AssetOpsBench-Live: Privacy-Aware Online Evaluation of Multi-Agent Performance in Industrial Operations

Dhaval Patel[1*], Nianjun Zhou[1*], Shuxin Lin[1*], James Rayfield[1*], Chathurangi Shyalika[2*], Suryanarayana Reddy Yarrabothula[3*]

[1]IBM Research, Yorktown
[2]Artificial Intelligence Institute, University of South Carolina
[3]Steel Authority of India Limited

Abstract

Industrial automation increasingly relies on multi-agent AI, yet evaluation remains difficult due to task complexity and data confidentiality. We present `AssetOpsBench-Live`, a demo of a competition-ready platform for real-time, privacy-preserving evaluation of multi-agent AI in industrial contexts. The platform integrates `AssetOpsBench`, which measures six dimensions of multi-agent performance and performs automated failure-mode discovery, with `Codabench`, which supports reproducible, code-oriented competitions. End users first validate agents locally, then submit containerized code for execution on hidden industrial scenarios. Instead of raw trajectories, the system provides quantitative scores and clustered failure modes (e.g., *reasoning–action mismatch*, *step repetition*), enabling participants to identify failures, apply targeted improvements, and iteratively resubmit. By combining competition-based engagement with actionable diagnostics, `AssetOpsBench-Live` delivers reproducible, real-time insights reflecting real-world industrial constraints.

Demo — https://tinyurl.com/ypv98e85

Introduction

Benchmarking of agentic AI is shifting from model-centric accuracy to end-to-end, domain-specific evaluation (Patel et al. 2025; Jha et al. 2025), which requires reasoning, planning, tool use, task coordination, and privacy preservation. Recent efforts such as AgentBench (Liu et al. 2023), AgentRewardBench (Lù et al. 2025), MultiAgentBench (Zhu et al. 2025), and REALM-Bench (Geng and Chang 2025) highlight this trend, yet most benchmarks are limited to planning modules (Shen et al. 2024) and trajectory analysis tool (Desmond et al. 2025). They lack infrastructure for code-oriented, reproducible evaluation in applied domains particularly industrial asset management, where privacy constraints limit data sharing.

Competitions complement benchmarks by engaging participants in iterative improvement. Existing competition platforms like Kaggle often only return scores, leaving participants uncertain about why their systems failed. Empirical evidence indicates that the absence of actionable feedback reduces participant engagement and increases dropout, underscoring the necessity for benchmarking platforms to provide iterative, human-in-the-loop feedback. A central design goal of `AssetOpsBench-Live` is therefore to combine competition-based engagement with actionable diagnostics.

Motivation. One such diagnostic comes from failure-mode analysis. As reported in our `AssetOpsBench` paper, nearly 10% of multi-agent task failures stem from missing clarifying questions. As a developer, we traced this mechanism in our open-source agent (`ReActXen`(Rayfield et al. 2025)) and added support for *clarifying interactions* (`enable_agent_ask=True`), yielding substantial gains:

Model	Before	After
LLaMA-4 Maverick	59%	66% (+7)
LLaMA-3-405B	44%	61% (+17)

This resulted in a **leaderboard-topping agent**, illustrating how actionable feedback can translate into concrete, code-level improvements. Building on this principle, we introduce `AssetOpsBench-Live`[1], which integrates `AssetOpsBench` with `Codabench` (Xu et al. 2022) to enable privacy-aware evaluation, automated failure-mode discovery by extending static taxonomy (Cemri et al. 2025), and structured feedback loops. Our system supports simulated multi-agent environments for local Docker-based testing and debugging, leverages scalable distributed backends capable of executing long-running tasks reliably, and incorporates evaluation protocols tailored to multi-agent reasoning, collaboration, and real-world execution. It automates submission pre-validation with optional human checks and uses a multi-phase (`Bootstrap`, `Leaderboard`), multi-track (`Planning`/`Execution`) design to balance learning and application while ensuring transparent, reproducible, and trustworthy workflows.

System Architecture

Figure 1(a) illustrates the pipeline for local development, reproducible submissions, and privacy-preserving global evaluation, supporting long-running code-oriented workflows for multi-agent industrial tasks.

