specific execution steps that address observed issues. For each problematic step s_p identified in the execution logs, the retrieval function is:

$$\mathcal{E}_{\text{step}}(s_p) = \arg\text{top}_m\Big(S_{\text{step}}(E_i, s_p)\Big) \tag{6}$$

Where the step similarity score S_{step} is calculated as:

$$S_{\text{step}}(E_i, s_p) = \max_{s_j \in E_i.S}\Big(\text{sim}_{step}(s_j, s_p) \cdot \text{issue\_match}(s_j, s_p)\Big) \tag{7}$$

The step similarity function compares both the functional purpose and the execution details:

$$\text{sim}_{step}(s_j, s_p) = w_{\text{func}} \cdot \text{sim}_{\text{func}}(s_j, s_p) + w_{\text{exec}} \cdot \text{sim}_{\text{exec}}(s_j, s_p) \tag{8}$$

The issue matching function assesses how well the retrieved step addresses the specific issue observed:

$$\text{issue\_match}(s_j, s_p) = \text{sim}(s_j.\text{issue\_type}, s_p.\text{issue\_type}) \cdot s_j.\text{resolution\_effectiveness} \tag{9}$$

By combining these scoring functions, the teacher agent can identify steps that specifically address the execution issues encountered by the student agent, enabling precise refinement of problematic steps without disrupting the overall workflow structure.

C. Experimental Details

C.1. Experimental Cost

All services used in this work rely on third-party API calls to OpenAI's language models (GPT-4.1, Claude-3-7-sonnet, o1, etc). The total cost of execution is primarily determined by the number of tokens processed during both prompt input and model output generation. Specifically, we report the token cost associated with different modules of our AGENT KB (Knowledge Base) system, as well as the per-agent token consumption during task execution.

As summarized in Table 3, the token cost of GPT-4.1 varies significantly depending on the complexity of the agent and its interaction with the knowledge base. For instance, the Action Agent requires a relatively high number of reasoning steps (up to 12), resulting in a higher cumulative token count across multiple interactions. In contrast, the Student Agent and Teacher Agent , while still utilizing LLM-based inference, operate in a more passive or structured manner, leading to fewer dynamic interactions and correspondingly lower token usage. The Database Generation module incurs a one-time cost during initialization, where large volumes of domain-specific knowledge are encoded into structured prompts for retrieval-augmented generation.

Given that OpenAI pricing is typically calculated based on both input and output tokens, the total cost of our experiments remains moderate due to suitable prompt engineering and step-limited execution strategies.

Table 3. Analysis of computational costs on the GAIA benchmark for AGENT KB. All costs, excluding database generation, correspond to a single evaluation on the GAIA validation set (165 tasks).

Type	Module	Prompt Tokens	Completion Tokens	Cost	Max Steps
Action agent	Action	~34M	~7M	~$84.32	12
Database Generation	AGENT KB	~5M	~750K	~$10.88	-
Log summary	AGENT KB	~1M	~10K	~$1.41	-
Student agent	AGENT KB	~35K	~15K	~$0.13	-
Teacher agent	AGENT KB	~45K	~15K	~$0.14	-

Token prices: $1.36/M prompt token, $5.44/M completion token.

As shown in Table 4, the computational costs of SWE-bench evaluation under the AGENT KB framework vary based on the source and structure of the hint material. Reasoning modules using RepoClassBench incur higher token costs due to deeper reasoning chains and longer hint contexts. In contrast, lightweight configurations such as Top-n SWE-Gym with shorter hints and fewer reasoning steps significantly reduce per-item cost. By tailoring the prompt size and controlling the number of refinement steps, we maintain a low average cost (under $0.008 per instance), ensuring the framework is scalable for large-scale software engineering benchmarks.

Table 4. Analysis of computational costs on the SWE-bench benchmark for AGENT KB modules. All costs correspond to per-item inference using GPT-4.1

Hint Source	**Module**	**Prompt Tokens**	**Completion Tokens**	**Cost** (/item)	**Hint Length** (tokens/item)	**Max Steps**
RepoClassBench	Reasoning	~6.5K	~850	~$0.007805	~90	100
RepoClassBench	Refine	~4.2K	~450	~$0.0028	~130	100
Top-n SWE-Gym	Retrieval+Refine	~2.8K	~300	~$0.001875	~60	100
Top-n RepoClassBench	Retrieval+Refine	~3.1K	~350	~$0.002125	~70	100

Token prices: $1.36/M prompt tokens, $5.44/M completion tokens.

