Deploying Agents in the Wild: Failure Modes from Healthcare Access Optimization

Diego Estuar
Cedars-Sinai Medical Center
Los Angeles, CA, USA
Claremont Graduate University
Claremont, CA, USA
diego.estuar@cshs.org
diego.estuar@cgu.edu

Abstract

We report on a multi-agent system deployed for patient access optimization at a large academic medical center. The system queries databases, processes heterogeneous documents, searches institutional knowledge, and synthesizes clinic-specific action plans for analyst review. In a pilot across 38 ambulatory specialties, analysts estimated that draft generation decreased from a typical 3–4 hours of manual preparation to approximately 5 minutes of agent generation, followed by 30–60 minutes of expert review. All drafts were useful, but also required revision. Revisions clustered around three semantic failures: context omission, metric misuse, and policy overextension. Drawing on three facilitated working sessions conducted during a three-month pilot, we derive a qualitative failure taxonomy that distinguishes semantic failures from technical failures in deployed agent systems.

1 Introduction

Tool-enabled LLM agents [Schick et al., 2023, Yao et al., 2023] can reason over natural language, invoke structured APIs, and iteratively refine their approach. At the same time, multi-agent frameworks [Wu et al., 2024, Hong et al., 2024] and engineering patterns for tool design, harnesses, and orchestration [Aizawa, 2025, Young, 2025, Rajasekaran, 2026, Martin et al., 2026] have matured rapidly. Despite this progress, comparatively few accounts describe what happens when such systems are deployed in production settings.

Healthcare operations offers a particularly demanding environment for these systems [Topol, 2019, Rajkomar et al., 2019, Sutton et al., 2020]. Academic medical centers typically depend on disconnected EHR-derived warehouses, scheduling dashboards, survey reports, and policy repositories that have limited interoperability. Although prior work has examined administrative applications of LLM agents [Gebreab et al., 2024] and benchmarked systems on structured datasets [Sahu et al., 2024], real-world deployments remain difficult to evaluate. In practice, they require synthesis across noisy sources and are often judged through unstructured expert critique rather than clean benchmark labels.

The deployment we study emerged from this operational context. Within the institution, the Performance Improvement (PI) team produces clinic-specific action plans by reviewing survey reports, cross-referencing dashboards, querying the data warehouse, and consulting an institutional countermeasure knowledge base. Producing a single action plan typically requires 3–4 hours of analyst effort, which constrains delivery. To address this bottleneck, we deployed a multi-agent system that generates draft action plans for analyst review across 38 ambulatory specialties. Ambulatory specialties refers to outpatient clinical service lines, such as specialty clinics that manage appointment access outside the inpatient hospital setting.

1st Conference on AI and Agentic Systems (ACM CAIS 2026). Workshop: AI Agents for Discovery in the Wild.

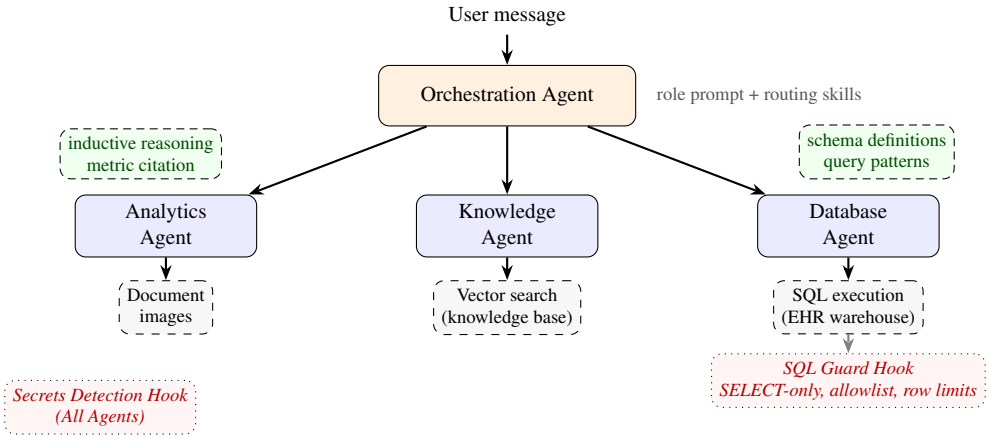

Figure 1: System architecture. The orchestration agent routes to sub-agents via tool calls. Each sub-agent is configured with a role prompt, skills, scoped tools, and a hook pipeline. Hooks enforce safety invariants at the harness level: the SQL Guard Hook validates every query by AST parsing; the Secrets Detection Hook redacts credentials across all agents.

