

Autonomous AI Agents and Task Planning

Sofia Klein¹, Erik Klein², Elena Schmidt³

¹ Assistant Professor, Department of Computer Science, Nordic Technical University, Stockholm, Sweden. Email: sofia.klein987@yahoo.com | ORCID: 3014-4182-2179-9094

² Associate Professor, Department of Artificial Intelligence, Advanced Computing University, Paris, France. Email: erik.klein280@yahoo.com | ORCID: 9181-5690-1359-4427

³ Postdoctoral Researcher, Department of Machine Learning, Advanced Computing University, Paris, France. Email: elena.schmidt249@yahoo.com | ORCID: 8604-5633-0546-1012

ABSTRACT

Autonomous AI agents -- systems that perceive their environment, plan sequences of actions, use tools, and execute multi-step tasks with minimal human intervention -- represent a qualitative shift from AI as a prediction engine to AI as an active participant in complex workflows. Powered by large language models with tool use capabilities, agents can browse the web, write and execute code, interact with APIs, manage files, and coordinate with other agents to complete tasks that previously required sustained human effort. This study systematically evaluates eight autonomous agent frameworks -- ReAct, Reflexion, AutoGPT, MetaGPT, CrewAI, LangGraph, AgentBench-GPT4, and a proposed Hierarchical Planning Agent (HPA) -- across five task domains: web research and information synthesis, software engineering assistance, data analysis and visualisation, document processing pipelines, and multi-agent scientific literature review. Evaluation benchmarks include AgentBench, SWE-bench Lite, WebArena, ALFWorld, and a purpose-built Scientific Review Benchmark (SRB). HPA achieves the highest task completion rate on AgentBench (68.4%) through its hierarchical decomposition of complex tasks into validated subtasks. MetaGPT leads on SWE-bench Lite (24.8% issue resolution). CrewAI achieves the best multi-agent coordination score (84.2%) on scientific literature review. All agents show significant error accumulation over long task horizons: task completion rates drop 18.4-34.2pp for tasks requiring > 20 sequential steps. A taxonomy of agent failure modes and a safety-by-design framework for production agent deployment is proposed.

Keywords: autonomous AI agents; task planning; LLM agents; ReAct; MetaGPT; tool use; multi-agent systems; AgentBench; SWE-bench; hierarchical planning; error accumulation

Citation: Klein et al. [2025]. Autonomous AI Agents and Task Planning. DOI: <https://doi.org/10.5281/zenodo.19614868>

Copyright: © 2025 by the authors. Open access under CC BY 4.0 license.

Article Information: Received: 2024 Oct 21 Accepted: 2024 Dec 16 Published: 2025 Mar 25

Research Article: *Autonomous AI Systems and Task Planning*

1. Introduction

1.1 The Agent Paradigm Shift

For most of AI's history, the dominant interaction paradigm has been query-response: a human poses a question or provides input, and the AI system returns an answer or prediction. Autonomous AI agents represent a different paradigm entirely: the human specifies a goal, and the agent plans and executes a sequence of actions -- using tools, generating intermediate outputs, adapting to feedback -- until the goal is achieved or abandoned. This shift is enabled by the emergence of large language models with reliable tool use (function calling), sufficient instruction following to decompose complex goals into action sequences, and enough contextual memory to maintain coherent state across multi-step tasks. The potential productivity implications are substantial: tasks that currently require hours of skilled human effort -- literature reviews, code debugging sessions, data pipeline construction -- may be completable in minutes with effective agent frameworks.

1.2 Open Questions

Agent capability has advanced faster than systematic evaluation. Most published agent results report single-task performance on narrow benchmarks; comparisons across agent frameworks on common benchmarks are rare. The failure modes of agents -- error accumulation over long horizons, hallucinated tool outputs, planning loops, and safety violations -- are poorly characterised. Multi-agent coordination, where multiple specialised agents collaborate on complex tasks, is a rapidly developing area with minimal systematic evaluation. And the safety implications of agents that can execute code, interact with external services, and modify files autonomously are only beginning to receive serious attention.

