Rethinking Failure Attribution in Multi-Agent Systems: A Multi-Perspective Benchmark and Evaluation

Anonymous ACL submission

Abstract

Existing benchmarks and methods for MAS failure attribution largely assume a single deterministic root cause for each execution failure. We challenge this assumption, arguing that MAS failures often admit multiple plausible attributions depending on an observer's analytical perspective. To address this, we propose *multi-perspective failure attribution* and introduce MP-BENCH, the first benchmark and evaluation protocol designed for this setting. Through extensive experiments, we show that the perceived limitations of LLMs in failure attribution stem not from insufficient model capability, but from evaluation assumptions that fail to account for the inherently multi-perspective nature of MAS failures.

1 Introduction

Multi-agent systems (MAS) enable multiple LLM-based agents to collaborate through complex interactions to solve tasks across diverse domains (Ghafarollahi and Buehler, 2025; Zhang et al., 2024). MAS development follows an iterative refinement cycle: systems are evaluated against benchmarks, failures are attributed to specific components, and the system is revised accordingly.

Failure attribution, which aims to identify the failure-inducing steps from execution logs, is therefore a critical bottleneck in this cycle. However, manual step-by-step inspection is labor-intensive and time-consuming, prompting recent studies to explore automated approaches (Zhang et al., 2025a,b; Kong et al., 2025; Ma et al., 2025; West et al., 2025). A pioneering effort is Who&When (Zhang et al.), the first benchmark for failure attribution in MAS, which labels each execution with a single, deterministic failure-inducing step for the task failure. Subsequent studies adopt this same formulation, inheriting the assumption that failure attribution can be uniquely determined (Zhang et al., 2025a; Kong et al., 2025).

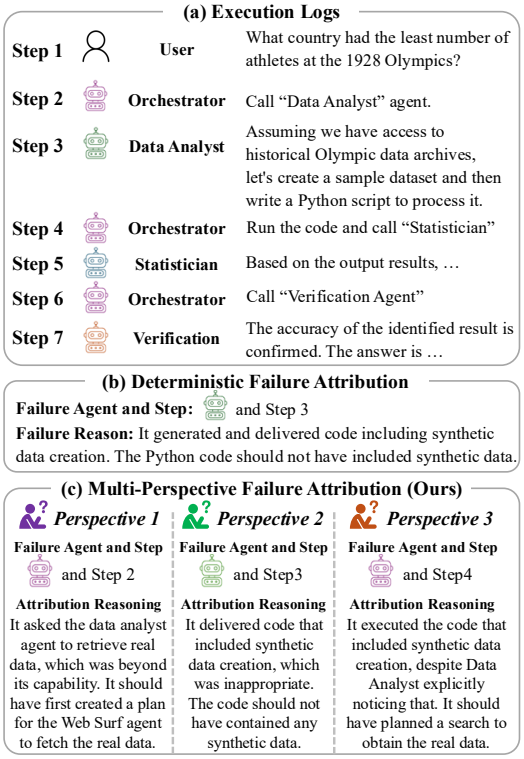

Figure 1: Motivating examples illustrating the multi-perspective nature of MAS failure attribution.

However, we should rethink: *Is the failure-inducing step for a MAS failure deterministically decided?* As illustrated in Figure 1(b), the existing works implicitly assume that the oracle of failure-inducing steps is deterministic, thereby only one failure step is decided during annotation process. In practice, the failure-inducing step of a MAS failure is often **not deterministic and admits multiple plausible attributions**. As shown in Figure 1(c), one analyst expects a web-first strategy, while another considers a code-first approach acceptable—both perspectives being reasonable.

Nevertheless, existing works largely overlook this multi-perspective nature, limiting their practical applicability. Since current benchmarks are built upon the assumption that the oracle of a failure-inducing step is deterministic, they often

misjudge reasonable predictions as incorrect, undermining evaluation reliability. Furthermore, models built on the assumption inherit biases that fail to capture diverse causal patterns, hindering effective attribution of MAS failure.

To address these limitations, we propose a more practical paradigm: ***multi-perspective failure attribution*** for MAS. In contrast to prior work that frames failure attribution as a rigid, deterministic labeling task, the multi-perspective failure attribution system identifies failure-inducing steps from multiple perspectives, each accompanied by explicit internal reasoning that explains the system's judgment from that viewpoint. This new paradigm necessitates a corresponding benchmark dataset and evaluation protocol.

To this end, we introduce a new benchmark, **m**ulti-**p**erspective failure attribution **bench**mark (MP-BENCH), consisting of execution logs collected from diverse tasks and MAS configurations. Each log is annotated by multiple expert annotators who are carefully screened. The annotations include the failure-inducing step, and explicit internal reasoning that justifies the attribution from each annotator's perspective. This expert-driven annotation process provides substantially stronger reliability than prior benchmarks, which largely rely on automated pipelines. We further design an evaluation protocol to assess whether current LLMs can perform multi-perspective failure attribution on MP-BENCH. The evaluation focuses on two core capabilities: (1) the ability to identify failure-inducing steps from multiple perspectives, and (2) the ability to provide reliable and well-grounded reasoning to justify these attributions.

Through extensive experiments with state-of-the-art LLMs, we show that when sampled multiple times, LLMs effectively identify diverse failure attributions that correspond to different perspectives. Under our evaluation protocol, these attributions and the associated reasoning align closely with human expert's internal reasoning. Notably, these findings contrast sharply with prior results reported in deterministic benchmarks such as Who&When, where LLM-based step-level failure attribution was shown to perform near random. Together, our results demonstrate that the limitations of LLMs in failure attribution observed in existing works stem not from insufficient model capability, but from biased evaluation assumptions that overlook the multi-perspective nature of MAS failures. Based on our findings, we provide practical guidance for real-world MAS failure attribution. The key contributions of this work are as follows:

- We identify a fundamental assumption in prior work that limits the practicality of MAS failure attribution and propose a more realistic task formulation, *multi-perspective failure attribution*.
- We introduce a new benchmark and evaluation protocol specifically designed for multi-perspective failure attribution in MAS.
- We show that the prevailing belief that current LLMs perform poorly at failure attribution arises from flawed deterministic assumptions in existing benchmarks, rather than from inherent limitations of the models themselves.

2 Related Works

2.1 LLM-based Multi-Agent Systems

Multi-agent systems built on LLMs have emerged as an effective paradigm for solving complex tasks through collaboration among multiple specialized agents (Hong et al., 2023; Wu et al., 2024; Zhang et al., 2024; Song et al., 2024; Ye et al., 2025). While such systems benefit from modularity and role specialization, their intricate inter-agent dependencies introduce reliability issues: errors propagate across agents and failures arise from complex interactions (Cemri et al., 2025; Zhang et al.), becoming more pronounced in long-horizon settings. These reliability challenges motivate the need for systematic failure attribution in MAS, which is the focus of this work.

