복구율을 넘어: LLM 기반 에이전트 실패 대응 평가에서의 사후 검증과 보고 항목 분석

정지원, 차현묵
롯데이노베이트
Jiwon-jeong@lotte.net, hyunmook.cha@lotte.net

Beyond Recovery Rate: Post-Recovery Verification and Reporting Gaps in Failure-Aware Evaluation of LLM Agents

Jiwon Jeong, Hyunmook Cha
Lotte Innovate

요 약

대규모 언어 모델(Large Language Model, LLM) 기반 에이전트 평가는 작업 성공률을 넘어 실행 과정, 반복 신뢰성, 오류 대응·복구, 통제된 교란으로 확장되고 있다. 본 논문은 최근 실패 대응 벤치마크를 검토해 오류 감지, 대응, 사후 검증(post-recovery verification), 복구 비용(recovery cost), 정상 조건 성능의 보고 여부를 비교한다. 복구율(recovery rate)은 연구마다 분모와 산정 기준이 달라 직접 비교하기 어렵고, 대응 후 상태를 재확인하는 사후 검증은 일부 연구에서만 다루어지며 독립 지표로 일관되게 보고되지 않았다. 이에 오류 감지–대응–사후 검증과 복구 비용을 구분하는 보고 관점과 사후 검증의 예비 측정 기준을 제안한다.

Ⅰ. 서 론

대표 벤치마크인 AgentBench 는 상호작용 환경에서의 작업 수행 능력을 평가하였고 [1], AgentBoard 는 progress rate 로 중간 목표 달성도를, τ-bench 는 pass^k 로 반복 신뢰성을 평가하였다 [2, 3]. 에이전트 평가에 관한 서베이는 비용 효율성·강건성·세분화 평가를 미해결 과제로 지적하며 [4], Process Evaluation 은 최종 결과만으로 실행 과정의 오류를 포착하기 어렵다고 지적한다 [5].

이러한 흐름을 바탕으로, 본 논문은 실행 과정과 반복 신뢰성, 그리고 비적대적 오류·노이즈·환경 장애 상황에서의 대응·복구를 다룬 최근 에이전트 평가 연구를 검토한다. 본 논문에서 비적대적 조건은 정상 사용 중 발생 가능한 장애와 이를 모사한 교란을 의미하며, 프롬프트 주입 등 적대적 보안 평가는 검토 범위에서 제외한다. 핵심 주장은 세 가지다. 첫째, 실패 조건과 산정 기준이 달라 같은 복구율(recovery rate)도 직접 비교하기 어렵다. 둘째, 오류 감지와 복구 비용의 분리 보고가 드물고, 검토 범위에서는 **사후 검증(post-recovery verification)**을 별도로 평가한 사례가 제한적이다. 여기서 사후 검증은 에이전트가 대응 후 환경 상태를 다시 관찰하여 복구 여부를 확인하는 행위를 의미한다. 셋째, **오류 감지–대응–사후 검증과 복구 비용을 구분**하면 성능 저하가 발생한 단계를 보다 구체적으로 파악할 수 있다.

Ⅱ. 에이전트 평가 범위의 확장

최근 에이전트 평가는 최종 작업 성공률을 넘어 수행 과정, 반복 신뢰성, 오류 대응과 복구까지 확장되고 있다. AgentBoard 와 Process Evaluation 은 수행 과정 평가를 다루고 [2,5], τ-bench 와 ReliabilityBench 는 각각 반복 실행 신뢰성과 교란 조건에서의 신뢰성을 평가한다 [3,10]. CRITICTOOL 은 도구 오류의 감지·진단·복구를 [6], Hell or High Water (HOHW)는 함수 제거 상황의 대체 계획을 [7], Recovery-Bench 는 실패 상태에서의 복구를 평가한다 [8]. AgentErrorBench 는 모듈 및 연쇄 실패를 분석한다 [9].

AgentNoiseBench 와 ToolMaze 는 각각 노이즈 환경에서의 강건성과 도구 오류 상황에서의 복구를 평가하고 [11, 12], AgentRx 는 핵심 실패 단계를 식별한다 [13]. MAS-FIRE·AgentChaos·ToolRobustBench 는 평가 대상을 다중 에이전트, LLM API, 파이프라인 수준으로 확장한다 [14–16]. Tools Fail 은 명시적 신호가 없는 도구 오류 감지를 [18], Failing Tools 는 실행 중 도구 실패에 대한 감지·재시도·대체·상태 검증을 평가한다 [19].

Ⅲ. 평가 불일치와 보고 제안

동일한 명칭의 지표라도 산정 기준은 다르다. Recovery-Bench 는 실패 궤적에서 재구성한 손상된 시작 상태로부터 전체 작업의 최종 해결 여부를 측정하고 정상 조건과 대조한다 [8]. 반면 ToolMaze 의 교란 복구율(Perturbation Recovery Rate, PRR)은 실제로 교란된 도구 호출을 경험한 실행만을 분모로 하며, 명시적 오류와 암묵적 의미 오류 간 PRR 격차는 평균 약 37%로 보고된다 [12]. 따라서 **두 연구의 복구율은 직접 비교하기 어렵다**.