Against this background, the paper makes two contributions. First, we provide a deployment report from a real healthcare institution over a three-month pilot. Second, we present a six-part failure taxonomy that distinguishes semantic failures from technical failures in deployed agent systems.

2 System Overview

The multi-agent system leverages a minimal ReAct-style harness [Yao et al., 2023] and the OpenAI GPT-5.2 model API. Model output is streamed, requested tool calls are executed, results are appended to context, and the loop repeats until a final response is produced. We deliberately avoided planning modules, persistent memory, and framework-specific abstractions [Orogat et al., 2026] so that observed failures would remain attributable to specific agents and tools.

As shown in Figure 1, the system consists of an orchestration agent and three task-specific sub-agents. Each sub-agent shares the same harness and is configured through four components: a *role prompt*, *skills* implemented as natural-language heuristics appended to the prompt, a *tool set*, and a *hook pipeline* that enforces invariants before and after tool calls [Rebedea et al., 2023, Dong et al., 2024]. The orchestration agent routes document extraction and metric synthesis to the Analytics Agent, institutional best-practice questions to the Knowledge Agent, and live data retrieval to the Database Agent. The three-agent decomposition mirrored the operational separation among evidence types: documents, institutional knowledge, and live database metrics. This separation allowed prompts, tools, and safety hooks to be scoped to each evidence source. We did not evaluate this decomposition against a single-agent or template-based design, so we treat it as a pragmatic deployment choice rather than an optimized architecture.

During each turn, the orchestration agent invokes a sub-agent with its role prompt, static skills, scoped tools, and hook pipeline. Tools include document-image viewing, vector search, and SQL execution. Hooks operate at the harness level: the SQL Guard Hook enforces SELECT-only access, table allowlists, and row limits, while the Secrets Detection Hook redacts credential patterns. The orchestration agent's own tool set consists entirely of sub-agent invocation functions, so it never accesses data sources directly.

3 Deployment and Evaluation

The system was piloted with the PI team across 38 ambulatory specialties at a large academic medical center. Evaluation combined head-to-head comparison against manually produced worksheets with three facilitated working sessions held from January through March 2026. These sessions involved

analysts, managers, directors, and the development team. Each session included a live demonstration, followed by discussion of output quality, gaps, and failure modes.

We frame this process as practitioner evaluation within an operational improvement effort rather than as formal qualitative research. Accordingly, we did not perform systematic coding or inter-rater reliability analysis, and developer participation in both design and evaluation is an important limitation. To reduce bias, however, the PI team rather than the developers determined which revisions were required before delivery.

The operational effect was nonetheless substantial. Manually producing an action plan typically requires 3–4 hours per specialty. By contrast, the agent generated drafts in approximately 5 minutes, and analysts estimated an additional 30–60 minutes for review, yielding end-to-end times of roughly 35–65 minutes. All 38 drafts were considered useful, but none were accepted without revision. We define a revision as any analyst-required change before a draft action plan could be delivered to operational stakeholders.

Analysts reviewed all drafts and recorded required changes before delivery. Because revisions were documented during operational review rather than coded as a formal qualitative dataset, we report recurring practitioner-observed patterns rather than quantified prevalence estimates. These patterns informed the failure taxonomy in Section 4.

A smaller subset of the pilot allowed direct comparison against historical manual work. Four specialties had manually produced worksheets available for side-by-side review. Across these cases, agent and manual reports aligned on core recommendations such as template optimization and time-off planning. Template optimization refers to changes in clinician scheduling templates, such as adjusting appointment slot types, session structure, or time-off planning to improve access. Agent drafts were generally more comprehensive and analytically detailed, and they surfaced additional data-driven recommendations that were absent from the manual reports. Manual worksheets, however, captured contextual knowledge that the system could not access. In one case, the agent explicitly flagged missing subspecialty information in a *Data Gaps* section rather than inferring details, a behavior participants viewed favorably.