2.2 Benchmark of Failure Attribution in MAS

Several benchmarks have been proposed to evaluate failure attribution systems in MAS, each with notable limitations. MAST (Cemri et al., 2025) operates at the execution level and cannot assess step-level attribution. TracerTraj (Zhang et al., 2025a) and Aegis (Kong et al., 2025) achieve finer-grained step- and agent-level attribution, but rely on fully automated construction pipelines, raising reliability concerns given the complexity of MAS logs and causal failure patterns. Who&When (Zhang et al.) improves annotation quality through expert annotators, yet all of these benchmarks share a common limitation: they assume a deterministic oracle, selecting a single failure-inducing step per execution. This conflicts with real-world MAS failures where multiple plausible attributions coexist.

Furthermore, none supports evaluating the reasoning behind an attribution, which is essential for systematic diagnosis. We address these gaps by introducing the first multi-perspective failure attribution benchmark, constructed through rigorous expert annotation and accompanied by an evaluation protocol that assesses both attribution accuracy and reasoning quality (see Table 1).

Table 1: Comparison between MP-BENCH and others.

	Attribution Granularity	Annotation Method	Attribution Reasoning Eval	Perspective	Size
Cemri et al. (2025)	Execution	Hybrid	✗	Single	1,600
Zhang et al.	Step	Expert	✗	Single	184
Zhang et al. (2025a)	Step	Automated	✗	Single	2,500
Kong et al. (2025)	Agent	Automated	✗	Single	9,500
MP-BENCH	Step	Expert	✓	Multi	289

2.3 Annotator Disagreement

Recent literature has increasingly adopted a perspectivist approach that treats annotator disagreement as a meaningful signal rather than noise (Fleisig et al., 2024; Wan et al., 2025), with various modeling strategies proposed to preserve and leverage diverse viewpoints (Dehghan et al., 2025; Xu et al., 2025; Homayounirad et al., 2025). However, these studies have largely focused on socially subjective tasks—such as hate speech or offensive content detection (Dehghan et al., 2025; Riabi et al., 2025; Orlikowski et al., 2025; Fleisig et al., 2023)—where disagreements stem from personal, cultural, or normative values. In contrast, we investigate annotator disagreement in the context of MAS failure attribution, a setting where disagreements arise from differing technical interpretations of complex execution traces rather than subjective values. To our knowledge, this is the first work to apply a perspectivist framework to a MAS failure attribution.

3 Task Formulation

3.1 Preliminaries

Multi-Agent Systems. We adopt a widely used turn-based LLM multi-agent protocol (Hong et al., 2023; Li et al., 2023; Wu et al., 2024; Zhang et al.). We consider an LLM-powered MAS $\mathcal{M} = \langle \mathcal{N}, S, A, P, \phi \rangle$ consisting of N agents $\mathcal{N} = \{1, \ldots, N\}$ operating in discrete time steps, where S is the set of environment states, A is the global action space, $A_i \subseteq A$ is the allowable actions for agent i, and $\phi(t)$ specifies the active agent at step t. State transitions follow $P(s_{t+1} \mid s_t, a_t, \phi(t))$, and an execution is represented as a finite trajectory $\tau = \{(s_1, a_1, \phi(1)), \ldots, (s_T, a_T, \phi(T))\}$.

Deterministic Failure Attribution. Existing failure attribution works assume that, for each failed execution τ, there exists a deterministic set of failure-inducing steps that can be uniquely identified. Under this assumption, failure attribution is formulated as a mapping $f_{\text{det}}(\tau) \rightarrow \mathcal{F}^*$, where $\mathcal{F}^* \subseteq \{1, \ldots, T\}$ denotes a fixed set of time steps whose associated agent actions $(a_t, \phi(t))$ are deemed responsible for the failure. Most benchmarks further simplify this by assuming $|\mathcal{F}^*| = 1$ (Zhang et al., 2025a).

3.2 Multi-Perspective Failure Attribution

Definition. We argue that MAS failure attribution is inherently non-deterministic: due to the existence of multiple valid execution trajectories and diverse expectations of a correct strategy, a failed execution may admit multiple plausible attributions, each grounded in a distinct analytical perspective. We define *multi-perspective failure attribution* as the task of identifying a set of plausible failure-inducing steps together with their associated rationales, formally $\mathcal{F}(\tau) = \{(t_k, r_k)\}_{k=1}^{K}$, where t_k denotes a failure-inducing step and r_k is an explicit rationale justifying that attribution under perspective k. Multiple attributions can be simultaneously valid even when they implicate different steps and agents.

Task Objective. Given a failed trace τ, the goal is to predict $\widehat{\mathcal{F}}(\tau)$ that aligns with human expert judgments—success is defined not by matching a single oracle label, but by recovering multiple reasonable attributions with well-grounded rationales.

Scope. This work focuses on offline, post-hoc attribution from completed execution logs, a common and practical scenario in iterative MAS development (Zhang et al., 2025a; Kong et al., 2025; Cemri et al., 2025; Kim et al., 2026). We do not address online debugging, dynamic intervention, or real-time repair during execution, which constitute a complementary but distinct problem.

4 MP-BENCH

4.1 Dataset Construction

The overall procedure is summarized in Fig 2(a).

4.1.1 MAS Execution Log Collection

To ensure diversity in system design and execution behavior, following (Zhang et al.), we include both

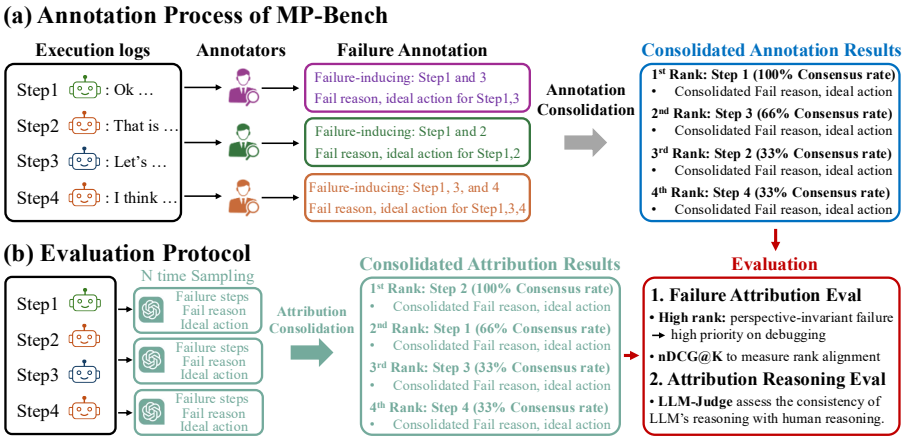

Figure 2: Overall framework of the (a) annotation process and (b) evaluation protocol of MP-BENCH.

hand-crafted and fully-automated MAS configurations. Specifically, we collect 289 logs from a total of 121 distinct MAS configurations and various tasks. For hand-crafted MAS, we use MAgenticOne (Fourney et al., 2024). For fully-automated MAS, we use CaptainAgent (Song et al., 2024). For tasks, we use GAIA (Mialon et al., 2023) and AssistantBench (Yoran et al., 2024), which are widely utilized to evaluate the agentic capabilities of MAS.