평가 대상이 다르면 수치의 의미도 달라진다. ReliabilityBench 에서 입력 교란 강도 ε 가 0 에서 0.2 로 증가할 때 pass@1 은 96.9%에서 88.1%로 감소한다 [10]. 이는 전체 성공률의 8.8%p 감소인 반면, ToolMaze 의 약 37%는 명시적 오류와 암묵적 의미 오류 간 PRR 격차를 나타낸다. 따라서 **수치의 크기보다 측정 대상과 분모를 먼저 확인할 필요가 있다**. 표 1 은 주요 연구의 평가 결과를 보고 항목별로 비교한다.

표 1. 주요 실패 대응 연구의 보고 항목 비교.

연구	오류 감지	대응	사후 검증	복구 비용	정상 조건
CRITICTOOL [6]	○	○	—	—	—
HOHW [7]	—	○	—	—	○
Recovery-Bench [8]	—	○	—	—	○
ReliabilityBench [10]	—	△	—	△	○
AgentNoiseBench [11]	—	△	—	—	○
ToolMaze [12]	△	○	—	○	○
Tools Fail [18]	○	△	—	—	—
Failing Tools [19]	○	○	△	—	—

비적대적 교란·실패 조건의 단일 에이전트 연구를 비교하였다. ○: 독립 정량 지표, △: 조건별·정성 분석/평가 기준, —: 별도 보고 없음.

오류 감지는 CRITICTOOL, Tools Fail, Failing Tools 에서 평가되지만, 표 1 의 비교 범위에서 복구 비용을 독립적으로 보고한 연구는 ToolMaze 뿐이다. 사후 검증은 Failing Tools 의 평가 기준에 포함되나 [19], 검토 범위에서는 **이를 복구율과 분리된 독립 정량 지표로 보고한 사례를 확인하지 못하였다**. Outcome Monitors 는 외부

모니터가 결과 계약 위반을 탐지하여 복구 단서를 제공하지만 [17], 이는 대응 이전의 탐지·개입으로 에이전트가 대응 이후 상태를 스스로 재확인하는 사후 검증과는 구별된다. 즉, 사후 검증은 제한적으로 다루어지고 있으며 독립적인 보고 기준으로 정착되지 않았다.

이에 본 논문은 **오류 감지-대응-사후 검증을 구분하고, 복구 비용을 별도의 효율성 지표로 보고하는 관점**을 제시한다. 오류 감지는 오류의 인지 여부를, 대응은 재시도·대체 도구·재계획·안전한 종료 등 적절한 행동의 선택 여부를, 사후 검증은 대응 이후 상태의 재확인 여부를 의미한다. 이러한 구분은 Failing Tools 의 수행 궤적 평가 항목과 상응하며 [19], 본 논문은 이를 벤치마크 간 공통 보고 기준으로 확장한다.

사후 검증의 예비 측정 기준으로는, 대응이 발생한 수행 궤적에서 **대응의 영향을 받은 상태를 후속 도구 호출이나 독립 관측으로 다시 확인하고, 그 결과가 이후 행동에 반영된 경우**를 사후 검증으로 본다. 대응과 무관한 조회나 확인 결과와 관계없이 동일한 행동이 이어지는 경우는 제외한다. 시도율은 대응 궤적 중 사후 검증을 시도한 비율로, 정확도는 검증을 시도하고 실제 상태를 판정할 수 있는 사례 중 검증 판단이 실제 상태와 일치한 비율로 정의한다. 이를 통해 검증 미수행과 검증 오판을 구분할 수 있다. 본 논문은 이러한 측정 기준의 제안에 초점을 두며, 실제 에이전트 수행 궤적을 이용한 타당성 검증은 후속 과제로 남긴다.

이러한 구분은 기존 결과의 실패 단계를 보다 구체적으로 해석하는 데에도 활용될 수 있다. ToolMaze 는 PRR 저하의 주요 원인으로 손상된 도구 출력에 대한 과신을 지목하며 [12], 이는 오류 감지 실패와 일관된다. 그러나 PRR 만으로는 오류를 감지하지 못한 경우와 오류를 감지한 뒤 적절한 대응으로 이어지지 않은 경우를 구분하기 어렵다. CRITICTOOL 은 감지와 대응을 분리하여 평가하지만 [6], 대응 이후 상태의 재확인은 평가하지 않는다. 따라서 **최종 복구율만으로는 성능 저하가 발생한 단계를 구체적으로 파악하기 어렵다**.

요컨대 본 논문은 복구율과 함께 오류 감지·대응·사후 검증을 구분하여 보고하고, 복구 비용과 가능한 경우 정상 조건 성능도 함께 제시할 것을 제안한다. 이는 여러 벤치마크를 하나의 점수로 통합하기 위한 것이 아니라, 성능 저하가 발생한 단계를 해석하기 위한 보고 관점이다.

