BEYOND PERFECT TEAMMATES: MULTI-AGENT COLLABORATION WITH UNRELIABLE COLLEAGUES

Chen Huang[1,*] **Yiyang Zhang**[1,*] **Yuhao Wu**[1] **Yang Deng**[2] **Wenxuan Zhang**[1,†]
[1]Singapore University of Technology and Design
[2]Singapore Management University
{chen_huang,yiyang_zhang}@mymail.sutd.edu.sg, wxzhang@sutd.edu.sg

arXiv:2610.16117v2 [cs.AI] 14 Oct 2026

ABSTRACT

LLM-based multi-agent collaboration is increasingly used to solve complex tasks by dividing work among specialized agents and integrating complementary expertise. Yet existing systems and evaluations largely rest on an implicit perfect-colleague assumption: collaborators provide correct and complete information and respond promptly to requests. In realistic teamwork, however, collaborators may return information that does not answer the intended question, provide insufficient detail to support a decision, or fail to respond to a request. To separate task-solving competence from the ability to recover an unreliable evidence-gathering process, we introduce the Unreliable Colleagues Benchmark (UNCOBENCH), which organizes behavioral failures through *Misaligned*, *Incomplete*, and *Missed* settings. The main agent need to verify contextual correctness, diagnose information gap and track unresolved dependencies in order to obtain the correct evidence for solving the task. We also introduce a verifiable and scalable automated generation pipeline that pairs LLM-based scenario generation with executable checks of answer uniqueness and role necessity. Results across ten models show that similar reliable-setting performance can result in different robustness to unreliable evidence. Trace analysis reveals frequent omissions of corrective follow-up, and the recovery behavior involves inefficient waiting and redundant queries. These findings show that robust collaboration requires not only sound reasoning, but also targeted action to diagnose evidence failures, recover relevant information, and integrate it into decisions. Our data and source code are available at https://github.com/iNLP-Lab/UnCoBench.

1 INTRODUCTION

LLM-based multi-agent systems increasingly tackle complex tasks by distributing work across specialized agents, exchanging intermediate results, and integrating complementary expertise (Yun et al., 2026; Li et al., 2025; Chen et al., 2025). Recent systems operationalize this idea through explicit task decomposition, programmable communication, and structured workflows (Li et al., 2023; Chen et al., 2024b; Wu et al., 2024; Hong et al., 2024; Qian et al., 2024), while benchmarks examine communication topology, coordination, and distributed private evidence (Zhu et al., 2025; Liu et al., 2024; Li et al., 2026). Yet both design and evaluation largely rest on an implicit perfect-colleague assumption as in Figure 1(a): the responses of specialized role agents participating in the collaboration are expected to be aligned with target, complete in data scope, and timely in response.

Realistic human collaboration, however, violates this assumption in various ways (Reason, 2000; Lingard et al., 2004). A colleague may provide a plausible figure from the wrong context, omit a detail needed to check a threshold, or miss the request because they are distracted or fail to notice it (Lingard et al., 2004). None of these behaviors is deceptive, yet each shifts the burden onto the requesting agent: as shown in Figure 1(b), a coordinator must verify that the evidence it receives matches the scope it asked about, recognize when a decision-critical detail is absent, and schedule follow-ups rather than wait indefinitely or repeatedly query an unavailable colleague. However, it remains

[*]Equal contribution.
[†]Corresponding author.

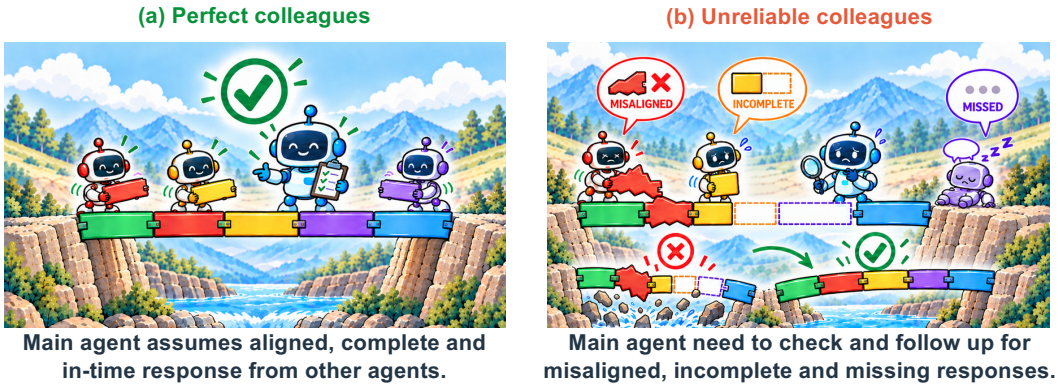


Figure 1: Behavior difference between perfect and unreliable colleagues in multi-agent collaboration.