4.1.2 Expert Annotator Recruitment

We recruit annotators on the Upwork[1] platform. To ensure high-quality and technically grounded annotations, we prioritize candidates with professional experience in LLMs, AI systems, or agent-based frameworks. We adopt a rigorous two-stage screening process. In the first stage, we conduct technical interviews to assess candidates' conceptual understanding of the task. Applicants are asked to explain, in their own words, what MAS failure attribution entails, and to describe how they would approach annotating a turn-based MAS execution log. In the second stage, we assess annotation quality using test cases. Specifically, we provide several MAS execution logs and ask candidates to annotate them according to a detailed predefined task description (see Figure 6 in Appendix). The authors manually review the submitted annotations, focusing on correctness, consistency, and the clarity of reasoning. Three candidates who pass both stages are selected.

4.1.3 Annotation Task

For each execution log, annotators are provided with the complete MAS execution trace and task description. Each trace consists of a sequence of agent actions following a turn-based protocol. Annotators independently annotate every step, assigning the following for each:

- **Failure-inducing Step:** a binary label indicating whether the step is failure-inducing.
- **Failure Reason:** a free-text explanation justifying the failure annotation.
- **Ideal Action:** a free-text description of the action that the agent should ideally have taken.

Critically, we provide no potential bias or guiding perspectives to annotators, allowing them to develop independent interpretations based on their expertise. This design genuinely captures the multi-perspective nature of MAS failure attribution, as annotators may reasonably disagree on which steps are failure-inducing based on different assumptions about correct execution strategies. The *failure reason* and *ideal action* fields explicitly capture each annotator's internal reasoning, grounding why a particular agent action is deemed problematic from their perspective. This enables MP-BENCH to evaluate failure attribution systems on two dimensions: (1) the ability to identify failure-inducing steps across multiple perspectives, and (2) the ability to generate well-grounded reasoning that aligns with expert judgment. This annotation framework moves beyond existing benchmarks that treat failure attribution as deterministic classification, instead assessing both attribution accuracy and reasoning quality.

4.1.4 Annotation Noise Filtering

While steps labeled as failure-inducing by multiple annotators are generally more reliable, single-annotator labels may sometimes reflect misunderstanding, insufficient context, or annotator bias rather than legitimate analytical perspectives. To

[1] https://www.upwork.com/

distinguish genuine multi-perspective disagreement from such noise, we randomly sample 200 steps labeled as failure-inducing by only a single annotator and ask an additional human reviewer to classify each as either a legitimate perspective or noise. The results indicate that 65.5% of these labels correspond to legitimate attributions, while the remaining 34.5% are attributable to annotation noise. To scale this filtering beyond a small-sample manual study, we evaluate whether an LLM (Claude-Sonnet-4.5) can reliably replicate these human judgments. The LLM achieves 92.5% agreement with the human reviewer (Cohen's κ=0.8401), demonstrating that it serves as a reliable and scalable proxy for human validation. We therefore apply this LLM-assisted filtering to the full dataset, removing steps identified as noise. In total, 31.6% of single-annotator-labeled steps are removed via this procedure.

4.1.5 Annotation Consolidation

We consolidate annotations into a ranking-based format. For each execution log, we compute a *consensus rate* for every step, defined as the fraction of annotators who labeled it as failure-inducing, and rank steps accordingly. Steps with 100% consensus represent perspective-invariant failures—the most salient and unambiguous failure points—and should be addressed with the highest priority. Steps identified by only a subset of annotators reflect perspective-dependent failure modes that warrant lower priority. Only steps labeled as failure-inducing by at least one annotator are included.

For free-text fields (failure reasons and ideal actions), we apply LLM-assisted summarization: an LLM synthesizes a unified explanation for each failure-inducing step from all annotator-provided rationales, instructed to comprehensively cover diverse viewpoints without introducing new information. We primarily use GPT-5.1 (OpenAI, 2025b) for this consolidation. To verify robustness, we additionally evaluate consolidations from GPT-4.1 (OpenAI, 2025a) and Claude-Sonnet-4.5 (Anthropic, 2025), finding that model rankings remain consistent across all three choices. Detailed results are in Appendix D.1.

4.2 Annotation Quality and Effort

MP-BENCH consists of 169 hand-crafted MAS executions (average 33 interaction steps) and 120 automatically generated executions (average 8 steps), totaling 289 instances with triple expert annotation. While this is smaller in scale than recent automated benchmarks (e.g., TracerTraj with 2,500 instances and Aegis with 9,500), those works rely on fully automated pipelines that cannot ensure annotation reliability for the complex causal reasoning inherent in MAS failures. Our quality-first approach instead employs multiple expert annotators per instance, capturing diverse perspectives and explicit reasoning for each attribution. Compared to Who&When, the closest manually curated prior work, MP-BENCH provides 57% more instances (289 vs. 184) while additionally introducing the multi-perspective paradigm. Details of annotator backgrounds and annotation effort are provided in Appendix B.

4.3 Data Analysis

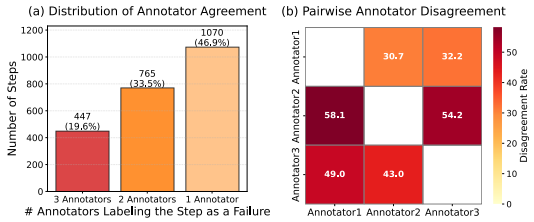

Figure 3: Analysis on annotations of MP-BENCH. (a) Distribution of steps grouped by annotator consensus on failure annotations. (b) Inter-annotator disagreement rates for failure annotations.

We analyze the annotation statistics of MP-BENCH, focusing on the degree of consensus among annotators regarding failure step annotations. As shown in Figure 3(a), only 19.6% of steps are annotated as failure-inducing by all three annotators. 33.5% of steps are labeled as failures by exactly two annotators, while a 46.9% are labeled as failures by only a single annotator. This distribution indicates that failure attribution is highly dependent on annotator perspective, underscoring the limitations of deterministic assumptions adopted by existing benchmarks. Similarly, Figure 3(b) reports inter-annotator disagreement rates, defined as cases where one annotator marks a step as failure while another does not. The results show that the disagreement can reach up to 58% across annotators, further confirming the multi-perspective nature of MAS failure attribution.

5 Evaluation Protocol

This section describes the evaluation protocol of MP-BENCH. The protocol is explicitly designed to address the following questions:

- Do LLM-based failure attribution systems capture failures from multiple perspectives? (Section 5.1)
- Do LLMs provide reliable reasoning for failure attribution? (Section 5.2)

An overview of the evaluation protocol is shown in Figure 2(b).

5.1 Failure Attribution Evaluation

Existing failure attribution benchmarks formulate evaluation as a classification task, assessing whether a system correctly identifies a single failure-inducing step. Such deterministic evaluation is insufficient in our setting, as a single execution may admit multiple plausible failure attributions depending on perspective.

To address the first question, in MP-BENCH, we adopt a ranking-based evaluation that measures the consistency between the rankings produced by LLM-based failure attribution systems and the ground-truth rankings derived from human annotations. Specifically, we run each LLM N times with a stochastic sampling, generating multiple independent failure attribution outputs. Each output consists of predicted failure-inducing steps along with their associated failure reasons, and ideal actions. Following the same consolidation procedure used for human annotations, we aggregate these outputs into a single predicted ranking based on consensus rates.