Ⅳ. 한계와 향후 과제

첫째, 실패는 단일 계층에 머물지 않고 여러 층위로 전파될 수 있다. AgentErrorBench 는 모듈 간 연쇄 실패를 [9], MAS-FIRE 는 다중 에이전트 간 실패 전파를 [14], AgentChaos 는 LLM API 계층의 오류를 [15], ToolRobustBench 는 복합 교란의 효과를 다룬다 [16]. 따라서 계층을 가로지르는 복합 오류의 전파와 복구를 추적할 필요가 있다.

둘째, 오류 주입 조건이 실제 운영 환경을 얼마나 반영하는지 검증할 필요가 있다. AgentNoiseBench 는 현실적 노이즈를 기반으로 분류 체계를 구성하고 [11], ReliabilityBench 는 실제 서비스 환경을 모사한 스트레스 조건(production-like stress)을 지향하지만 [10], 운영 로그의 오류 유형·빈도와 주입 조건 간 정량적 대응 관계는 충분히 보고되지 않았다. 셋째, 표 1 은 8 개 대표 연구의 보고 항목을 비교한 소규모 분석으로, 체계적 문헌고찰이나 메타분석을 목적으로 하지 않는다. 또한 제안한 사후 검증 기준의 타당성과 재현성은 실제 에이전트 수행 궤적에 대한 적용을 통해 검증할 필요가 있다.

Ⅴ. 결 론

LLM 기반 에이전트 평가는 작업 성공률을 넘어 수행 과정, 반복 신뢰성, 오류 진단·복구, 통제된 오류 주입으로 확장되었다. 그러나 실패 조건과 산정 기준이 달라 동일한 복구율도 직접 비교하기 어렵고, 검토한 연구 중 사후 검증을 별도로 평가한 사례는 제한적이다. 오류 감지·대응·사후 검증을 구분하고 복구 비용을 별도로 보고하면 성능 저하가 발생한 단계를 해석하는 데 도움이 될 수 있다.

ACKNOWLEDGMENT

본 연구는 과학기술정보통신부 및 정보통신기획평가원의 생성 AI 선도인재양성사업 연구 결과로 수행되었음(IITP-2026-RS-2026-25544188)

참 고 문 헌

[1] X. Liu et al., "AgentBench: Evaluating LLMs as Agents," ICLR, 2024.

[2] C. Ma et al., "AgentBoard: An Analytical Evaluation Board of Multi-turn LLM Agents," NeurIPS 37, 2024.

[3] S. Yao et al., "τ-bench: A Benchmark for Tool-Agent-User Interaction in Real-World Domains," ICLR, 2025.

[4] A. Yehudai et al., "A Survey on Evaluation of LLM-based Agents," Findings of ACL, 2026.

[5] M. Gritta et al., "Process Evaluation for Agentic Systems," Findings of EACL, 2026.

[6] S. Huang et al., "CRITICTOOL: Evaluating Self-Critique Capabilities of LLMs in Tool-Calling Error Scenarios," EMNLP, 2025.

[7] A. Wang et al., "Hell or High Water: Evaluating Agentic Recovery from External Failures," COLM, 2025.

[8] S. Tan et al., "Recovery-Bench: Evaluating Agentic Recovery from Mistakes," NeurIPS Workshop, 2025.

[9] K. Zhu et al., "Where LLM Agents Fail and How They can Learn From Failures," arXiv:2509.25370, 2025.

[10] A. Gupta, "ReliabilityBench: Evaluating LLM Agent Reliability Under Production-Like Stress Conditions," arXiv:2601.06112, 2026.

[11] R. Wang et al., "AgentNoiseBench: Benchmarking Robustness of Tool-Using LLM Agents Under Noisy Condition," ICML, 2026.

[12] D. Zhu et al., "When Tools Fail: Benchmarking Dynamic Replanning and Anomaly Recovery in LLM Agents," arXiv:2606.05806, 2026.

[13] S. Barke et al., "AgentRx: Diagnosing AI Agent Failures from Execution Trajectories," arXiv:2602.02475, 2026.

[14] J. Jia et al., "MAS-FIRE: Fault Injection and Reliability Evaluation for LLM-Based Multi-Agent Systems," arXiv:2602.19843, 2026.

[15] G. Tan et al., "AgentChaos: Chaos Engineering for Agent Systems via Programmatic Fault Injection," ASE, 2026.

[16] Y. Zheng et al., "ToolRobustBench: Stage-Wise Perturbation Evaluation and Failure Diagnosis for Tool-Calling Agents," arXiv:2608.23635, 2026.

[17] S. Panthi and R. Abdelfattah, "Outcome Monitors: Recovery Affordances for Silent Tool Failures," arXiv:2608.19303, 2026.

[18] J. Sun, S. Y. Min, Y. Chang, and Y. Bisk, "Tools Fail: Detecting Silent Errors in Faulty Tools," EMNLP, 2024.

[19] P. Huang, H. Mao, S. Osorio, J. Wang, S. Vuddanti, Y. Agarwal, D. Waqas, and Y. Cai, "Failing Tools: Benchmarking LLM Agent Recovery Under Runtime Tool Failures," NeurIPS, 2026.