Across the working sessions, three themes recurred. First, explicit citation of source metrics increased perceived trustworthiness. Second, participants expressed concern about definitive language when operational context was missing, a pattern consistent with known automation-bias risks [Parasuraman and Riley, 1997, Goddard et al., 2012]. Third, they identified recurring misinterpretation failures that informed the taxonomy in Section 4.

4 Failure Categorization

Table 1 presents a qualitative taxonomy of failure modes derived from working session observations and development logs. We retain the distinction between *semantic failures* and *technical failures*. Semantic failures occur when the system has plausible evidence but reaches an inappropriate operational conclusion. Technical failures occur when the system fails to extract, retrieve, or assemble the evidence needed for judgment. This distinction matters because technical failures are often visible through missing outputs, malformed evidence, or tool errors, while semantic failures may remain fluent, cited, and operationally plausible. In one session, for example, the agent cited a real scheduling metric but confused percentage of blocked time with percentage of total deployed time, producing a traceable but substantively wrong recommendation.

5 Discussion

The central limitation in this deployment was missing organizational knowledge-driven judgment. Experienced analysts interpret clinic metrics through tacit knowledge [Polanyi, 1966], including staffing realities, referral patterns, physician preferences, and local exceptions to institutional norms. Static skills [Zhang et al., 2025] and retrieval-augmented institutional knowledge helped, but remained brittle because a useful heuristic in one clinic could mislead in another. The system often aligned with analysts on data-visible issues while diverging on context-dependent judgments.

From these findings, three broader implications follow. First, institutionally rich domains may benefit from structured knowledge elicitation at the point of use rather than additional generic reasoning

Table 1: Failure taxonomy from the deployment. The taxonomy separates semantic failures, where available evidence is used to form an inappropriate operational conclusion, from technical failures, where the system fails to extract, retrieve, or assemble the evidence needed for judgment.

Type	Failure	Definition	Example
Semantic	Context omission	Forms a recommendation without local operational knowledge needed to judge feasibility.	Recommending access changes without staffing, referral, or subspecialty constraints.
	Metric misuse	Uses a real metric to support an inappropriate operational conclusion.	Treating percentage of blocked time as percentage of total deployed time.
	Policy overextension	Applies a general institutional rule or target outside its valid operational scope.	Applying a standard access target where it is structurally infeasible.
Technical	Perception error	Incorrectly extracts information from a visual or document artifact.	Misreading a stacked bar chart with overlapping labels.
	Query error	Retrieves incorrect or misleading structured data through a flawed database query.	Joining on a schema-valid column with nulls for the target population.
	Context assemblage error	Fails to carry relevant evidence into the final reasoning context because of chunking, context-window, or synthesis failure.	Omitting critical content from later pages of a long survey.

capability alone. Second, semantic failures deserve particular attention because they are harder for reviewers to detect and may be reinforced by provenance mechanisms that signal false assurance. Third, users appeared to develop more appropriate reliance boundaries over repeated exposure, treating the system as useful for initial orientation rather than as a substitute for direct analysis.

Citation functioned as a provenance mechanism rather than a validity mechanism. It helped analysts inspect where a recommendation came from, but it did not prove that the cited metric supported the proposed intervention. This created a false-assurance risk, where grounded recommendations could still use accurate evidence to justify a potentially inappropriate conclusion.

Missing operational context should not be read to mean that more context would eliminate expert revision. Even with richer inputs, agents may hallucinate, overgeneralize, misread policy, or misuse metrics. In this deployment, the most concerning failures were often that the system cited real metrics or policies and then drew the wrong operational conclusion. Grounding reduced fabrication risk, but did not prevent these semantic failures. The deployment suggests that the most consequential agent failures in institutional workflows may occur after evidence has been retrieved, when the system converts grounded information into operational judgment.

Several limitations qualify these conclusions. Time savings combine logged agent generation time, where available, with analyst-estimated manual preparation and review time, so they should be interpreted as operational estimates rather than controlled time-motion measurements. We did not compare the multi-agent design against single-agent or template-based alternatives, and the pilot used one model. The deliberate exclusion of planning modules, persistent memory, and framework-specific abstractions may limit transferability to other agent systems. Direct head-to-head comparison was possible for only four specialties. The taxonomy was developed from operational review notes without independent coding or inter-rater reliability measures, and expert review may have been affected by anchoring because analysts revised drafts rather than starting from blank worksheets. The system also lacked persistent correction capture across sessions.