To compare rankings produced by annotators and LLMs, we employ normalized Discounted Cumulative Gain (nDCG@K). This metric evaluates agreement between the predicted and ground-truth rankings, with greater emphasis on correctly prioritizing high-consensus failure steps. Moreover, it downweights low-consensus steps, structurally bounding the influence of potential annotation noise on evaluation outcomes.

5.2 Attribution Reasoning Evaluation

In the multi-perspective failure attribution task, beyond identifying failure-inducing steps, LLMs generate explicit attribution reasoning including failure reasons and ideal actions. Evaluating this reasoning quality is essential for assessing whether LLMs can serve as reliable failure attribution systems.

Following the same procedure as annotation consolidation (Section 4.1.5), we consolidate the attribution reasoning generated from N LLM runs. We then adopt an LLM-as-a-Judge framework using GPT-5.1 as the judge model. For each failure-inducing step, the judge assesses consolidated predictions against consolidated ground truths based on: (1) consistency with human judgments, (2) grounding in execution context, (3) explanatory adequacy, and (4) reasonableness of proposed actions. The judge assigns scores (1-10) for failure reason, ideal action, and overall reasoning quality. We calculate the average score of overall reasoning quality across all steps and executions. Detailed judge prompts are provided in Figure 8 in Appendix.

To verify robustness to judge model choice, we additionally evaluate using GPT-4.1 and Claude-Sonnet-4.5 as alternative judges. We find that model rankings remain highly consistent across all three judges, confirming that our evaluation results are not sensitive to this choice. Detailed results are presented in Appendix D.1.

To further validate that our results do not depend on LLM-based evaluation, we additionally report BLEU and embedding cosine similarity scores. Full results are in Appendix D.2.

6 Experiment

6.1 Experimental Details

For failure attribution systems, we adopt the All-at-Once method (Zhang et al.) with a diverse set of LLMs. Detailed implementation and evaluation settings are provided in Appendix C.1 and Appendix C.2, respectively.

6.2 Experimental Results

From the evaluation on MP-BENCH, we derive the following key observations.

First, stochastic sampling of LLMs naturally reflects diverse perspectives in failure attribution. When sampling multiple failure attributions from LLMs, we observe substantial disagreement among the predicted failure-inducing steps (e.g., GPT-5.1 in Figure 4(a)), a pattern that closely mirrors the disagreement observed among human expert annotators. This alignment provides further evidence that failure attribution in MAS is inherently non-deterministic and admits multiple plausible perspectives, rather than a single deterministic root cause.

Second, under a multi-perspective evaluation setting, LLMs perform failure attribution effectively when sampled multiple times. As shown in Table 2, when LLMs are sampled multiple times

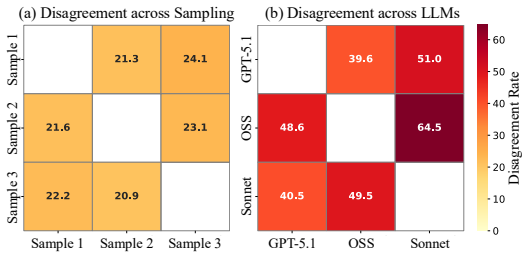

Figure 4: Disagreement analysis of the LLM-based failure attribution system. (a) Pairwise disagreement rates across different samplings of GPT-5.1. (b) Pairwise disagreement rates across different LLMs.

($N = 3$), they successfully identify multiple reasonable failure-inducing steps across different perspectives. Among the evaluated models, Claude-Sonnet-4.5 demonstrates the strongest performance across most settings, whereas Qwen3-8B exhibits limited failure attribution and reasoning capabilities, likely due to its relatively small model size. We also observe that nDCG scores are consistently higher on automatic MAS than on hand-crafted MAS. This is largely attributable to the difference in log length: hand-crafted logs average 33 steps, whereas automatic logs average only 8, meaning models face a smaller candidate pool and are more likely to rank failure-inducing steps highly by chance. This also explains why the gap between nDCG@5 and nDCG@full is narrower in the automatic setting.

Notably, reasoning-oriented or more recent models do not necessarily guarantee superior performance: for instance, GPT-5.1 scores markedly lower nDCG than GPT-4.1 on hand-crafted MAS, and oss-120B, while competitive on hand-crafted configurations, shows a notable drop on automatic MAS. That said, the resulting nDCG scores from all models are substantially higher than those of a random baseline. These results stand in contrast to prior findings from deterministic benchmarks such as Who&When (Zhang et al.), which suggested that LLMs struggle with step-level failure attribution by exhibiting near-random performance.

Third, LLM-generated failure attributions are accompanied by reliable and well-grounded reasoning. In Table 2, we observe high LLM-as-a-Judge scores when comparing the attribution reasoning produced by LLMs with that of human annotators. This indicates that LLM-generated multi-perspective attributions are not only plausible in terms of identified failure steps, but are also supported by reasoning that aligns well with human expert judgments. Consequently, these attributions can serve as meaningful diagnostic signals for MAS developers.

Table 2: Failure attribution (nDCG) and attribution reasoning (Reason. Score) performance comparison. Mean and standard deviation are computed over three runs.

	Hand-Crafted MAS			Automatic MAS		
	nDCG		Reason.	nDCG		Reason.
Model	@5	@full	Score	@5	@full	Score
Random	0.128	0.148	-	0.315	0.321	-
GPT-4.1	0.445 ±0.011	0.487 ±0.007	7.90 ±0.03	0.679 ±0.007	0.682 ±0.008	7.95 ±0.08
GPT-5.1	0.385 ±0.008	0.464 ±0.008	7.94 ±0.04	0.769 ±0.0124	0.772 ±0.010	8.30 ±0.02
Sonnet-4.5	0.432 ±0.004	0.498 ±0.004	8.09 ±0.07	0.788 ±0.0101	0.794 ±0.011	8.34 ±0.10
Qwen3-8B	0.317 ±0.016	0.366 ±0.019	6.26 ±0.09	0.676 ±0.0205	0.689 ±0.019	6.62 ±0.04
oss-120B	0.449 ±0.011	0.507 ±0.007	7.74 ±0.08	0.696 ±0.0143	0.706 ±0.015	8.09 ±0.11

Overall, our results challenge the prevailing assumption that MAS failure attribution is deterministic and that in-context LLMs are ineffective at this task. Instead, we show that MAS failure attribution exhibits a multi-perspective nature, and that LLMs possess sufficient capability to capture this structure when evaluated under an appropriate benchmark and evaluation protocol. The limitations reported in prior work therefore stem not from deficiencies in model capability, but from biased evaluation assumptions that overlook the multi-perspective nature of MAS failures.

6.2.1 Analyses on Number of LLM Sampling

Table 3: Performance comparison (nDCG@5) across different number of LLM runs N. Bold values indicate the best performance within each model.