*All authors contributed equally

[1]https://www.codabench.org/competitions/10206/

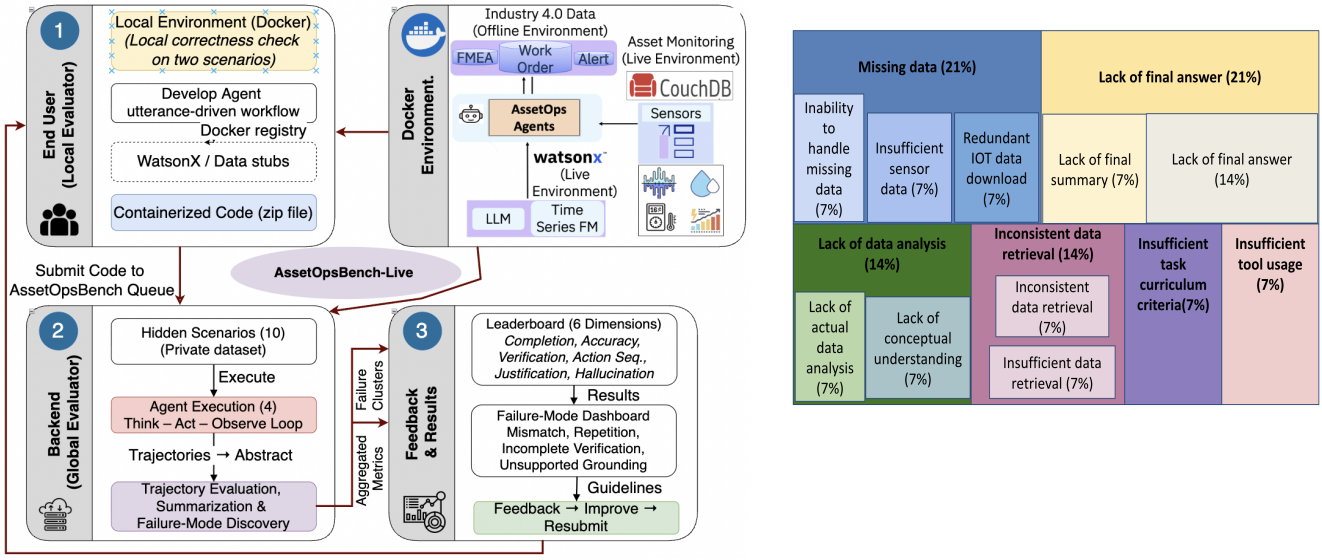

Figure 1: AssetOpsBench-Live: (a) System architecture of AssetOpsBench integrated with the Codabench, showing local development, global evaluation, and the feedback; (b) Example visualization of failure modes, clustered into interpretable categories.

First, we build a simulated environment using CouchDB and distribute it via a Docker registry, including reference agents and a small dataset. This environment offers sampled sensor telemetry, work orders, IoT streams, alerts, and failure-mode catalogs for experimentation. The simulated environment comprises: (i) an `IoT Agent` with *7* tools for sensor queries, (ii) an `FMSR Agent` with *3* tools for linking failure modes to signals, (iii) a `TSFM Agent` with *5* tools for forecasting and anomaly detection, and (iv) a `Work Order Agent` with *8* business objects for maintenance tasks. Here is a high-level description of the workflow, which consists of three key steps:

Step ①: Users build a Planning or Execution agent, run example scenarios locally, containerize it as a ZIP, and submit to Codabench for evaluation.

Step ②: Submissions are validated to ensure exactly one `.py` and one `.json` file (with required keys), then executed on one public scenario (previously runnable locally) and several hidden track-specific scenarios.

Step ③: Covers scoring, result generation, status updates, and robust error handling. *Evaluation* provides scenario-level and model-level analyses across 6 dimensions obtained using the LLM-as-Judge approach: Task Completion, Accuracy, Result Verification, Action Sequencing, Clarity, and Hallucinations. Below is an example of aggregated scores from run (#371162) evaluated on 11 hidden scenarios.

Task	Retrieval	Verification	Sequence	Clarity	Hallu.
6	6	7	8	9	2

Execution traces (trajectories) are never exposed to the user. Instead, we developed a novel algorithm that analyze these traces and dynamically generate failures that are grouped into diagnostic clusters (e.g., verification errors, hallucinations), preserving industrial confidentiality while providing interpretable insights. (Algorithm 1).

Algorithm 1: Dynamic Failure Mode Taxonomy from Traj.

Given trajectories T and initial taxonomy F_{init}
$E \leftarrow LLM\_Diagnostic(T)$
$F_{existing} \leftarrow \{e \in E \mid e \text{ matches } F_{init}\}$
$U \leftarrow \{e \in E \mid e \text{ does not match } F_{init}\}$
if $|U| > 0$ **then**
    $C_{new} \leftarrow \text{Cluster}(U)$
    $F_{ext} \leftarrow F_{init} \cup C_{new}$
else
    $F_{ext} \leftarrow F_{init}$
end if
Return F_{ext}

Users receive categorized reports indicating where and why their agent failed, without revealing any underlying data, utterances, or server-side scenarios. In Figure 1(b), we identify six failure clusters for the same run (#371162). Notably, the agent struggled the most with handling missing data, followed by issues in output validation and tool usage.