Ethics Statement

The system operates strictly as decision support, and all recommendations require expert review before delivery. The pilot was evaluated under institutional criteria for quality-improvement activities that do not constitute human subjects research.

References

Ken Aizawa. Writing effective tools for agents – with agents. Anthropic Engineering Blog, 2025. https://www.anthropic.com/engineering/writing-tools-for-agents.

Yi Dong, Ronghui Mu, Gaojie Jin, Yi Qi, Jinwei Hu, Xingyu Zhao, Jie Meng, Wenjie Ruan, and Xiaowei Huang. Safeguarding large language models: A survey. In *Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML-PKDD)*, 2024.

Feven Gebreab, Cheng-Bang Chiang, Shi Chen, and Shao-Chieh Li. LLM-based framework for administrative task automation in healthcare. *npj Digital Medicine*, 2024.

Kate Goddard, Abdul Roudsari, and Jeremy C Wyatt. Automation bias: a systematic review of frequency, effect mediators, and mitigators. *Journal of the American Medical Informatics Association*, 19(1):121–127, 2012.

Sirui Hong, Mingchen Zhuge, Jonathan Chen, Xiawu Zheng, Yuheng Cheng, Ceyao Zhang, Jinlin Wang, Zili Wang, Steven Ka Zhong Yau, Zijuan Lin, et al. MetaGPT: Meta programming for a multi-agent collaborative framework. In *International Conference on Learning Representations (ICLR)*, 2024.

Lance Martin, Gabe Cemaj, and Michael Cohen. Scaling managed agents: Decoupling the brain from the hands. Anthropic Engineering Blog, 2026. https://www.anthropic.com/engineering/managed-agents.

Charbel Orogat, Ahmad Noureddine, Joy Daou, and Joe Tekli. MAFBench: An evaluation framework for multi-agent LLM frameworks. In *AAAI Conference on Artificial Intelligence*, 2026.

Raja Parasuraman and Victor Riley. Humans and automation: Use, misuse, disuse, abuse. *Human Factors*, 39(2):230–253, 1997.

Michael Polanyi. *The Tacit Dimension*. University of Chicago Press, 1966.

Prithvi Rajasekaran. Harness design for long-running application development. Anthropic Engineering Blog, 2026. https://www.anthropic.com/engineering/harness-design-long-running-apps.

Alvin Rajkomar, Jeffrey Dean, and Isaac Kohane. Machine learning in medicine. *New England Journal of Medicine*, 380(14):1347–1358, 2019.

Traian Rebedea, Razvan Dinu, Makesh Narsimhan Sreedhar, Christopher Parisien, and Jonathan Cohen. NeMo Guardrails: A toolkit for controllable and safe LLM applications with programmable rails. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: System Demonstrations (EMNLP)*, pages 431–445. ACL, 2023.

Gaurav Sahu, Abhay Puri, Juan Rodriguez, Alexandre Drouin, Perouz Taslakian, Valentina Zantedeschi, Alexandre Lacoste, David Vazquez, Nicolas Chapados, and Christopher Pal. InsightBench: Evaluating business analytics agents through multi-step insight generation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 36, 2023.

Reed T Sutton, David Pincock, Daniel C Baumgart, Daniel C Sadowski, Richard N Fedorak, and Karen I Kroeker. An overview of clinical decision support systems: benefits, risks, and strategies for success. *npj Digital Medicine*, 3(1):17, 2020.

Eric J Topol. High-performance medicine: the convergence of human and artificial intelligence. *Nature Medicine*, 25(1):44–56, 2019.

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, et al. AutoGen: Enabling next-gen LLM applications via multi-agent conversation. In *International Conference on Machine Learning (ICML)*, 2024.

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

Justin Young. Effective harnesses for long-running agents. Anthropic Engineering Blog, 2025. `https://www.anthropic.com/engineering/effective-harnesses-for-long-running-agents`.