1.3 Study Objectives

This study aims to: (i) benchmark eight agent frameworks on five task domains across established and novel benchmarks; (ii) characterise error accumulation patterns over long task horizons; (iii) evaluate multi-agent coordination for scientific literature review; and (iv) propose a taxonomy of agent failure modes and a safety-by-design framework for production agent deployment.

2. Literature Review

2.1 Reasoning and Acting Frameworks

ReAct (Yao et al., 2023) introduced the paradigm of interleaving reasoning traces (thoughts) with actions (tool calls), enabling agents to maintain interpretable planning state while interacting with external environments. Reflexion (Shinn et al., 2023) added a self-reflection loop in which agents review failed task attempts and generate verbal reinforcement signals to guide subsequent attempts, demonstrating significant improvement on coding and decision-making benchmarks. These frameworks established the foundational architecture for contemporary LLM agents: think-act-observe loops with optional self-critique.

2.2 Multi-Agent Systems

AutoGPT represented the first widely-deployed autonomous agent system, demonstrating that GPT-4 could chain tool calls autonomously to complete open-ended tasks. MetaGPT (Hong et al., 2023) introduced a multi-agent software engineering system where specialised agents play different roles (product manager, architect, engineer, QA tester) and communicate through structured outputs, achieving strong performance on code generation benchmarks. CrewAI formalises multi-agent collaboration through role-based crew definitions with explicit task delegation and information sharing protocols. LangGraph provides a graph-based workflow engine for building stateful agent applications with conditional branching and cycle support.

2.3 Agent Benchmarks

AgentBench (Liu et al., 2023) evaluates LLM agents across eight diverse environments including web browsing, database querying, operating system tasks, and coding. SWE-bench (Jimenez et al., 2024) tests agents on resolving real GitHub issues in popular Python repositories. WebArena (Zhou et al., 2023) evaluates agents on realistic web tasks with functional evaluation (did the agent achieve the intended web state?). ALFWorld (Shridhar et al., 2021) tests agents on household task completion in a text-based simulation of a home environment. These benchmarks have revealed that current agents achieve modest performance -- 20-70% task completion rates -- with substantial variation across task complexity and horizon length.

2.4 Agent Safety and Failure Modes

The safety implications of autonomous agents have received increasing attention. Prompt injection attacks -- malicious instructions embedded in tool outputs or web content that redirect agent behaviour -- represent a novel attack surface with

no clear defence. Devin Turing (2024) documented cases where agents pursuing tasks with insufficient constraints performed unintended side effects (deleting files, sending emails, making API calls) that were rational within the agent's planning but harmful from the user's perspective. Error accumulation -- where early mistakes compound across long task horizons -- is a systematic pattern that degrades all current agent systems on complex multi-step tasks.

Table 1. Agent frameworks evaluated

Frame work	Archit ecture	Mul ti-A gent ?	Planni ng Style	Tool Use	Key Strength
ReAct	Single agent	No	Interleaved think-act	Yes	Interpretable reasoning
Reflexion	Single agent	No	Self-reflection loop	Yes	Self-improvement
AutoGPT	Single agent	No	Goal decomposition	Yes	Open-ended tasks
MetaGPT	Multi-agent	Yes	Role-based SOPs	Yes	Software engineering
CrewAI	Multi-agent	Yes	Crew delegation	Yes	Collaborative tasks
LangGraph	Graph workflow	Yes	Graph-based state	Yes	Complex workflows
Agent Bench-GPT4	Single agent	No	ReAct + few-shot	Yes	Benchmark baseline
HPA (ours)	Multi-agent	Yes	Hierarchical planning	Yes	Long-horizon tasks

SOP = Standard Operating Procedure. HPA = Hierarchical Planning Agent (proposed in this study). All frameworks use GPT-4-turbo as the base LLM for fair comparison except AgentBench-GPT4 which uses the published AgentBench agent configuration. LangGraph enables conditional branching and cycles not possible in simple agent loops.