Conclusion

The system enables iterative agent improvement while safeguarding industrial confidentiality. It has been tested with 225 users across more than 300 agent evaluations. Future work will focus on extending the platform to broader scenarios and addressing the subjectivity of LLM-as-a-Judge evaluation metrics, particularly "hallucination." By bridging benchmarks, competitions, and industrial realism, `AssetOpsBench-Live` establishes a robust foundation for evaluating agentic systems.

References

Cemri, M.; Pan, M. Z.; Yang, S.; Agrawal, L. A.; Chopra, B.; Tiwari, R.; Keutzer, K.; Parameswaran, A.; Klein, D.; Ramchandran, K.; et al. 2025. Why do multi-agent llm systems fail? *arXiv preprint arXiv:2503.13657*.

Desmond, M.; Lee, J. Y.; Ibrahim, I.; Johnson, J. M.; Sil, A.; MacNair, J.; and Puri, R. 2025. Agent Trajectory Explorer: Visualizing and Providing Feedback on Agent Trajectories. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, 29634–29636.

Geng, L.; and Chang, E. Y. 2025. REALM-Bench: A Benchmark for Evaluating Multi-Agent Systems on Real-world, Dynamic Planning and Scheduling Tasks. arXiv:2502.18836.

Jha, S.; Arora, R.; Watanabe, Y.; Yanagawa, T.; Chen, Y.; Clark, J.; Bhavya, B.; Verma, M.; Kumar, H.; Kitahara, H.; et al. 2025. ITBench: Evaluating AI Agents across Diverse Real-World IT Automation Tasks. In *Proceedings of the 42nd International Conference on Machine Learning (ICML)*. https://arxiv.org/abs/2502.05352, Code: https://github.com/itbench-hub/ITBench.

Liu, X.; Yu, H.; Zhang, H.; Xu, Y.; Lei, X.; Lai, H.; Gu, Y.; Ding, H.; Men, K.; Yang, K.; Zhang, S.; Deng, X.; Zeng, A.; Du, Z.; Zhang, C.; Shen, S.; Zhang, T.; Su, Y.; Sun, H.; Huang, M.; Dong, Y.; and Tang, J. 2023. AgentBench: Evaluating LLMs as Agents. arXiv:2308.03688.

Lù, X. H.; Kazemnejad, A.; Meade, N.; Patel, A.; Shin, D.; Zambrano, A.; Stańczak, K.; Shaw, P.; Pal, C. J.; and Reddy, S. 2025. AgentRewardBench: Evaluating Automatic Evaluations of Web Agent Trajectories. arXiv:2504.08942.

Patel, D.; Lin, S.; Rayfield, J.; Zhou, N.; Vaculin, R.; Martinez, N.; O'donncha, F.; and Kalagnanam, J. 2025. AssetOpsBench: Benchmarking AI Agents for Task Automation in Industrial Asset Operations and Maintenance. *arXiv preprint arXiv:2506.03828*. Code available at https://github.com/IBM/AssetOpsBench.

Rayfield, J. T.; Lin, S.; Zhou, N.; and Patel, D. C. 2025. ReAct Meets Industrial IoT: Language Agents for Data Access. In Potdar, S.; Rojas-Barahona, L.; and Montella, S., eds., *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: Industry Track*, 364–382. Suzhou (China): Association for Computational Linguistics. ISBN 979-8-89176-333-3.

Shen, Y.; Song, K.; Tan, X.; Zhang, W.; Ren, K.; Yuan, S.; Lu, W.; Li, D.; and Zhuang, Y. 2024. TaskBench: Benchmarking Large Language Models for Task Automation. arXiv:2311.18760.

Xu, Z.; Escalera, S.; Pavão, A.; Richard, M.; Tu, W.-W.; Yao, Q.; Zhao, H.; and Guyon, I. 2022. Codabench: Flexible, easy-to-use, and reproducible meta-benchmark platform. *Patterns*, 3(7): 100543.

Zhu, K.; Du, H.; Hong, Z.; Yang, X.; Guo, S.; Wang, Z.; Wang, Z.; Qian, C.; Tang, X.; Ji, H.; and You, J. 2025. MultiAgentBench: Evaluating the Collaboration and Competition of LLM agents. arXiv:2503.01935.