C.2. Ablation Details of `Reason`-`Retrieve`-`Refine` Modules

To evaluate the effectiveness of each component in our AGENT KB framework, we conduct a series of ablation studies. Our system consists of two agents: Student Agent and Teacher Agent, with distinct roles across two reasoning stages.

- **Student Agent** is responsible for the initial stage, which begins with `Reason` (to summarize key features from the

input), followed by `Retrieve` (to find relevant prior experiences), and concludes with `Refine` (to improve the suggestions based on retrieved information).

- **Teacher Agent** operates in the second stage, where it begins with `Reason` (to analyze the logs and identify key errors), followed by `Retrieve` (to gather relevant experience), and concludes with `Refine` (to improve or correct the suggestions based on the retrieved information).

The experimental setup involves systematically removing or disabling specific modules or agents to assess their individual contributions.

- w/o Student Agent: The first-stage steps are removed.
- w/o Teacher Agent: The second-stage steps are removed.
- w/o `Reason` Module : In both stages, no reasoning is performed; only retrieval based on raw data is conducted.
- w/o `Retrieve` Module : Both stages omit the retrieval process entirely. Agents rely solely on prompt-based instructions to generate responses, without consulting prior experiences.
- w/o `Refine` Module : n both stages, no refinement is performed; only the retrieved content is used as knowledge.
- w/ Raw Workflow : The full pipeline is used, but without any explicit modular control—i.e., the model follows a standard prompting strategy throughout, lacking structured guidance through the Reason, Retrieve, and Refine phases.

These ablation experiments provide insight into how each module contributes to overall performance, particularly in terms of accuracy, robustness, and coherence in complex reasoning tasks.

C.3. GAIA Details

Evaluated on the validation set of GAIA across three difficulty levels:

- **Level 1 (53 tasks)**: Basic tasks requiring simple reasoning or straightforward retrieval.
- **Level 2 (86 tasks)**: Intermediate complexity with multi-step reasoning or tool usage.
- **Level 3 (26 tasks)**: Advanced tasks demanding sophisticated reasoning and domain knowledge.

Performance is measured using an unweighted average over all 165 tasks.

Two metrics are used:

- `Pass@1`: Evaluates correctness of the first generated solution.
- `Pass@3`: Evaluates whether any of the three independently generated solutions is correct.

Method Configurations:

- +AGENT KB / +AGENT KB ✓ : Evaluated using `Pass@1`, representing the model's initial attempt or after one round of feedback.
- +AGENT KB ✓♡ : Uses `Pass@3` to align with standard practices and improve comparability with existing methods.

C.4. SWE-bench Details.

Performance is measured using an unweighted average over all 300 tasks.

Two metrics are used:

- `Pass@1`: Evaluates correctness of the first generated solution.
- `Pass@3`: Evaluates whether any of the three independently generated solutions is correct.

Model Configurations:

- +AGENT KB / +AGENT KB ✓ : Evaluated using Pass@1, representing the model's initial attempt or after one round of feedback.
- +AGENT KB ✓♡ : Uses Pass@3 to align with standard practices and improve comparability with existing methods.

D. Additional Details of Methodology

D.1. Experience Quality Update

After the complete execution cycle, we update the quality metrics of the utilized experiences based on their contribution to the outcome: $Q_{\text{new}}(E) = (1-\alpha) \cdot Q_{\text{old}}(E) + \alpha \cdot \text{ExecOutcome}(E, \hat{S})$, with $\alpha \in [0, 1]$ as a learning rate and ExecOutcome measuring success in the current context. This quality update ensures that more effective experiences are prioritized in future retrievals.

D.2. Experience Integration and Conflict Resolution

The teacher agent returns these refinement hints to the student agent, which must integrate them with the initial plan. This integration process requires resolving potential conflicts: $\text{Plan}_{\text{refined}} = \text{Integrate}(\text{Plan}_{\text{initial}}, \{\text{Hints}\})$, with conflict resolution following:

$$\text{Conflict}(p_1, p_2) = \begin{cases} \text{Merge}(p_1, p_2) & \text{if Compatible}(p_1, p_2) > \theta_c \\ \text{Select}(p_1, p_2) & \text{otherwise} \end{cases}$$

.

The student agent then executes this refined plan, typically achieving superior performance compared to the initial execution.