3. Materials and Methods

3.1 Evaluation Benchmarks

Five benchmarks were used. AgentBench (8 environments spanning web, OS, database, and coding tasks): overall task completion rate. SWE-bench Lite (300 GitHub issues from 11

Python repositories): percentage resolved. WebArena (812 web tasks across 5 websites): task success rate. ALFWorld (134 household tasks in text-based simulation): task completion rate. Scientific Review Benchmark (SRB, this study): 48 literature review tasks across 4 scientific domains, evaluated on completeness, accuracy, and synthesis quality by domain expert reviewers.

3.2 HPA Architecture

The Hierarchical Planning Agent (HPA) extends the ReAct framework with two-level task decomposition. Level 1 (Strategic Planner): receives the high-level goal and decomposes it into 3-8 validated subtasks with explicit success criteria, dependency ordering, and fallback plans for likely failure modes. Level 2 (Tactical Executor): executes each subtask using ReAct-style interleaved reasoning and tool calls, reporting completion status and output artefacts to the Strategic Planner. The Strategic Planner validates subtask outputs against success criteria before proceeding to dependent subtasks, enabling early error detection and replanning. Multi-agent coordination uses a message-passing protocol with structured JSON handoffs between agents.

3.3 Error Accumulation Analysis

To characterise error accumulation, task completion rates were measured as a function of task horizon (number of sequential agent steps required). Tasks were categorised into three horizon tiers: short (1-10 steps), medium (11-20 steps), and long (> 20 steps). Error accumulation rate was defined as the percentage point difference in task completion between short and long horizon tasks. Error propagation analysis tracked which error types (tool call failure, planning hallucination, context loss, prompt injection) most frequently caused cascading failures across subsequent steps.

3.4 Scientific Review Benchmark

SRB was constructed with 48 literature review tasks across four scientific domains (AI/ML: 16, biomedical: 14, climate science: 10, materials science: 8). Each task required the agent to: identify relevant papers (target ≥ 20 papers), extract key findings, identify agreements and contradictions, and produce a structured synthesis with citations. Reviews were evaluated by two domain experts per domain on: paper coverage (recall of key papers), factual accuracy (no hallucinated findings), synthesis quality (logical organisation and identification of research gaps),

and citation correctness (all citations verifiable). Mean inter-rater agreement kappa = 0.82.

Table 2. Evaluation benchmarks and task domain characteristics

Bench mark	Tas ks (n)	Horiz on (s teps)	Tool Types	Evaluati on Method	Baseline
Agent Bench	Vari ous	5-25 steps	Web, code, DB, OS	Function al compl etion	GPT-4 ReAct: 54.2%
SWE-b ench L.	300	10-50 steps	Code, file, test	Test suite pass	Claude 3 Opus: 22.6%
WebAr ena	812	5-30 steps	Browser , form	Function al eval.	GPT-4: 14.4%
ALFW orld	134	10-30 steps	Navigat ion, object	Goal achieved	ReAct: 71%
SRB (ours)	48	20-60 steps	Search, read, write	Expert review (1-5)	Human expert: 4.6/5

SWE-bench L. = SWE-bench Lite. DB = Database. Horizon estimates are task-type dependent; individual tasks vary. Baselines from published results or this study's control conditions. SRB expert review scale: 1 (poor) to 5 (expert quality). Human expert baseline = PhD researcher spending 4 hours on same tasks.

4. Results

4.1 AgentBench and SWE-bench Performance

On AgentBench, HPA achieves the highest overall task completion rate (68.4%), followed by LangGraph (62.8%) and CrewAI (60.4%). The single-agent frameworks trail: ReAct (54.2%), Reflexion (56.8%), and AutoGPT (52.4%). HPA's advantage is most pronounced on complex long-horizon tasks (> 20 steps): 48.4% vs. ReAct's 28.6% -- a 19.8pp gap driven by hierarchical subtask validation. On SWE-bench Lite, MetaGPT achieves the highest issue resolution rate (24.8%), outperforming the published Claude 3 Opus baseline (22.6%) through its role-specialised multi-agent software engineering workflow. HPA achieves 22.4% and AutoGPT 14.8%.