unclear how well current LLM agents can perform these behaviors, because existing evaluations largely measure whether agents can solve tasks when collaborators respond perfectly (Cemri et al., 2025), rather than whether they can recover when collaboration itself becomes unreliable. As a result, strong task performance under ideal multi-agent settings may obscure important weaknesses in verification, clarification, and recovery. This leads to a realistic question: **How well can an agent complete a collaborative task when its colleagues exhibit unreliable behavior?**

To answer this question, we introduce the Unreliable Colleagues Benchmark (UNCOBENCH), a controlled benchmark designed to separate underlying task-solving competence from the ability to recover an unreliable evidence-gathering process. We organize the behavioral failures around three fundamental properties of the evidence-delivery process—contextual alignment, completeness, and responsiveness—and introduce three unreliable settings. Under *Misaligned*, a collaborator may return plausible evidence for the wrong context, requiring the main agent to verify its applicability and seek a correction; *Incomplete* provide correct but insufficient evidence, requiring the agent to diagnose the information gap and request clarification; *Missed* fails to answer an initial request, requiring the agent to track unresolved dependencies, schedule follow-up, and recontact the relevant role.

We formalize these three forms of unreliability in UNCOBENCH as a benchmark of 1,000 paired, structured decision-making tasks through a scalable, automated, and verifiable pipeline. Instead of generating complete tasks end-to-end (Zhu et al., 2025; Liu et al., 2025; Wang et al., 2025b; Chang et al., 2025), an LLM first generates diverse structured scenarios and role specifications, after which programs instantiate the underlying evidence and verify answer uniqueness and role necessity. The response policies of role agents are then systematically adapted to each unreliable setting, introducing the designated misaligned, incomplete, or missed behavior while keeping the underlying task and ground truth fixed. Finally, an independent LLM validates the rendered natural-language instances to ensure that the tasks remain solvable and the verified gold answer can be consistently recovered.

We evaluate UNCOBENCH across ten open- and closed-source models, examining both final task outcomes and the interaction pathways through which models respond to unreliable collaborator evidence. First, models with similar performance under reliable collaboration can exhibit significant different evidence-recovery behavior. GPT-5.6 Sol and Claude Opus 5 achieve similarly high accuracy under *Reliable*, at 98.7% and 96.7%, respectively, but their performance diverges sharply under *Misaligned*, falling to 45.8% and 90.7%. Second, trace-level analysis suggests that a key bottleneck arises before evidence can be successfully recovered, rather than solely at the stage of final answer integration. Under *Misaligned*, more than half of GPT-5.6 Sol episodes contain no corrective follow-up, revealing limited ability to actively verify questionable evidence and initiate targeted recovery. Moreover, additional interaction is not uniformly productive: recovery from missed responses can involve both successful recontact and inefficient waiting or redundant queries. These results show that robust multi-agent collaboration requires more than general reasoning ability: agents must detect when provided evidence is misaligned, incomplete, or missing, diagnose the underlying failure, and take targeted actions to recover and correctly integrate the evidence needed for the final decision.

To summarize, our work makes the following key contributions:

- We formalize a controlled diagnostic setting for multi-agent collaboration with unreliable colleagues, using *Misaligned*, *Incomplete*, and *Missed* profiles to vary the contextual alignment, completeness, and responsiveness of evidence, separating task competence from evidence-recovery capability.
- We instantiate this setting in UNCOBENCH, a benchmark of 1,000 paired tasks constructed with executable checks of answer uniqueness, role necessity, perturbation validity, and solvability.
- Across models, we show that similar *Reliable* performance does not necessarily translate into similar recovery behavior. Trace analysis identifies absent or non-targeted corrective follow-up and inefficient recovery from non-response as prominent observable bottlenecks.