	Hand-Crafted (N)			**Automatic (N)**		
	3	**5**	**7**	**3**	**5**	**7**
GPT-4.1	0.445 ±0.011	0.444 ±0.008	**0.447 ±0.001**	0.679 ±0.007	0.709 ±0.006	**0.711 ±0.005**
GPT-5.1	0.385 ±0.008	0.412 ±0.003	**0.436 ±0.004**	0.769 ±0.012	**0.796 ±0.001**	0.794 ±0.003
Sonnet-4.5	0.432 ±0.004	0.453 ±0.005	**0.464 ±0.002**	0.788 ±0.010	0.793 ±0.005	**0.802 ±0.004**
Qwen3-8B	0.317 ±0.016	**0.339 ±0.003**	**0.339 ±0.007**	0.676 ±0.020	0.694 ±0.015	**0.709 ±0.002**
oss-120B	0.449 ±0.011	0.464 ±0.008	**0.468 ±0.003**	0.696 ±0.014	0.719 ±0.003	**0.721 ±0.007**

We examine the impact of the number of LLM runs (N) on failure attribution performance by varying N from 3 to 7 and reporting nDCG@5 for different LLMs. As shown in Table 3, increasing N beyond 3 generally improves performance, but the gains are not monotonic. For some models, performance improves noticeably up to $N = 7$, while

others saturate earlier: certain models plateau at $N = 5$, and in the automatic MAS setting, saturation at $N = 5$ is the more common pattern across models. These results suggest that while a minimum level of sampling diversity is important, blindly increasing N does not guarantee proportional gains. Practitioners therefore balance attribution coverage against computational cost, rather than defaulting to the largest feasible N.

6.2.2 Analyses on Multi-LLM Combination

Table 4: Performance comparison between single-LLM and multi-LLM failure attribution systems. We denote GPT-4.1, GPT-5.1, GPT-OSS-120B, and Claude-Sonnet-4.5 as G-4.1, G-5.1, OSS, and Sonnet, respectively.

Model Combination				nDCG@5	
G-4.1	G-5.1	OSS	Sonnet	Hand.	Auto.
✓	✗	✗	✗	0.445 ±0.011	0.679 ±0.007
✗	✓	✗	✗	0.385 ±0.008	0.769 ±0.012
✗	✗	✓	✗	0.449 ±0.011	0.696 ±0.014
✗	✗	✗	✓	0.432 ±0.004	0.794 ±0.011
✓	✓	✓	✗	0.480 ±0.009	0.789 ±0.011
✓	✓	✗	✓	0.482 ±0.010	0.811 ±0.009
✓	✗	✓	✓	0.493 ±0.013	0.795 ±0.004
✗	✓	✓	✓	0.491 ±0.003	0.810 ±0.012

We study whether combining multiple LLMs improves failure attribution performance, motivated by the hypothesis that different models encode distinct reasoning patterns and thus offer complementary attribution perspectives. We compare single-LLM systems (each sampled $N = 3$ times) with multi-LLM systems in which each model is sampled once, constructed from GPT-4.1, GPT-5.1, oss-120B, and Claude-Sonnet-4.5. All results are evaluated using nDCG@5.

As shown in Table 4, multi-LLM systems consistently outperform their single-LLM counterparts. The largest gains arise when combining models from different families: mixtures of Claude, GPT-based, and OSS models achieve the highest attribution performance, suggesting that cross-family diversity is particularly effective.

This is further supported by the disagreement analysis in Figure 4: pairwise disagreement across repeated samplings of the same LLM is substantially lower than that observed across different LLMs, indicating that repeated sampling captures primarily stochastic variation, whereas combining distinct models introduces qualitatively different attribution perspectives. The performance gains of multi-LLM systems thus stem from complementary reasoning patterns across models, rather than from increased sampling alone. We additionally find that increasing N per model in the multi-LLM setting yields further improvements, with detailed results provided in Appendix D.3.

6.3 Practical Guidance for Practitioners

Based on our experimental findings, we provide practical guidance for designing effective and reliable automatic failure attribution systems for MAS.

First, MAS failure attribution is inherently multi-perspective. Rather than relying on a single deterministic attribution result, practitioners should emphasize aggregating attributions from multiple perspectives, which enables capturing diverse causal patterns and supports more effective MAS diagnosis and refinement.

Second, combining LLMs from different model families is strongly recommended. Models from different families tend to provide complementary perspectives, and aggregating their attributions leads to more informative and robust failure diagnosis.

Third, when higher reliability is required, increasing the number of samples could be beneficial. Aggregating a larger set of attribution results enables more stable and accurate failure attribution.

7 Conclusion

We revisit failure attribution in MAS and show that deterministic formulations fail to capture the inherently multi-perspective nature of real-world MAS failures. To address this limitation, we propose multi-perspective failure attribution and introduced MP-BENCH, the first benchmark and evaluation framework designed for this setting. MP-BENCH combines expert annotations and multi-perspective evaluation to enable faithful assessment of failure attribution quality and reasoning. Our experiments reveal that prior claims about LLMs' limitations in failure attribution largely stem from restrictive benchmark assumptions, and that modern LLMs can produce diverse, human-aligned attributions when evaluated appropriately. Overall, our work underscores the need to rethink failure attribution benchmarks and evaluation practices, and provides a practical foundation for reliable MAS failure attribution.

Limitations and Future Work

While MP-BENCH establishes the first rigorous benchmark for multi-perspective failure attribution, it also has several limitations that suggest directions for future work.

Domain Coverage. Our benchmark currently focuses on general-purpose assistant tasks. Extending the benchmark to more specialized domains, such as scientific research, software engineering, and creative tasks, would broaden its applicability and enable a more comprehensive evaluation of failure attribution methods.

Scale and Framework Diversity. Although our quality-first approach ensures reliable and high-fidelity annotations, it limits the scale of the benchmark and the diversity of covered multi-agent system frameworks. Future work could investigate hybrid expert–automated annotation pipelines to improve scalability while maintaining annotation quality, as well as incorporate a wider range of MAS frameworks.

Downstream Debugging Utility. MP-BENCH is a meta-benchmark for evaluating failure attribution systems, not a debugging system itself. Our contributions concern evaluation design: we demonstrate that multi-perspective attribution yields more principled and informative evaluation signals than deterministic formulations. Empirically validating whether these signals lead to measurable improvements in developer efficiency or downstream system performance—e.g., through user studies or closed-loop MAS refinement experiments—is an important direction that we leave for future work.

Despite these limitations, MP-BENCH provides a strong gold-standard foundation for multi-perspective failure attribution research and serves as a valuable resource for advancing this line of work.

Ethics Statement

This dataset is constructed exclusively using publicly available tasks and publicly accessible multi-agent systems. It does not contain any personally identifiable information. To the best of our knowledge, the dataset does not raise any ethical concerns.

References

Sandhini Agarwal, Lama Ahmad, Jason Ai, Sam Altman, Andy Applebaum, Edwin Arbus, Rahul K Arora, Yu Bai, Bowen Baker, Haiming Bao, and 1 others. 2025. gpt-oss-120b & gpt-oss-20b model card. *arXiv preprint arXiv:2508.10925*.

Anthropic. Anthropic api. https://www.anthropic.com/api/.

Anthropic. 2025. Introducing claude sonnet 4.5.

Mert Cemri, Melissa Z Pan, Shuyi Yang, Lakshya A Agrawal, Bhavya Chopra, Rishabh Tiwari, Kurt Keutzer, Aditya Parameswaran, Dan Klein, Kannan Ramchandran, and 1 others. 2025. Why do multi-agent llm systems fail? *arXiv preprint arXiv:2503.13657*.