D.3. Knowledge Evolution

AGENT KB continuously evolves through collaborative experience refinement: $E_{\text{refined}} = \text{Refine}(E, \mathcal{U})$, where $\mathcal{U}$ is the usage history containing information about when and how the experience has been used. Similar experiences from different agents are merged:

$$E_{\text{merged}} = \text{Merge}(E_i, E_j) = \langle \pi_{ij}, \gamma_{ij}, S_{ij}, \mathcal{C}_{ij}, \mu_{ij}, \mathcal{F}_{ij}, \mathcal{R}_{ij} \rangle,$$

while outdated or low-value experiences are pruned:

$$\text{Prune}(\mathcal{KB}) = \{E \in \mathcal{KB} | \text{Utility}(E, t_{\text{current}}) > \theta_p\}$$

, with utility decaying over time unless reinforced:

$$\text{Utility}(E, t) = Q(E) \cdot e^{-\lambda(t - t_{\text{recent}})} + \sum_{i=1}^{n} \text{UsageImpact}(E, t_i),$$

The complete `Reason`-`Retrieve`-`Refine` pipeline operates within both the student and teacher agents, though with different objectives and contexts:

$$\text{RRR}(\hat{S}, \hat{\pi}, \hat{\gamma}) = \text{Refine}(\text{Retrieve}(\hat{S}, \hat{\pi}, \hat{\gamma}), \hat{S}),$$

and the knowledge base evolves according to:

$$\mathcal{KB}_{t+1} = \text{Update}(\mathcal{KB}_t, \{\text{Reason}(W_i)\}_{i=1}^{N_W}, \{\text{Feedback}(E_j)\}_{j=1}^{N_E}).$$

The framework-agnostic design allows different agents to both contribute to and benefit from the shared knowledge base, creating a virtuous cycle of collective intelligence improvement that enhances multi-agent system performance over time.

E. Retrieval Details

E.1. Retrieval Architecture

AGENT KB employs a two-stage retrieval framework designed to progressively refine the selection of relevant past experiences for effective task planning and execution:

Summary-based Retrieval. The second retrieval phase conducts a fine-grained analysis of execution logs (e.g., intermediate_steps) associated with the retrieved experiences. Specifically, we summarize both the overall plan structure and individual reasoning or action steps from these logs. These summaries are then used to perform a more detailed retrieval, aligning the current task state with specific subroutines or decision points from past executions. This step facilitates the identification of effective low-level actions or reasoning patterns that are contextually aligned with the current execution trajectory.

Criticism-Based Retrieval. The system actively searches for past experiences based on shared error patterns rather than task goals or outcomes. This stage focuses on identifying historical execution logs that contain similar types of mistakes—such as flawed reasoning steps, incorrect actions, or strategic misjudgments—as the current task. By encoding and matching these failure modes semantically, the retrieval process surfaces relevant cases where similar problems arose, allowing the planner to learn from prior failures and avoid repeating them. This error-driven approach enables a more proactive and reflective planning process grounded in lessons from past critiques.

E.2. Retrieval Types.

To ensure robust and contextually relevant experience retrieval, we incorporate multiple retrieval mechanisms that operate at different levels of abstraction. Within this framework, we utilize three primary types of retrieval: Text similarity retrieval, semantic retrieval, and hybrid retrieval, each offering distinct advantages in capturing relevance between the current task and historical experiences.

Text similarity retrieval. Text similarity retrieval is based on surface-level term matching and relies on traditional information retrieval techniques such as TF-IDF (Term Frequency-Inverse Document Frequency). This method quantifies the importance of terms within a document relative to a corpus and represents textual content as sparse, high-dimensional vectors. It excels at identifying documents that share significant keyword overlap with the query, making it particularly effective when vocabulary alignment is strong.

Semantic Retrieval. Semantic retrieval goes beyond keyword matching by encoding text into dense vector representations that capture meaning and contextual relationships. In our implementation, we use the `sentence-transformers/all-MiniLM-L6-v2` model, a lightweight yet powerful transformer-based encoder that maps sentences and paragraphs into a continuous vector space. This allows for the computation of cosine similarity between embeddings, enabling the system to retrieve experiences that are semantically related—even if they do not share exact text similarity overlap.