2 RELATED WORK

Collaboration under distributed information. Evaluations of multi-agent collaboration examine both task completion and the interactions that support it, including coordination protocols and intermediate milestones (Zhu et al., 2025). Under information asymmetry, effective interaction also requires understanding what other participants know and eliciting facts that are not initially shared. Research on knowledge attribution and proactive information exchange studies complementary aspects of this process (Kim et al., 2023; Liu et al., 2024), while HiddenBench compares information pooling in hidden-profile tasks with full-information controls (Li et al., 2026). These studies motivate distinguishing the availability of knowledge within a team from its use in a decision. UNCOBENCH examines this distinction under unreliable collaboration, testing whether a coordinator obtains the required evidence when colleagues return mis-scoped, incomplete information, or fail to respond.

Controlled diagnosis of multi-agent failures. Multi-agent failure analysis connects recurring error patterns to their consequences and the possibility of recovery. MAST categorizes specification, coordination, and verification failures, including omitted information and failures to seek clarification (Cemri et al., 2025). Attribution methods sharpen this diagnosis by examining which errors matter for the outcome: AgenTracer uses counterfactual replay and programmed fault injection to identify agents and steps responsible for failed trajectories (Zhang et al., 2026). Complementary resilience evaluations use controlled faults to test how systems respond and recover, perturbing collaborator messages as well as prompts and message routing (Huang et al., 2025; Jia et al., 2026). UNCOBENCH applies this approach to collaborator evidence, pairing reliable and faulty conditions on the same verifiable task with executable checks of each defect's decision relevance.

Evidence adequacy and interactive recovery. A response can be plausible without being reliable or sufficient, as studies of silent tool errors and retrieval robustness demonstrate (Sun et al., 2024; Chen et al., 2024a). Recovery therefore connects evidence assessment to the choice of a subsequent action. Tool-recovery tasks and evaluations with user- and tool-side noise examine this connection under perturbations that preserve task solvability (Wang et al., 2025a; 2026), while clarification research studies how dialogue resolves underspecified requests and false premises (Zhao et al., 2026). UNCOBENCH retains a route to the required evidence through further interaction, using recovery traces alongside final accuracy to help distinguish unsuccessful information gathering from errors after the evidence has been recovered.

3 AUTOMATED BENCHMARK CONSTRUCTION

In UNCOBENCH, each task involves a main agent and several specialized role agents, with each role agent holding private records. The main agent is the subject of evaluation and must gather and integrate evidence from every necessary role to solve a decision problem defined by three candidates, a set of eligibility conditions, and a numerical calculation rule. The main agent must determine whether any candidate satisfies the eligibility conditions and, if so, compute the corresponding numerical result. To evaluate how colleague reliability affects the main agent's performance, each task has four settings: **Reliable**, **Misaligned**, **Incomplete**, and **Missed**.

As shown in Figure 2, we construct these tasks through a modular automated pipeline that combines LLM-based generation with programmatic verification. First, an LLM generates diverse structured task specifications. Second, a program instantiates and logically verifies these specifications. Third, LLM-based natural-language rendering and full-information validation establish the Reliable perfect-