Somaiyeh Dehghan, Mehmet Umut Sen, and Berrin Yanikoglu. 2025. Dealing with annotator disagreement in hate speech classification. *arXiv preprint arXiv:2502.08266*.

Eve Fleisig, Rediet Abebe, and Dan Klein. 2023. When the majority is wrong: Modeling annotator disagreement for subjective tasks. *arXiv preprint arXiv:2305.06626*.

Eve Fleisig, Su Lin Blodgett, Dan Klein, and Zeerak Talat. 2024. The perspectivist paradigm shift: Assumptions and challenges of capturing human labels. *arXiv preprint arXiv:2405.05860*.

Adam Fourney, Gagan Bansal, Hussein Mozannar, Cheng Tan, Eduardo Salinas, Friederike Niedtner, Grace Proebsting, Griffin Bassman, Jack Gerrits, Jacob Alber, and 1 others. 2024. Magentic-one: A generalist multi-agent system for solving complex tasks. *arXiv preprint arXiv:2411.04468*.

Alireza Ghafarollahi and Markus J Buehler. 2025. Sciagents: automating scientific discovery through bioinspired multi-agent intelligent graph reasoning. *Advanced Materials*, 37(22):2413523.

Amir Homayounirad, Enrico Liscio, Tong Wang, Catholijn M Jonker, and Luciano C Siebert. 2025. Will annotators disagree? identifying subjectivity in value-laden arguments. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 15237–15252. Association for Computational Linguistics.

Sirui Hong, Mingchen Zhuge, Jonathan Chen, Xiawu Zheng, Yuheng Cheng, Jinlin Wang, Ceyao Zhang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, and 1 others. 2023. Metagpt: Meta programming for a multi-agent collaborative framework. In *The Twelfth International Conference on Learning Representations*.

Wonjoong Kim, Sangwu Park, Yeonjun In, Sein Kim, Dongha Lee, and Chanyoung Park. 2026. Beyond the final answer: Evaluating the reasoning trajectories of tool-augmented agents. *Preprint*, arXiv:2510.02837.

Fanqi Kong, Ruijie Zhang, Huaxiao Yin, Guibin Zhang, Xiaofei Zhang, Ziang Chen, Zhaowei Zhang, Xiaoyuan Zhang, Song-Chun Zhu, and Xue Feng. 2025. Aegis: Automated error generation and attribution for multi-agent systems. *arXiv preprint arXiv:2509.14295*.

Guohao Li, Hasan Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. 2023. Camel: Communicative agents for" mind" exploration of large language model society. *Advances in Neural Information Processing Systems*, 36:51991–52008.

Guoqing Ma, Jia Zhu, Hanghui Guo, Weijie Shi, Jiawei Shen, Jingjiang Liu, and Yidan Liang. 2025. Automatic failure attribution and critical step prediction method for multi-agent systems based on causal inference. *arXiv preprint arXiv:2509.08682*.

Grégoire Mialon, Clémentine Fourrier, Thomas Wolf, Yann LeCun, and Thomas Scialom. 2023. Gaia: a benchmark for general ai assistants. In *The Twelfth International Conference on Learning Representations*.

OpenAI. Openai api. https://openai.com/index/openai-api/.

OpenAI. 2025a. Introducing gpt-4.1 in the api.

OpenAI. 2025b. Introducing gpt-5.1 in the api.

Matthias Orlikowski, Jiaxin Pei, Paul Röttger, Philipp Cimiano, David Jurgens, and Dirk Hovy. 2025. Beyond demographics: Fine-tuning large language models to predict individuals' subjective text perceptions. *arXiv preprint arXiv:2502.20897*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Arij Riabi, Virginie Mouilleron, Menel Mahamdi, Wissam Antoun, and Djamé Seddah. 2025. Beyond dataset creation: critical view of annotation variation and bias probing of a dataset for online radical content detection. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 8640–8663.

Linxin Song, Jiale Liu, Jieyu Zhang, Shaokun Zhang, Ao Luo, Shijian Wang, Qingyun Wu, and Chi Wang. 2024. Adaptive in-conversation team building for language model agents. *arXiv preprint arXiv:2405.19425*.

TogetherAI. Togetherai api. https://www.together.ai/.

Ruyuan Wan, Haonan Wang, Ting-Hao Huang, and Jie Gao. 2025. From noise to nuance: Enriching subjective data annotation through qualitative analysis. In *Proceedings of the Fourth Workshop on Bridging Human-Computer Interaction and Natural Language Processing (HCI+ NLP)*, pages 240–254.

Alva West, Yixuan Weng, Minjun Zhu, Zhen Lin, Zhiyuan Ning, and Yue Zhang. 2025. Abduct, act, predict: Scaffolding causal inference for automated failure attribution in multi-agent systems. *arXiv preprint arXiv:2509.10401*.

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, and 1 others. 2024. Autogen: Enabling next-gen llm applications via multi-agent conversations. In *First Conference on Language Modeling*.

Yinuo Xu, Veronica Derricks, Allison Earl, and David Jurgens. 2025. Modeling annotator disagreement with demographic-aware experts and synthetic perspectives. *arXiv preprint arXiv:2508.02853*.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Rui Ye, Shuo Tang, Rui Ge, Yaxin Du, Zhenfei Yin, Siheng Chen, and Jing Shao. 2025. Mas-gpt: Training llms to build llm-based multi-agent systems. *arXiv preprint arXiv:2503.03686*.

Ori Yoran, Samuel Joseph Amouyal, Chaitanya Malaviya, Ben Bogin, Ofir Press, and Jonathan Berant. 2024. Assistantbench: Can web agents solve realistic and time-consuming tasks? *arXiv preprint arXiv:2407.15711*.

Guibin Zhang, Junhao Wang, Junjie Chen, Wangchunshu Zhou, Kun Wang, and Shuicheng Yan. 2025a. Agentracer: Who is inducing failure in the llm agentic systems? *arXiv preprint arXiv:2509.03312*.

Heng Zhang, Yuling Shi, Xiaodong Gu, Haochen You, Zijian Zhang, Lubin Gan, Yilei Yuan, and Jin Huang. 2025b. Graphtracer: Graph-guided failure tracing in llm agents for robust multi-turn deep search. *arXiv preprint arXiv:2510.10581*.

Jiayi Zhang, Jinyu Xiang, Zhaoyang Yu, Fengwei Teng, Xionghui Chen, Jiaqi Chen, Mingchen Zhuge, Xin Cheng, Sirui Hong, Jinlin Wang, and 1 others. 2024. Aflow: Automating agentic workflow generation. *arXiv preprint arXiv:2410.10762*.

Shaokun Zhang, Ming Yin, Jieyu Zhang, Jiale Liu, Zhiguang Han, Jingyang Zhang, Beibin Li, Chi Wang, Huazheng Wang, Yiran Chen, and 1 others. Which agent causes task failures and when? on automated failure attribution of llm multi-agent systems. In *Forty-second International Conference on Machine Learning*.

A Use of Generative AI Tools

Generative AI tools are used solely for minor drafting and editing purposes during the manuscript preparation.