Hybrid Retrieval. To combine the strengths of both text similarity and semantic approaches, we also implement hybrid retrieval, which fuses results from both retrieval methods using a weighted ranking strategy. For instance, the final relevance score of a retrieved experience can be computed as a linear combination of its text similarity and semantic similarity scores:

$$\text{Hybrid Score} = \alpha \cdot \text{Similarity Score} + (1 - \alpha) \cdot \text{ Semantic Score}$$

where α is a tunable parameter (default: 0.5) balancing the influence of each retrieval modality. Hybrid retrieval offers a balanced trade-off between precision and generalization, mitigating the limitations of individual methods. It ensures that the retrieval mechanism remains robust to both syntactic variation and conceptual drift while maintaining interpretability and performance.

F. Additional Experiment

F.1. Additional Evaluations

This section provides comprehensive results for the experiments conducted in main text. We present detailed performance metrics across different models and retrieval strategies on the GAIA and SWE-bench, as well as ablation studies to analyze the effectiveness of our proposed components.

Table 5 presents the detailed performance of various large language models, including GPT-4o, GPT-4.1, o3-mini, Claude-3.7, Qwen-3 32B, and DeepSeek-R1, under different experimental settings. The evaluation includes baseline performance and improvements achieved by incorporating the +AGENT KB , +AGENT KB ✓ , and +AGENT KB ✓♡ methods. Performance is measured using average accuracy and per-level accuracy on GAIA validation set, along with SWE-bench resolved scores. The final row ("Gap") indicates the improvement from the baseline to the best-performing method for each model. Notably, all models show significant gains when using the enhanced reasoning and retrieval capabilities introduced by our framework.

Table 5. Detailed results of various base models on GAIA.

Model	Method	GAIA				SWE-bench
		Average	Level 1	Level 2	Level 3	Resolved
GPT-4o	Baseline	45.06	62.26	45.35	15.38	16.33
	+AGENT KB	46.67	66.04	44.19	15.38	20.33
	+AGENT KB ✓	55.15	71.70	48.84	42.31	29.33
	+AGENT KB ✓♡	**58.79**	**77.36**	**52.33**	**42.31**	**31.33**
	Gap	Δ 13.73	Δ 15.10	Δ 6.98	Δ **26.93**	Δ **15.00**
GPT-4.1	Baseline	55.15	67.92	53.49	34.62	24.33
	+AGENT KB	61.21	79.25	58.14	34.62	28.33
	+AGENT KB ✓	67.27	83.02	67.44	34.62	37.33
	+AGENT KB ✓♡	**73.94**	**84.91**	**73.26**	**53.85**	**38.00**
	Gap	Δ **18.79**	Δ 16.99	Δ **19.77**	Δ 19.23	Δ **13.67**
o3-mini	Baseline	32.12	47.17	26.74	19.23	23.00
	+AGENT KB	29.09	39.62	25.58	19.23	31.67
	+AGENT KB ✓	33.33	45.28	30.23	19.23	35.33
	+AGENT KB ✓♡	**40.60**	**52.83**	**38.37**	**23.08**	**37.00**
	Gap	Δ 8.48	Δ 5.66	Δ 11.63	Δ 3.85	Δ 14.00
Claude-3.7	Baseline	58.79	64.15	61.63	38.46	30.00
	+AGENT KB	65.45	75.47	66.28	38.46	46.67
	+AGENT KB ✓	69.70	79.25	69.77	50.00	49.67
	+AGENT KB ✓♡	**75.15**	**84.91**	**74.42**	**57.69**	**51.00**
	Gap	Δ 16.36	Δ 20.76	Δ 12.79	Δ 19.23	Δ 9.67
Qwen-3 32B	Baseline	35.76	47.17	38.37	3.85	18.33
	+AGENT KB	41.82	64.15	33.72	23.08	20.67
	+AGENT KB ✓	46.67	71.70	37.21	26.92	28.67
	+AGENT KB ✓♡	**49.70**	**75.47**	**40.70**	**26.92**	**30.33**
	Gap	Δ 13.94	Δ **38.30**	Δ 2.33	Δ 23.07	Δ 12.00
DeepSeek-R1	Baseline	49.70	62.26	50.00	23.08	24.33
	+AGENT KB	50.91	69.81	50.00	15.38	26.67
	+AGENT KB ✓	58.18	73.58	56.98	30.77	31.00
	+AGENT KB ✓♡	**63.64**	**79.25**	**61.63**	**38.46**	**32.67**
	Gap	Δ 13.94	Δ 16.99	Δ 11.63	Δ 15.38	Δ 8.34

F.2. Retrieval Analysis

Table 6 compares summary-based and criticism-based retrieval methods across text similarity, semantic similarity, and hybrid strategies on GAIA and SWE-bench. Three key patterns emerge: (1) Hybrid retrieval achieves peak performance for summary-based methods (67.27 average on GAIA), while criticism-based methods perform best with text similarity (66.06 average). (2) Task complexity inversely correlates with performance across all methods, with Level 3 GAIA scores declining to 34.62-38.46% versus 73.58-83.02% for Level 1. (3) SWE-bench results show narrower margins between methods (4% resolved scores), suggesting benchmark-specific sensitivity to retrieval approaches.