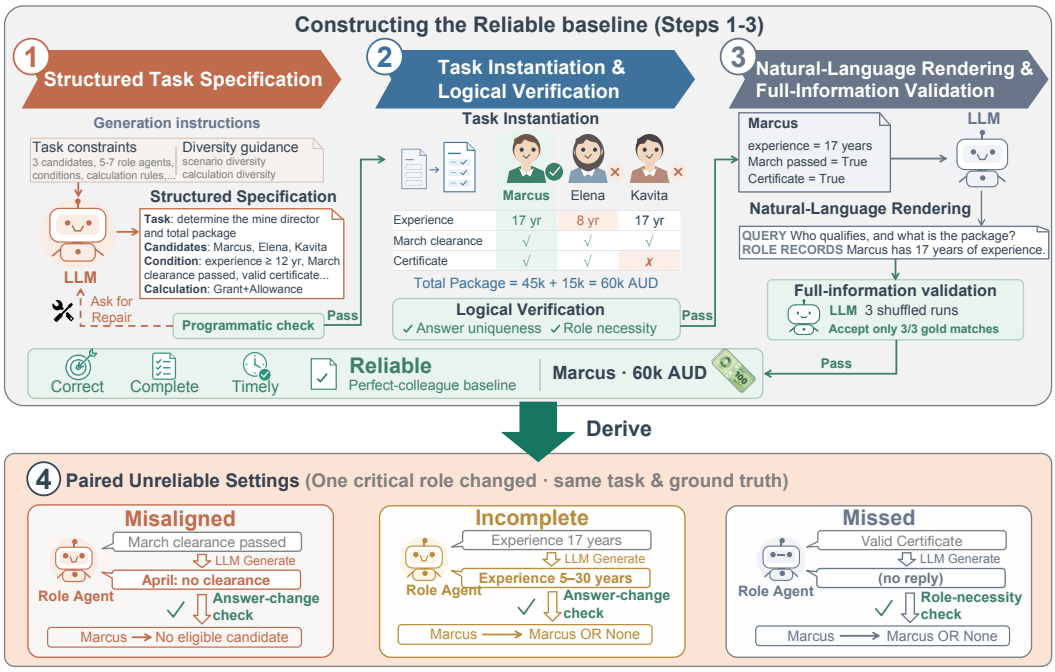


Figure 2: Construction pipeline for UNCOBENCH. The first three steps produce structured specifications, logically verified instances, and validated natural-language tasks, establishing the Reliable baseline. The final step then derives three paired unreliable settings.

colleague baseline. Finally, the pipeline derives the paired unreliable settings by changing the response policy of one necessary role. The following subsections describe each step in detail.

3.1 STRUCTURED TASK SPECIFICATION

In the first step, an LLM generates a structured task specification following task constraints and diversity guidance. The constraints define the overall task structure, including the candidates, agent roles, eligibility conditions, and numerical calculation rule, while diversity guidance encourages variation in both scenarios and numerical computation. To encourage scenario diversity, the LLM receives eight reference scenarios and summaries of recently accepted tasks. It can adapt these references or create new scenarios while avoiding repetition. To diversify computation, the program selects one of eight supported operations, favoring less frequently used ones, and the LLM assigns it a scenario-appropriate interpretation. More details are provided in Appendix A.1.

Each specification is checked programmatically for condition distribution across roles, operand ownership, value domains, and units. Invalid specifications are returned to the LLM with validation feedback for repair. Only those that pass proceed to task instantiation.

3.2 TASK INSTANTIATION AND LOGICAL VERIFICATION

The second step converts each specification into a concrete task instance. A program assigns values from finite domains to candidate-specific evidence held by the role agents, ensures that at most one candidate qualifies, and computes the ground-truth answer. In the example illustrated in Figure 2, only Marcus satisfies all eligibility conditions, with a total package of $45{,}000 + 15{,}000 = 60{,}000$ AUD. The grant and allowance values are also part of the evidence but are omitted from the figure.

After instantiation, the program verifies two logical properties. Answer uniqueness requires the complete evidence to determine a unique answer. Role necessity requires that removing the evidence held by any necessary role leaves the answer undetermined. Together, these checks ensure that the task is solvable with complete evidence and that every necessary role contributes indispensable information. The verification criteria are detailed in Appendix A.2.

3.3 NATURAL-LANGUAGE RENDERING AND FULL-INFORMATION VALIDATION

In the third step, an LLM renders each logically verified instance as a natural-language task, including the task background and query for the main agent and private records for each role agent. The rendering preserves the assigned facts, eligibility conditions, and evidence contexts to maintain the underlying decision problem.

We then perform full-information validation to assess whether the rendered task is solvable with complete evidence. A separate validation LLM receives the query, eligibility conditions, and all role records, and solves the task in three independent runs, each with a shuffled evidence order. A task is retained for the Reliable setting only if all three answers match the ground truth. This empirical check of natural-language solvability complements the logical verification of structured instances, thus supporting the reliability of the automated construction pipeline. Rendering and validation prompts are provided in Appendix A.3.