B Annotation Effort

Dataset Composition. MP-BENCH consists of 169 hand-crafted MAS executions (average 33 interaction steps) and 120 automatically generated executions (average 8 steps), totaling 289 instances with triple expert annotation.

Annotator Qualifications. All annotators possess substantial technical expertise in AI and multi-agent systems: Annotator 1 (7 years in AI engineering and NLP), Annotator 2 (10+ years in applied AI and LLM-based agentic systems), and Annotator 3 (5 years in AI/ML development, LLMs, and AI agents). This ensures annotations reflect genuine expert judgment on MAS failure patterns.

Annotation Effort. Each hand-crafted execution requires approximately 31 minutes per annotator on average, while automatic executions requires approximately 14 minutes, reflecting the complexity difference in execution length and interaction patterns. Across all 289 instances and three annotators, the total annotation effort amounts to approximately 346 expert-hours (115 hours per annotator). Annotators were compensated at competitive rates commensurate with their expertise levels.

This substantial time investment prioritizes annotation quality and multi-perspective coverage over dataset scale, ensuring the reasoning quality essential for reliable evaluation of failure attribution systems. While this approach limits immediate scalability, it establishes a gold-standard benchmark for validating future automated annotation methods.

C Implementation Details

C.1 Failure Attribution Systems Details

We implement failure attribution systems using a diverse set of LLMs, including both proprietary and open-source models: GPT-4.1 (OpenAI, 2025a), GPT-5.1 (OpenAI, 2025b), GPT-oss-120B (Agarwal et al., 2025), Claude-Sonnet-4.5 (Anthropic, 2025), and Qwen3-8B (Yang et al., 2025). For failure attribution, we adopt the All-at-Once method (Zhang et al.) in a zero-shot setting, which provides a favorable balance between efficiency and accuracy. Unless otherwise specified, we set the number of LLM runs to $N = 3$, the sampling temperature to $\tau = 1.0$, and the maximum output length to 4,196 tokens. Proprietary models are accessed via their official APIs (OpenAI; Anthropic). For the open-source GPT-oss-120B, we use the Together.ai API (TogetherAI). Qwen3-8B is run locally on our own machine equipped with an NVIDIA RTX A6000 GPU.

C.2 Evaluation Details

For failure attribution evaluation, we report nDCG@5 and nDCG@full with exponential gain. This design accounts for the fact that exponential gain places greater emphasis on steps with high consensus rates. By additionally reporting full-ranking metrics, we provide a more comprehensive evaluation that also reflects performance on lower-consensus, perspective-dependent failure steps. For attribution reasoning evaluation, we use GPT-5.1 as the consolidation and judge model with $\tau = 0.0$ to ensure stable and deterministic judgments. Based on manual inspection, we find that the judge model produces reliable evaluation results.

D Additional Experimental Results

D.1 Sensitivity Analyses on LLM choice for Consolidation and LLM-as-a-Judge

In our main experiments, we use GPT-5.1 (OpenAI, 2025b) for both annotation consolidation (Section 4.1.5) and LLM-as-a-Judge evaluation (Section 5.2). To verify that our evaluation is robust to these choices, we conduct comprehensive sensitivity analyses using GPT-4.1 (OpenAI, 2025a) and Claude-Sonnet-4.5 (Anthropic, 2025) as alternative models. With three consolidation models and three judge models, we evaluate all nine possible combinations.

Ranking Consistency: Hand-Crafted MAS. For hand-crafted MAS executions, we compute Kendall's rank correlation coefficient across all $\binom{9}{2} = 36$ configuration pairs. The results demonstrate strong consistency:

- Average $\tau = 0.844$
- Minimum $\tau = 0.600$, Maximum $\tau = 1.00$
- 31 of 36 pairs pairs achieve $p < 0.1$

These correlations substantially exceed conventional thresholds for strong agreement, confirming that model rankings are robust to evaluation configuration.

Ranking Consistency: Automatic MAS. For automatic MAS executions, we observe similarly strong consistency:

- Average $\tau = 0.856$
- Minimum $\tau = 0.733$, Maximum $\tau = 1.00$
- 30 of 36 pairs achieve $p < 0.1$

Conclusion. Relative rankings remain highly stable across all configurations. This consistency demonstrates that our conclusions about model performance are not artifacts of specific consolidation or judge choices, but reflect genuine differences in failure attribution quality. The strong agreement (average $\tau \simeq 0.85$ for both settings) provides confidence that practitioners can reliably identify top-performing systems regardless of evaluation setup.

D.2 Attribution Reasoning Evaluation via BLEU and Embedding Similarity

The primary goal of this additional analysis is to demonstrate that the relative performance trends among different MAS failure attribution systems remain consistent, regardless of whether the metric is LLM-dependent or not. To this end, we have incorporated BLEU (lexical overlap) (Papineni et al., 2002) and Cosine Similarity (semantic alignment via text-embedding-3-small) as complementary metrics. Table 5 compares our LLM-Judge approach (utilizing GPT-5.1) against BLEU and Embedding Similarity across all evaluated systems.

Table 5: Comparison of reasoning evaluation metrics across failure attribution systems. BLEU and embedding cosine similarity (Embed. Sim.) are reported as complementary non-LLM-dependent metrics.

	Hand-Crafted MAS			Automatic MAS		
Model	LLM-Judge	BLEU	Embed. Sim.	LLM-Judge	BLEU	Embed. Sim.
GPT-4.1	7.89	5.93	0.70	7.87	7.03	0.73
GPT-5.1	7.91	4.92	0.68	8.28	6.88	0.75
Claude-Sonnet-4.5	8.16	6.76	0.72	8.24	8.19	0.75
Qwen3-8B	6.36	4.00	0.63	6.60	5.24	0.67
oss-120B	7.79	5.48	0.66	7.98	7.16	0.71

To quantify the stability of these trends, we calculated the Pearson correlation coefficients (r) between our LLM-Judge scores and the deterministic metrics.

Pearson correlation coefficients between LLM-Judge and the two deterministic metrics are reported in Table 5. On hand-crafted MAS, LLM-Judge correlates with BLEU at $r = 0.842$ and with embedding similarity at $r = 0.863$. On automatic MAS, the corresponding values are $r = 0.890$ and $r = 0.938$, respectively. These consistently high correlations confirm that model rankings under LLM-Judge are stable across evaluation metrics.

These high correlation coefficients ($r > 0.83$) provide empirical evidence that the LLM Judge is highly consistent with both lexical overlap (BLEU) and semantic alignment (Embedding). More importantly, this consistency demonstrates that the performance trends across various failure attribution systems remain stable regardless of the evaluation metric used.

D.3 Scaling N in Multi-LLM Failure Attribution

We investigate whether the scaling behavior with respect to N observed in Section 6.2.1 also holds in the multi-LLM failure attribution setting. For multi-LLM configurations, we follow the same setup as in Section 6.2.2, where three LLMs collaboratively perform failure attribution. We vary N from 1 to 10, corresponding to a total of 3 to 30 LLM runs (3 $\times$ N). Performance is evaluated using the average nDCG@5 across both hand-crafted and automatically generated MAS datasets. As shown in Figure 5, increasing the number of samples (N) leads to steady performance improvements, consistent with prior observations.