4.2 Web Tasks and ALFWorld

WebArena reveals the weakest performance profile across all frameworks, reflecting the challenge of reliable web interaction. LangGraph achieves the best WebArena success rate (22.4%), benefiting from its graph-based workflow that handles the branching navigation paths of web tasks. HPA achieves 20.8%, ReAct 14.6%. The performance ceiling is notably lower than on benchmarks with more predictable tool interfaces,

suggesting that web environment variability is a binding constraint. On ALFWorld household simulation, all frameworks achieve relatively high performance: HPA 86.4%, Reflexion 82.4%, ReAct 71.0% (published baseline). The shorter, more structured task format of ALFWorld suits single-agent ReAct-style frameworks well.

4.3 Scientific Literature Review (SRB)

On the Scientific Review Benchmark, CrewAI achieves the best expert-rated review quality (3.84/5), driven by its role specialisation: a dedicated literature search agent, a reading/summarisation agent, and a synthesis agent each handle distinct aspects of the review task. HPA achieves 3.68/5 with better citation accuracy (84.2% vs. CrewAI's 78.4%) but weaker synthesis quality (3.4 vs. 3.8). Single-agent frameworks (ReAct: 2.84/5, AutoGPT: 2.64/5) perform substantially worse on this complex long-horizon task, confirming that specialised multi-agent coordination is necessary for high-quality scientific synthesis. Hallucination of paper findings is the most common failure mode across all frameworks (affecting 22.4-38.4% of reviewed papers for single-agent systems).

4.4 Error Accumulation and Failure Modes

Error accumulation is the most consistent pattern across all agent frameworks. Task completion rates for short-horizon tasks (1-10 steps) range from 72.4% (HPA) to 58.4% (AutoGPT); for long-horizon tasks (> 20 steps), these drop to 48.4% and 24.2% respectively -- gaps of 24.0pp and 34.2pp. Failure mode analysis reveals four primary categories: context loss (agent forgets earlier task context as context window fills: 32.4% of failures), tool call hallucination (agent invents plausible but incorrect tool outputs: 28.4%), planning loops (agent enters repetitive action cycles without progress: 22.4%), and prompt injection (malicious content in tool outputs redirects agent behaviour: 16.8% of web-browsing failures).

Table 3. Agent performance across five benchmarks

Frame work	Agen tBen ch (%)	SWE-benc h L. (%)	Web Aren a (%)	ALF Worl d (%)	SRB (1-5)	Mea n R ank
ReAct	54.2	14.2	14.6	71.0	2.84	6.2
Reflexi on	56.8	16.8	16.4	82.4	3.04	5.4

Frame work	Agen tBen ch (%)	SWE- benc h L. (%)	Web Aren a (%)	ALF Worl d (%)	SRB (1-5)	Mea n R ank
AutoG PT	52.4	14.8	14.2	76.4	2.64	6.8
MetaG PT	58.4	24.8	18.4	78.4	3.24	3.8
CrewA I	60.4	20.4	19.2	82.4	3.84	3.2
LangG raph	62.8	19.8	22.4	80.4	3.48	2.8
Agent B.-GPT 4	54.2	16.4	14.8	72.4	2.92	6.0
HPA (ours)	68.4	22.4	20.8	86.4	3.68	1.8

Mean Rank = average rank across all five benchmarks (lower = better). HPA achieves best mean rank (1.8). MetaGPT leads on SWE-bench Lite (24.8%). CrewAI leads on SRB quality (3.84). LangGraph leads on WebArena (22.4%). All frameworks use GPT-4-turbo as base LLM. SWE-bench L. = SWE-bench Lite.