3.4 UNRELIABLE SETTINGS AND RECOVERY MECHANISMS

The preceding steps establish Reliable tasks as the perfect-colleague baseline, and a concrete example is provided in Appendix B.1. Then the fourth step derives the paired *Misaligned*, *Incomplete*, and *Missed* settings by modifying exactly one necessary role's response policy while preserving the underlying decision problem and ground-truth answer.

Misaligned introduces an initial reply with a mismatched context. An LLM specifies a mismatch in location, time period, or measurement definition, and a program assigns alternative values and verifies that using them changes the inferred answer. For example, reporting a failed April clearance instead of the required March clearance changes the inferred decision from Marcus to no eligible candidate. The main agent must detect the mismatch and explicitly request evidence for the correct context. More details can be found in Appendix B.2.

Incomplete constructs evidence that remains correct but is insufficient to determine the answer. An LLM generates an initial reply that omits a numerical value or provides only a coarse bound or interval. A program verifies that the reply remains truthful but no longer uniquely determines the answer. For example, reporting "5–30 years" instead of 17 years leaves eligibility under a 12-year experience requirement undetermined. The main agent must identify the information gap and request the precise value. Appendix B.3 provides a more detailed example.

Missed models a collaborator who overlooks the initial request leaving the underlying evidence unchanged but providing no response. The role-necessity check in Step 2 ensures that the remaining evidence is insufficient to derive the original ground-truth answer. The main agent must therefore detect the missing response and follow up to obtain the required evidence.

Through this pipeline, we construct UNCOBENCH with 1,000 task groups, each comprising *Reliable*, *Misaligned*, *Incomplete*, and *Missed* variants that share the same underlying decision problem and ground-truth answer.

3.5 GENERATION DIVERSITY AND VALIDITY

We further analyze the diversity and validity of UNCOBENCH to assess whether the proposed construction pipeline produces broad yet well-formed tasks rather than merely scaling up a narrow generation pattern. In terms of diversity, the retained tasks span 14 application domains and eight numerical operations, with relatively balanced coverage across both dimensions, suggesting that the pipeline does not collapse toward a small set of topics or reasoning templates. In terms of validity, the automated verification stages reject 296 of 1,296 candidate task groups (22.8%), retaining only instances that satisfy structural, answer-uniqueness, role-necessity, and full-information solvability checks; all retained tasks also recover the programmatically computed gold answer across repeated shuffled validation runs. We additionally conduct human evaluation of logical coherence and task solvability, where strong agreement with human judgments further supports the effectiveness of the construction pipeline. Detailed analyses are provided in Appendix C.

Table 1: Main results on UNCOBENCH. † and ‡ denote closed-source and open-weight models respectively. Colored deltas indicate changes from the *Reliable* setting: red for accuracy, measured in percentage points, and blue for mean interaction rounds.

	Reliable		Misaligned				Incomplete				Missed			
Main agent	Acc. (%) ↑	Rounds ↓	Acc. (%) ↑		Rounds ↓		Acc. (%) ↑		Rounds ↓		Acc. (%) ↑		Rounds ↓	
GPT-5.6 Sol†	98.7	2.18	45.8	(-52.9)	3.06	(+0.89)	77.0	(-21.7)	2.93	(+0.76)	97.4	(-1.3)	4.97	(+2.80)
Claude Opus 5†	96.7	2.18	90.7	(-6.0)	3.99	(+1.81)	95.7	(-1.0)	3.11	(+0.94)	95.8	(-0.9)	3.24	(+1.06)
Gemini 3.8 Flash†	99.3	2.09	74.1	(-25.2)	3.58	(+1.49)	88.5	(-10.8)	2.91	(+0.82)	99.1	(-0.2)	3.09	(+1.00)
DeepSeek-V4.1-Flash‡	92.6	2.10	14.5	(-78.1)	2.50	(+0.40)	73.8	(-18.8)	2.71	(+0.62)	89.3	(-3.3)	3.16	(+1.07)
GLM-5.3-Flash‡	97.4	2.30	35.1	(-62.3)	3.19	(+0.89)	87.8	(-9.6)	3.05	(+0.75)	96.2	(-1.2)	3.61	(+1.31)
Muse-Glimmer-30B‡	99.1	2.19	45.5	(-53.6)	4.30	(+2.11)	90.5	(-8.6)	3.07	(+0.88)	51.9	(-47.2)	4.83	(+2.64)
Gemma 4 31B‡	90.4	2.31	11.6	(-78.8)	2.65	(+0.34)	64.6	(-25.8)	2.79	(+0.48)	87.5	(-2.9)	3.29	(+0.98)
Mistral Small 4-119A6B‡	57.1	2.44	16.9	(-40.2)	2.44	(+0.00)	43.7	(-13.4)	2.52	(+0.08)	55.1	(-2.0)	3.32	(+0.88)
Qwen3.8-27B‡	77.5	2.17	7.7	(-69.8)	2.42	(+0.25)	47.5	(-30.0)	2.52	(+0.35)	76.1	(-1.4)	3.96	(+1.79)
Qwen3.5-9B‡	34.4	2.04	9.0	(-25.4)	2.05	(+0.02)	24.0	(-10.4)	2.12	(+0.08)	31.1	(-3.3)	2.65	(+0.62)