The ablation study in Table 7 reveals three parameterization insights: (1) Optimal top-k values differ by method - text similarity peaks at k=3 (64.24 GAIA average), semantic similarity at k=5 (62.42), and hybrid search at k=3 (67.27). (2) Level 3 performance shows counterintuitive trends, with text similarity declining 7.7% from k=1 to k=5 while hybrid search improves 11.5%. (3) Parameter sensitivity varies substantially, with hybrid retrieval showing minimal $k = 1$ to $k = 5$ variance versus text similarity's 3.4% drop.

Cross-analysis identifies two critical interactions: (1) Summary-based hybrid retrieval with k=3 configuration achieves maximum GAIA performance (83.02% Level 1, 67.44% Level 2). (2) Criticism-based text similarity with k=1 yields best Level 3 results (38.46%), outperforming all hybrid configurations. These findings demonstrate that optimal retrieval configurations depend on both content type (summary vs. criticism) and task complexity, necessitating adaptive strategy selection rather than universal solutions.

Table 6. Retrieval results by different retrieval types on GAIA and SWE-bench.

Retrieval	Type	GAIA				SWE-bench
		Average	Level 1	Level 2	Level 3	Resolved
Summary-based	Text Similarity	64.24	77.36	65.11	34.62	36.00
	Semantic similarity	58.79	69.81	59.30	34.62	34.33
	Hybrid search	67.27	83.02	67.44	34.62	37.33
Criticism-based	Text similarity	66.06	77.36	67.44	38.46	32.33
	Semantic similarity	62.42	73.58	63.95	34.62	33.33
	Hybrid search	63.03	77.36	62.79	34.62	34.67

Table 7. Retrieval performance across different top-k on GAIA and SWE-bench.

Retrieval Type	Top-k	GAIA				SWE-bench
		Average	Level 1	Level 2	Level 3	Resolved
Text sim.	$k = 1$	63.03	75.47	62.79	38.46	34.67
	$k = 3$	64.24	77.36	65.11	34.62	36.00
	$k = 5$	62.42	77.36	62.79	30.77	34.33
Semantic sim.	$k = 1$	60.00	73.58	58.13	38.46	31.00
	$k = 3$	58.79	69.81	59.30	34.62	34.33
	$k = 5$	62.42	75.47	61.63	38.46	33.33
Hybrid.	$k = 1$	63.64	79.25	62.79	34.62	34.00
	$k = 3$	67.27	83.02	67.44	34.62	37.33
	$k = 5$	66.67	81.13	66.28	38.46	35.33

F.3. Knowledge Source Comparison

We also investigate the impact of different knowledge sources on AGENT KB performance. Table 8 compares performance using knowledge derived from different sources: Hand (manually crafted knowledge entries created by domain experts) and Generate (automatically generated knowledge entries derived from agent interactions). Additionally, we compare our method against SOTA (state-of-the-art results achieved by current closed-source agent frameworks on GAIA) and Open Source (state-of-the-art results achieved by current open-source agent frameworks on GAIA). Interestingly, we find that automatically generated knowledge ("Generate") performs comparably to manually crafted knowledge ("Hand") across

Table 8. Performance comparison across different experience types on GAIA and SWE-bench.

Experience type	GAIA				SWE-bench
	Average	Level 1	Level 2	Level 3	Resolved
Hand	76.97	84.91	79.07	53.85	44.00
Generate	75.15	84.91	74.42	57.69	51.00
SOTA	78.79	88.68	79.07	57.69	55.00
Open Source	72.73	86.79	73.26	42.31	47.00

most metrics. This suggests that our knowledge acquisition pipeline effectively captures and structures agent experiences, demonstrating that the automated generation of knowledge can ultimately achieve performance comparable to that of manually curated knowledge.