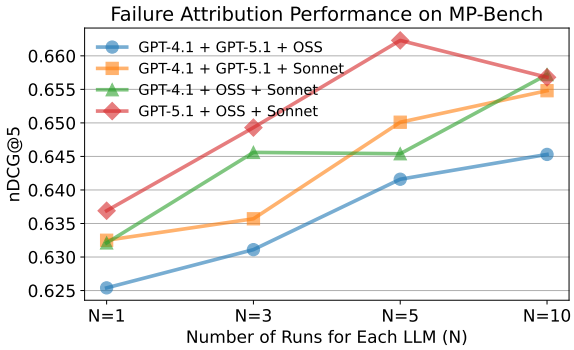

Figure 5: Failure attribution performance across varying numbers of LLM runs (N) on MP-BENCH. OSS denotes GPT-OSS-120B, and Sonnet denotes the Claude-Sonnet-4.5 model.

Brief Description

Annotators will receive json files containing MAS execution logs. Each log includes:

- A task description (e.g., ″Find popular hiking trails with waterfalls in Yosemite″)
- Step-by-step agent actions (e.g., WebSurfer, Verifier, Orchestrator outputs)
- A final system answer (which may or may not be correct)

In the json file, we have multiple fields: ″question″, ″ground_truth″, ″is_corrected″, and ″history.″

- ″question″
  - Describes the user′s task.
- ″ground_truth″
  - Represents the correct answer to the user′s task.
  - If not provided, the annotator must come up with the ground truth and fill it in.
  - If already provided, the annotator does not need to generate it.
- ″is_corrected″
  - indicates whether this MAS execution produced the correct answer.
  - true → The execution produced the correct answer.
  - false → The execution produced an incorrect conclusion.
- ″history″

  Contains a sequence of agent actions, with the following subfields:
  - ″content″: the action taken by each agent.
  - ″step″: The order of agent actions,
  - ″failure_annotation″, ″fail_category″, ″fail_reason″, and ″ideal_action″ are the annotation targets. The freelancer should complete these for each instance.

Detailed Annotation Guides

1. Failure Annotation – Identify whether each agent action was correct (0) or incorrect (1). This field has binary values, 1 or 0. If the same type of failure occurs multiple times, please annotate all such occurrences without omission.

2. Failure Reason – For incorrect actions, briefly describe the reason for failure in clear, natural language.

3. Ideal Action – Suggest the ideal action the agent should have taken. Do not include the actual simulation results of the agent. Instead, provide guidance for the failed action (e.g., ″The WebSurfer agent should have scrolled down the page to explore the information instead of visiting other pages″). The corrected action must be directly aligned with the failure reason you annotate.

4. Ground Truth – The log can optionally contain the ground truth answer for the user query. If it is not provided, please come up with the ground truth.

5. Is Corrected – Judge the correctness of the final response from the MAS with respect to the ground truth answer: “true” if the final answer is correct, “false” if incorrect. If the provided value is wrong, please correct it based on your judgment.

Figure 6: Annotation description provided to annotators.

You are an AI assistant tasked with analyzing a multi-agent conversation history when solving a real-world problem. The problem is: `{problem}`.
Identify all agents who made errors, at which steps each error occurred, and analyze the nature of these errors.
Here's the conversation:

`{chat history content}`

Based on this conversation, please provide a detailed analysis of the mistakes in the following JSON format. The output must be a list of objects, ordered by the step number. Each object must include:
1. agent_name: The name of the agents who made any mistakes during the conversation. Directly output the name of the Expert.
2. step_number: The integer step number where the mistake occurred (e.g., 0, 1, 2...). For example, in a conversation structured as follows:
{
"agent a": "xx",
"agent b": "xxxx",
"agent c": "xxxxx",
"agent a": "xxxxxxx"
},

the textual chat history is structured as follows:

Step 0: agent a: x
Step 1: agent b: xxxx
Step 2: agent c: xxxxx
Step 3: agent a: xxxxxxx

each entry represents a 'step' where an agent provides input. The 'x' symbolizes the speech of each agent. Please determine the step numbers where the mistakes occurred.
Please specify the step number as presented."
3. failure_reason: Briefly describe the reason for failure in clear, natural language.
4. ideal_action: Suggest the ideal action the agent should have taken. Do not include actual simulation results; instead, provide guidance aligned with the failure reason (e.g., 'The WebSurfer agent should have scrolled down the page to explore the information instead of visiting other pages').
Please answer ONLY in the following JSON format:
[
{
"agent_name": "AgentName1",
"step_number": "only integer",
"failure_reason": "Description of why it failed",
"ideal_action": "Guidance on what should have been done",
},
...
]
Please order the results by the step number. Do not include any other text or comments.

Figure 7: Prompt for failure attribution method (All At Once (Zhang et al.)).

You are an expert evaluator assessing the quality of a large language model's failure attribution reasoning by comparing it with a human expert annotator's annotation.
Your goal is NOT to check for exact wording matches, but to evaluate whether the model provides a plausible, well-grounded reasoning that aligns with the human annotator's judgment.
—
Task: `{question}`
Failure Step: `{step_num}`
Execution Log (context around step `{step_num}`):
`{chat history content}`
—
Model Prediction:
- Failure Reason: `{pred_fail_reason}`
- Ideal Action: `{pred_ideal_action}`

Human Expert Annotation:
- Failure Reason: `{GT_fail_reason}`
- Ideal Action: `{GT_ideal_action}`

—
Evaluation Criteria:
Evaluate the model's prediction by jointly considering the failure reason and ideal action, based on the following aspects:
1. Reasoning Alignment
Does the model capture the core reasoning behind why the step is considered a failure and what should have been done instead, as reflected in the human annotation?

2. Faithfulness to Execution Context
Is the model's reasoning grounded in the actual execution context of this step?
- Does it rely only on information available in the execution trace?
- Does it avoid introducing unsupported assumptions or hindsight knowledge?

3. Coverage and Completeness
Does the model address the key issues and considerations raised by the annotator, even if expressed differently?

4. Plausibility of Ideal Action
Is the proposed ideal action reasonable, actionable, and consistent with the annotator's intent?

—
Scoring Guidelines (1–10 scale):
- 9–10 (Excellent):
Strong alignment with the annotator's reasoning; well-grounded, faithful to the execution context, and captures the key rationale and ideal action clearly.

- 7–8 (Good):
Largely aligned with minor omissions or differences; reasoning is plausible and context-faithful.

- 5–6 (Moderate):
Partially aligned but missing important aspects or containing some questionable assumptions.

- 3–4 (Poor):
Weak alignment; reasoning diverges significantly from the annotator or lacks grounding in the execution context.

- 1–2 (Very Poor):
Largely incorrect, unfaithful to the execution context, or fails to provide a meaningful explanation.

—
Only return the JSON object. Do not include any additional text. Output Format (JSON only):
{ "overall_score": "integer between 1 and 10",
"fail_reason_score": "integer between 1 and 10",
"ideal_action_score": "integer between 1 and 10",
"reasoning": "Concise explanation of the judgment" }

Figure 8: LLM-as-a-Judge prompt for attribution reasoning evaluation.