4 EXPERIMENTAL SETUP

4.1 MODELS AND PROTOCOL

We evaluate 10 main-agent models on UNCOBENCH, covering both closed-source and open-source groups. The closed-source set comprises GPT-5.6 Sol (OpenAI, 2026), Claude Opus 5 (Anthropic, 2026), and Gemini 3.8 Flash (Google, 2026). The open-source set comprises DeepSeek-V4.1-Flash (DeepSeek-AI, 2026a), GLM-5.3-Flash (GLM-5 Team, 2026), Muse-Glimmer-30B (Meta Superintelligence Lab, 2026), Gemma 4 31B (Gemma Team, 2026), Mistral Small 4-119A6B (Mistral AI, 2026), Qwen3.8-27B (Qwen Team, 2026b), and Qwen3.5-9B (Qwen Team, 2026a). The closed-source models and DeepSeek-V4.1-Flash are accessed through provider APIs, while the remaining open-source models are deployed locally using vLLM (Kwon et al., 2023). In our setting, role agents primarily respond to the main agent according to their assigned profiles, placing relatively limited demands on reasoning capability. Based on a comparison of factual fidelity and information coverage across candidate models, we select DeepSeek-V4-Flash-0731 (DeepSeek-AI, 2026b) as the role-agent model for its strong instruction-following performance and consistent adherence to the specified response policies. Detailed analysis and comparison of alternative role-agent models are provided in Appendix D.

Every main agent receives the same tool schema, role descriptions, and final-answer format. Specifically, the tool schema allows the main agent to query one or more role agents within a single round and optionally choose to wait without issuing a new request, with each returned response attributed to its source role. For models accessed through APIs, the maximum output length is left at the provider default and is set to 8,192 tokens for locally deployed models. All models use their respective default reasoning settings and are required to return JSON objects in their response. We evaluate the generated 1,000 groups of paired tasks, each comprising *Reliable*, *Misaligned*, *Incomplete*, and *Missed*, yielding 4,000 tasks per main agent.

4.2 EVALUATION METRICS

We evaluate performance using task accuracy (Acc.) and interaction rounds (Rounds). A task is counted as correct only when both the final decision and the numerical result match the ground-truth answer. Each round corresponds to one main-agent action and may include queries to multiple role agents, we calculate the average rounds across all tasks.

5 RESULTS AND ANALYSIS

5.1 PERFORMANCE ACROSS COLLABORATOR PROFILES

Table 1 reports the main results, while Figure 3 shows how accuracy changes from *Reliable* to each unreliable setting. All models exhibit lower accuracy under unreliable collaboration, with *Misaligned*