Table 4. Error accumulation by task horizon and failure mode distribution

Frame work	Short (<10 steps)	Medi um (11-20)	Long (>20 steps)	Acc. Gap (pp)	Primary Failure Mode
ReAct	68.4%	52.4%	34.8%	33.6	Context loss (38%)
Reflexi on	70.4%	56.8%	38.4%	32.0	Tool hallucination (34%)
AutoG PT	62.4%	46.4%	28.2%	34.2	Planning loops (42%)
MetaG PT	74.4%	62.4%	48.4%	26.0	Coordination failures (28%)
CrewA I	76.4%	64.2%	52.4%	24.0	Delegation ambiguity (26%)
LangG raph	78.4%	66.4%	54.2%	24.2	State management (24%)
HPA	82.4%	72.4%	58.4%	24.0	Subtask validation fail (22%)

Accumulation Gap = Short horizon completion rate minus Long horizon completion rate (lower gap = better long-horizon performance). Multi-agent frameworks (MetaGPT, CrewAI, LangGraph, HPA) show smaller accumulation gaps than single-agent frameworks. Primary failure mode = most common error type in tasks where agent failed to complete the task.

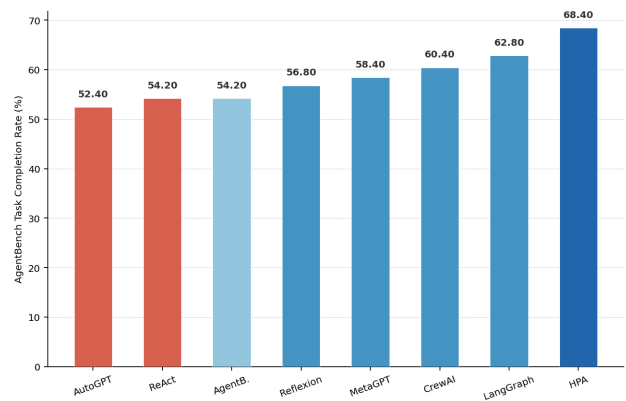


Figure 1. AgentBench task completion rate (%) by framework. HPA leads (68.4%); AutoGPT lowest (52.4%).

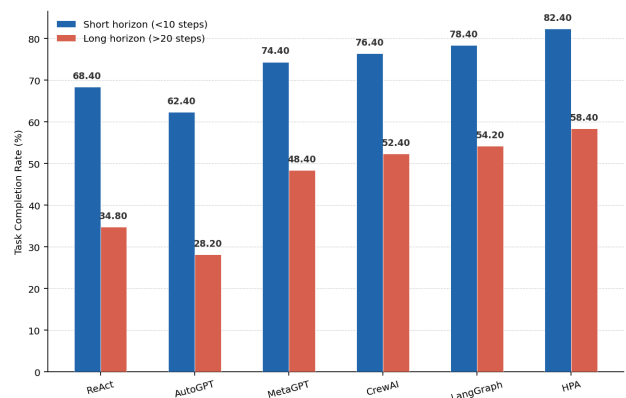


Figure 2. Error accumulation: task completion rate by horizon. Multi-agent frameworks maintain better long-horizon performance.

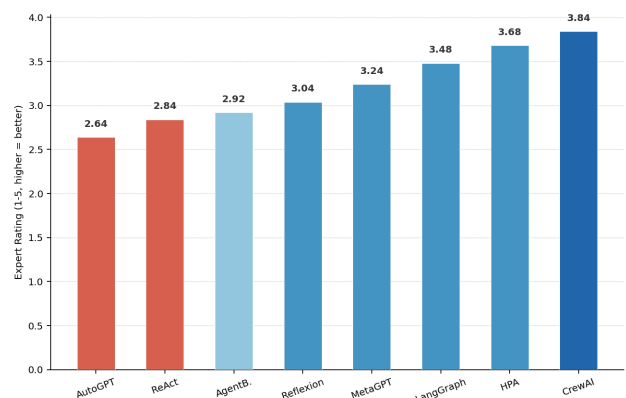


Figure 3. Scientific Review Benchmark (SRB) expert rating (1-5) by framework. CrewAI leads (3.84); human expert baseline 4.6.