Barry Zhang, Keith Lazuka, and Mahesh Murag. Equipping agents for the real world with agent skills. Anthropic Engineering Blog, 2025. `https://www.anthropic.com/engineering/equipping-agents-for-the-real-world-with-agent-skills`.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction state three contributions: (1) a deployment report from a three-month pilot (Sections 3–5), (2) a qualitative taxonomy of analyst revisions characterizing where agent output diverges from expert judgment (Section 3), and (3) a failure categorization distinguishing semantic from technical failures (Section 4). Each is addressed in the corresponding sections.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Section 5 discusses tacit knowledge challenges, the brittleness of static skills, and limitations of the evaluation including: absence of architectural ablations, single-model deployment, limited head-to-head comparison scope (four specialties), builder-evaluator overlap, and the possibility that observed trust dynamics reflect familiarity effects rather than a generalizable pattern. Section 3 describes the evaluation methodology and its constraints, including explicit acknowledgment of the builder-evaluator overlap and the mitigations employed.

3. **Theory assumptions and proofs**

   Answer: [N/A]

   Justification: This paper does not include theoretical results; it is a systems and deployment paper.

4. **Experimental result reproducibility**

   Answer: [No]

   Justification: The high-level architecture is described in sufficient detail to inform an independent implementation, but exact reproduction is not possible because institutional prompts, skill files, data sources, and knowledge-base contents cannot be released under privacy and governance constraints. Qualitative findings are grounded in observations from facilitated working sessions (methodology described in Section 3), but the evaluation is not reproducible in the traditional sense. We view this as inherent to deployment reports on institutional systems.

5. **Open access to data and code**

   Answer: [No]

   Justification: The system operates on institutional healthcare data that cannot be shared due to privacy and institutional data governance policies. The agent harness architecture is described in detail sufficient for independent implementation.

6. **Experimental setting/details**

   Answer: [Yes]

   Justification: Section 1 describes the institutional setting. Section 2 describes the agent architecture in detail, including the harness loop, the four configurable components (role prompts, skills, tool sets, hook pipelines), the routing logic, and the safety mechanisms. Section 3 describes the pilot deployment across 38 specialties, including head-to-head comparisons on four specialties, time-to-delivery measurements, a qualitative revision taxonomy, and three facilitated working sessions.

7. **Experiment statistical significance**

   Answer: [N/A]

   Justification: The pilot study reports time-to-delivery measurements and qualitative comparisons rather than statistical hypothesis tests. The sample size (four specialty comparisons) supports descriptive findings but not statistical significance claims.

8. **Experiments compute resources**

   Answer: [N/A]

   Justification: The system uses commercial LLM APIs; no custom training or large-scale compute was involved.

9. **Code of ethics**

   Answer: [Yes]

   Justification: The Ethics Statement describes the decision-support-only design, data governance, and human oversight requirements. The system does not access individual patient records in its recommendation pipeline.

10. **Broader impacts**

    Answer: [Yes]

    Justification: The Ethics Statement and Section 5 discuss human oversight requirements, trust calibration, automation-bias risks, and the importance of failure transparency. The system is designed as decision support with mandatory expert review.

11. **Safeguards**

    Answer: [Yes]

    Justification: Section 2 describes the safety hook system in detail, including AST-level SQL validation, table allowlists, row limits, and credential redaction. Hooks operate at the harness level and enforce invariants regardless of model output.

12. **Licenses for existing assets**

    Answer: [N/A]

    Justification: No external datasets or pre-existing code assets are used in the paper.

13. **New assets**

    Answer: [N/A]

    Justification: No new datasets or models are released with this paper.

14. **Crowdsourcing and research with human subjects**

    Answer: [No]

    Justification: Working session participants (Section 3) were members of the operational team evaluating their own tooling as part of routine workflow improvement, not research subjects. We acknowledge the QI/research boundary tension in the Ethics Statement.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Answer: [N/A]

    Justification: The pilot was evaluated under institutional criteria for quality-improvement activities that do not constitute human subjects research. The Ethics Statement acknowledges the tension in publishing QI findings at a research venue.

16. **Declaration of LLM usage**

    Answer: [Yes]

    Justification: LLMs are the core component of the system described. Section 2 details the agent architecture and the model used (GPT-5.2), including how LLMs are used for document extraction, knowledge synthesis, database querying, and recommendation generation.