SelfEvolveAgent: Continual Learning in AI Agents using Memory

Hiva Mohammadzadeh (Zaad)
Stanford University
hiva@stanford.edu

1 Problem Statement

Current large language model (LLM) agents demonstrate remarkable capabilities in complex reasoning and multi-step tasks across domains such as web browsing and software engineering. However, these agents suffer from a critical limitation: they cannot learn from their experiences. This constraint manifests in several ways: agents repeatedly make past errors, discard valuable insights after task completion, fail to transfer successful strategies across similar tasks, and cannot develop increasingly sophisticated reasoning capabilities over time. This inability to learn from experience represents a fundamental bottleneck in developing truly autonomous systems with cumulative intelligence.

While ReasoningBank introduces a breakthrough memory framework that distills generalizable reasoning strategies from an agent's self-judged successful and failed experiences, significant gaps remain in the current landscape. These include: (1) the lack of comprehensive, open-source implementations that limit research adoption and reproducibility, (2) memory storage systems that do not scale effectively with prolonged use as agents accumulate thousands of experiences, and (3) single-domain operation that fails to leverage cross-domain transfer of reasoning strategies.

2 Research Approach

This project addresses these limitations through a three-pronged approach:

Baseline Implementation: We implement a ReAct (Reasoning + Acting) framework as our "No Memory" baseline. This baseline agent is a simple Think + Act + Observe loop that performs reasoning and action without any experience retention, providing a control for measuring the impact of memory enhancement. The agent utilizes BrowserGym for environment interactions and employs "accessibility tree" observations for state representation.

ReasoningBank Implementation: We implement the ReasoningBank framework with the following core components: (1) a memory storage system that distills reasoning strategies from agent experiences, structured as JSON objects containing titles, descriptions, and detailed content arrays, (2) a retrieval system using gemini-embedding-001 for embedding generation and FAISS vector database for context-aware memory access during task execution, and (3) a self-evolution mechanism that enables continuous learning from success and failure patterns.

Evaluation Infrastructure: We establish comprehensive evaluation pipelines across three primary benchmarks: WebArena (web navigation tasks), Mind2Web (web interaction understanding), and SWE-Bench-Verified (software engineering tasks). Our compute infrastructure includes API credits for Google Gemini and Anthropic Claude models, AWS credits for WebArena hosting, and BrowserGym integration for agent actions.

3 Intermediate Results

3.1 Evaluation Setup

We evaluate on WebArena benchmarks using Gemini-2.5-Flash, Gemini-2.5-Pro and Claude-3.7-sonnet models. Our baseline is a standard ReAct agent without memory. The ReasoningBank's memory system uses JSON-structured reasoning strategies with gemini-embedding-001 embeddings

and FAISS for retrieval. We measure success rate and steps to success across Multi (29 samples), Reddit (106 samples), and additional WebArena subdomains (GitLab, Shopping, Admin), Mind2Web and SWE-Bench-Verified evaluations in progress. Infrastructure includes AWS-hosted WebArena environments and BrowserGym for browser automation.

3.2 Preliminary Results

Table 1 and 2 present our current evaluation results on two WebArena subsets. Our ReasoningBank implementation demonstrates strong performance, achieving a 6.9 percentage-point improvement on Multi (from 10.34% $\rightarrow$ 17.24%) and a 1.9 percentage-point improvement on Reddit (from 68.87% $\rightarrow$ 70.75%) under Gemini-2.5-Flash. With Gemini-2.5-Pro, ReasoningBank achieves a 6.9 percentage-point improvement on Multi (from 13.79% $\rightarrow$ 20.69%) and a 4.7 percentage-point improvement on Reddit (from 64.15% $\rightarrow$ 68.87%). The memory-enhanced agent also reduces steps to success by 41–46%, indicating more efficient reasoning pathways.

Table 1: Performance on WebArena subsets for Gemini-2.5-Flash (success rate and average steps)

Dataset	No Memory	ReasoningBank	NoMem	RBank(Expected)
Multi (29)	10.34% (17.0)	17.24% (10.0)	10.3% (10.0)	13.8% (8.8)
Reddit (106)	68.87% (9.62)	70.75% (10.32)	55.7% (6.7)	67.0% (5.6)

Table 2: Performance on WebArena subsets for Gemini-2.5-Pro (success rate and average steps)

Dataset	No Memory	ReasoningBank	NoMem(Expected)	RBank(Expected)
Multi (29)	13.79% (14.50)	20.69% (9.83)	6.9% (8.8)	13.8% (8.2)
Reddit (106)	64.15% (10.53)	68.87% (9.60)	71.7% (6.0)	80.2% (5.1)

Implementation Status. We have completed the ReAct baseline, core ReasoningBank architecture (memory storage, embedding generation, and retrieval system), and preliminary evaluation. Current work focuses on GitLab environment integration, comprehensive WebArena evaluation, and testing additional model variants. Key challenges include GitLab authentication configuration, balancing memory retrieval efficiency at scale, and ensuring cross-task generalization.