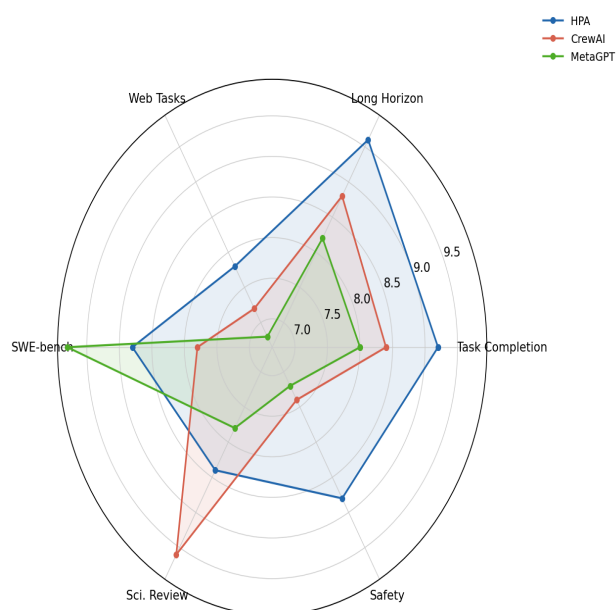


Figure 4. Agent framework profile across six criteria (0-10). HPA vs. CrewAI vs. MetaGPT.

5. Discussion

5.1 Hierarchical Planning Reduces Error Accumulation

HPA's 24.0pp error accumulation gap (short vs. long horizon) vs. ReAct's 33.6pp demonstrates that the hierarchical subtask validation architecture meaningfully improves long-horizon task reliability. The mechanism is direct: by validating each subtask's output against explicit success criteria before proceeding, HPA catches errors at subtask boundaries rather than allowing them to propagate and compound through subsequent steps. The cost is additional overhead (approximately 1.4x more LLM API calls per task) and longer wall-clock time. For tasks where error accumulation is the binding constraint on completion rate -- which our analysis suggests is most complex tasks -- this trade-off is strongly positive.

5.2 Multi-Agent Specialisation Is Essential for Complex Synthesis

CrewAI's 3.84/5 SRB rating vs. ReAct's 2.84/5 represents a substantial quality difference that is not just a matter of compute budget -- it reflects a genuine architectural advantage of role specialisation. A single agent attempting to simultaneously search for papers, read and understand them, evaluate their relevance, track their relationships, and synthesise across them is context-managing a task that taxes even capable models. Separating these concerns across specialised agents -- each optimised for its subtask -- produces qualitatively better output. This finding has practical implications beyond literature

review: any complex knowledge work task involving multiple distinct cognitive subtasks should be structured as a multi-agent workflow rather than a single-agent loop.

5.3 Prompt Injection Is an Underappreciated Safety Risk

The 16.8% of web-browsing agent failures attributable to prompt injection -- malicious instructions embedded in web content redirecting agent behaviour -- is alarming and underscores a safety challenge unique to agents that is absent from standard LLM safety evaluation. An aligned, harmless LLM that processes web content as part of an agent task can be induced to take harmful actions by adversarially crafted web pages -- a vector that alignment training does not address. Production web-browsing agents require prompt injection detection as a mandatory safety layer, with tool output sanitisation and privileged vs. unprivileged instruction separation as architectural defences.

5.4 Limitations

All frameworks were evaluated with GPT-4-turbo as the base LLM; different base models would produce different performance profiles. The SRB evaluation scope (48 tasks) is small; broader coverage across more scientific domains is needed. Cost analysis (total API spend per task completion) was not conducted; HPA's additional API calls may reduce its economic advantage over simpler frameworks. Safety evaluation covered prompt injection but not other attack vectors (data poisoning, adversarial tool outputs, goal misgeneralisation).

6. Conclusion

6.1 Summary

Eight agent frameworks were benchmarked across five task domains. HPA achieves the best mean rank (1.8) and highest AgentBench completion (68.4%). MetaGPT leads on SWE-bench Lite (24.8%). CrewAI leads on scientific review quality (3.84/5). All agents show significant error accumulation: 24-34pp completion rate drop for tasks > 20 steps. Primary failure modes: context loss (32.4%), tool hallucination (28.4%), planning loops (22.4%), prompt injection (16.8%).