4 Timeline and Remaining Work

Phase 0: Baseline Replication (Weeks 5-6). Complete WebArena evaluation across all subdomains and establish baseline metrics with multiple model variants.

Phase 1: Memory Enhancements (Weeks 7). Develop memory compression algorithms, and hierarchical organization strategies, to improve the memory storage and retrieval.

Phase 2: Cross-Domain Transfer (Weeks 8). Implement transfer mechanisms enabling reasoning strategies learned in web browsing to inform software engineering tasks, with domain-agnostic memory representations.

Phase 3: Final Evaluation (Week 9-10). Conduct comprehensive performance analysis, statistical significance testing, and prepare final deliverables.

Expected Deliverables: (1) Open-source ReasoningBank implementation with full replication results, (2) Enhanced memory system demonstrating scalability improvements, (3) Cross-domain transfer mechanism with preliminary analysis, (4) Comprehensive evaluation across WebArena, Mind2Web, and SWE-Bench-Verified.

5 References

1. Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K., & Cao, Y. (2022). ReAct: Synergizing Reasoning and Acting in Language Models. *arXiv preprint arXiv:2210.03629*. https://arxiv.org/abs/2210.03629

2. Ouyang, S., Yan, J., Hsu, I-H., Chen, Y., Jiang, K., Wang, Z., Han, R., Le, L. T., Daruki, S., Tang, X., Tirumalashetty, V., Lee, G., Rofouei, M., Lin, H., Han, J., Lee, C-Y., & Pfister, T. (2025). ReasoningBank: Scaling Agent Self-Evolving with Reasoning Memory. *arXiv preprint arXiv:2509.25140*. https://arxiv.org/abs/2509.25140
3. Pan, C., Liu, Y., Zhang, J., & Wang, W. (2025). A Survey of Continual Reinforcement Learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. https://arxiv.org/abs/2506.21872
4. Wu, C-K., Chen, Y-C., & Lin, H-T. (2024). StreamBench: Towards Benchmarking Continuous Improvement of Language Agents. *arXiv preprint arXiv:2406.08747*. https://arxiv.org/abs/2406.08747v1
5. Zheng, J., Zhang, Y., Liu, X., & Wang, L. (2025). Lifelong Learning of Large Language Model based Agents: A Roadmap. *arXiv preprint arXiv:2501.07278*. https://arxiv.org/pdf/2501.07278v1
6. Zhou, S., Xu, F. F., Zhu, H., Zhou, X., Lo, R., Sridhar, A., Cheng, X., Ou, T., Bisk, Y., Fried, D., Alon, U., & Neubig, G. (2023). WebArena: A Realistic Web Environment for Building Autonomous Agents. *arXiv preprint arXiv:2307.13854*. https://arxiv.org/abs/2307.13854
7. Deng, X., Gu, M., Zheng, B., Chen, S., Stevens, S., Wang, B., Sun, H., & Su, Y. (2023). Mind2Web: Towards a Generalist Agent for the Web. *arXiv preprint arXiv:2306.06070*. https://arxiv.org/abs/2306.06070
8. Jimenez, C. E., Yang, J., Wettig, A., Yao, S., Pei, K., Press, O., & Narasimhan, K. (2024). SWE-bench: Can Language Models Resolve Real-World GitHub Issues? *arXiv preprint arXiv:2310.06770*. OpenAI. (2024). Introducing SWE-bench Verified. https://openai.com/index/introducing-swe-bench-verified/
9. Drouin, A., Gasse, M., Caccia, M., Lacoste, A., Le Sellier De Chezelles, T., Boisvert, L., Thakkar, M., Marty, T., Assouel, R., Shayegan, S. O., Jang, L. K., Lù, X. H., Yoran, O., Kong, D., Xu, F. F., Reddy, S., Cappart, Q., Neubig, G., Salakhutdinov, R., Chapados, N., & Lacoste, A. (2024). The BrowserGym Ecosystem for Web Agent Research. *arXiv preprint arXiv:2412.05467*. https://arxiv.org/abs/2412.05467
10. Johnson, J., Douze, M., & Jégou, H. (2017). Billion-scale similarity search with GPUs. *arXiv preprint arXiv:1702.08734*. Facebook AI Research. FAISS: Facebook AI Similarity Search. https://github.com/facebookresearch/faiss
11. Google AI. (2025). Gemini Embedding now generally available in the Gemini API. Google Developers Blog. https://developers.googleblog.com/en/gemini-embedding-available-gemini-api/
12. Mallen, A., Vemuri, K., Sharma, A., Pandya, V., & Keller, F. (2025). Generalizable Embeddings from Gemini. *arXiv preprint arXiv:2503.07891*. https://arxiv.org/abs/2503.07891