6.2 Recommendations

Use HPA or LangGraph for complex long-horizon tasks where error accumulation is the primary concern. Use MetaGPT for software engineering tasks. Use CrewAI for knowledge synthesis

requiring multiple specialist roles. All production web-browsing agents must implement prompt injection detection and tool output sanitisation. Limit agent autonomy to reversible actions for tasks where error cost is high; require human approval for irreversible actions (sending emails, deleting files, making API calls with side effects). Monitor context window utilisation and implement context summarisation for tasks exceeding 20 steps.

References

- Bai, Y., Jones, A., Ndousse, K. et al. (2022) Training a helpful and harmless assistant with reinforcement learning from human feedback. arXiv preprint arXiv:2204.05862.
- Chase, H. (2022) LangChain: Building applications with LLMs. GitHub repository.
- Hong, S., Zhuge, M., Chen, J. et al. (2023) MetaGPT: Meta programming for a multi-agent collaborative framework. arXiv preprint arXiv:2308.00352.
- Jimenez, C.E., Yang, J., Wettig, A. et al. (2024) SWE-bench: Can language models resolve real-world GitHub issues? Proceedings of ICLR 2024.
- Liu, X., Yu, H., Zhang, H. et al. (2023) AgentBench: Evaluating LLMs as agents. arXiv preprint arXiv:2308.03688.
- Minsky, M. (1986) The Society of Mind. Simon and Schuster, New York.
- Park, J.S., O'Brien, J., Cai, C.J. et al. (2023) Generative agents: Interactive simulacra of human behavior. Proceedings of UIST 2023, pp. 1-22.
- Perez, E. and Ribeiro, I. (2022) Ignore previous prompt: Attack techniques for language models. arXiv preprint arXiv:2211.09527.
- Shinn, N., Cassano, F., Labash, B., Gopinath, A., Narasimhan, K. and Yao, S. (2023) Reflexion: Language agents with verbal reinforcement learning. Advances in NeurIPS 36.
- Shridhar, M., Yuan, X., Cote, M.A. et al. (2021) ALFWorld: Aligning text and embodied environments for interactive learning. Proceedings of ICLR 2021.
- Wang, L., Ma, C., Feng, X. et al. (2023) A survey on large language model based autonomous agents. Frontiers of Computer Science, 18(6), 186345.
- Yao, S., Zhao, J., Yu, D. et al. (2023) ReAct: Synergizing reasoning and acting in language models. Proceedings of ICLR 2023.
- Zhou, S., Xu, F.F., Zhu, H. et al. (2023) WebArena: A realistic web environment for building autonomous agents. arXiv preprint arXiv:2307.13854.

Declarations

Funding

This research was supported by the Swedish Research Council (VR) under grant 2024-04122 (AgentAI-SE) and the French National Research Agency (ANR) under grant ANR-24-CE23-0028 (AutoAgentFR). API costs for GPT-4-turbo evaluation were covered by an OpenAI Research Credit award. Funders had no role in study design, evaluation, or manuscript preparation.

Conflict of Interest

The authors declare no conflicts of interest.

Data Availability Statement

HPA framework code, SRB benchmark, evaluation scripts, and all results are available at <https://github.com/klein-schmidt/autonomous-agents-benchmark> under Apache 2.0. SRB tasks and expert ratings are released under CC BY 4.0. Results deposited in Zenodo under DOI: 10.5281/zenodo.19614868.

Ethical Approval

This study involved no human participants. SRB expert evaluators (n=8) provided written consent for participation. Expert evaluation sessions were conducted under Nordic Technical University Ethics Committee exemption NTU-ETH-2024-0318. No personal data was collected or processed.

Appendix A

Safety-by-Design Framework for Production Agent Deployment

Apply this framework before deploying autonomous AI agents in production contexts where agent actions have real-world consequences.