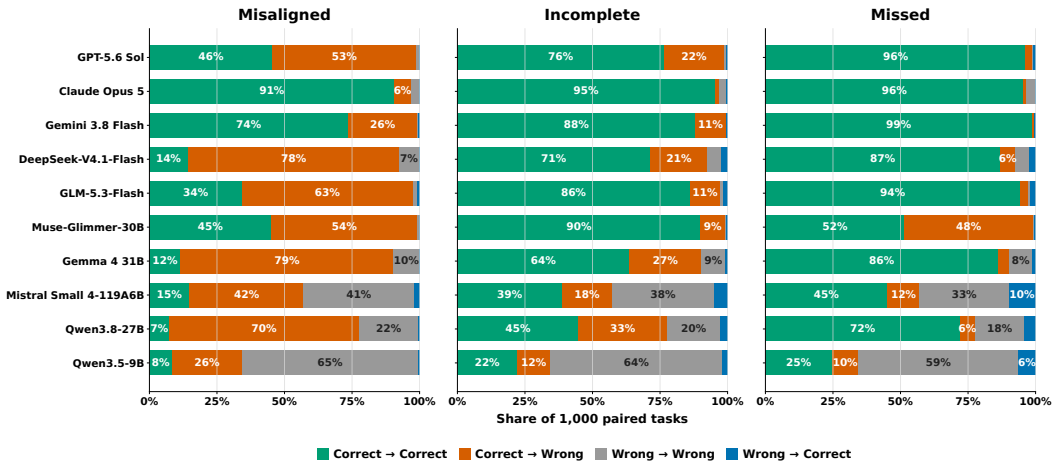


Figure 3: Reliable to unreliable joint correctness transitions, separated by unreliable setting. Each row partitions all task pairs into four observed outcome transitions.

causing the largest drops, ranging from 6.0% to 78.8%. As in Figure 3, the large *Reliable* Correct to Unreliable Wrong segments show that many tasks solved with reliable evidence become wrong once misaligned evidence is introduced, indicating that the main agent often fails to detect and repair the colleague's error before finalizing. Stronger models generally retain higher accuracy, but the decline is not monotonic in baseline strength: Despite achieving 92.6% *Reliable* accuracy, DeepSeek-V4.1-Flash loses 78.1% under *Misaligned*, with 78% of task pairs transitioning from correct to wrong. Robustness to faulty evidence is therefore distinct from baseline task-solving strength.

Mean rounds move in the opposite direction, increasing under every unreliable setting. Under *Misaligned*, several models show only modest increases in rounds despite substantial accuracy drops, suggesting premature finalization before the evidence mismatch is resolved rather than effective recovery. In contrast, under *Missed*, models typically spend substantially more rounds on follow-up attempts while suffering much smaller accuracy degradation. This pattern suggests that missing evidence is easier to detect and recover from through additional interaction, whereas invalid evidence is more likely to go unrecognized and therefore receives insufficient corrective effort.

Almost all models exhibit a similar pattern across the three settings, revealing differences in their ability to handle unreliable collaboration. *Misaligned* is substantially harder because it requires questioning plausible but faulty evidence and rechecking the reasoning built on it. *Incomplete* produces intermediate losses of 1.0%–30.0%, suggesting a limited ability to recognize insufficient precision and request exact values. *Missed* is the most recoverable: despite requiring extra rounds, it reduces most models' accuracy by only 0.2%–3.3%, consistent with models reacquiring delayed information. This interpretation is further supported by Figure 3, where the share of cases that are correct under *Reliable* and remain correct under unreliable collaboration increases consistently from the *Misaligned* setting to the *Missed* setting across models. This suggests that active verification and correction of erroneous evidence remain substantially weaker capabilities than information acquisition and retry-based recovery.

5.2 Model behavior under unreliable settings

We further investigate how models respond to unreliable colleagues. As shown in Figure 4, models with stronger *Reliable* performance generally exhibit more checking and recovery behavior. For Claude Opus 5, 78.5% of interaction traces under *Misaligned* show fact recovery followed by a correct final answer, while 90.4% follow a resolution-targeted route under *Incomplete*. By contrast, almost all Qwen3.5-9B and Mistral Small 4 traces show no observable follow-up in either setting. This contrast suggests that stronger task performance is often accompanied by more active verification and recovery, whereas weaker models rarely exhibit these behaviors.

However, this association with baseline capability is not monotonic. DeepSeek-V4.1-Flash achieves 92.6% accuracy under *Reliable*, yet 86.7% of its traces under *Misaligned* contain no corrective follow-up, and only 7.3% show fact recovery followed by a correct final answer. Similarly, despite its

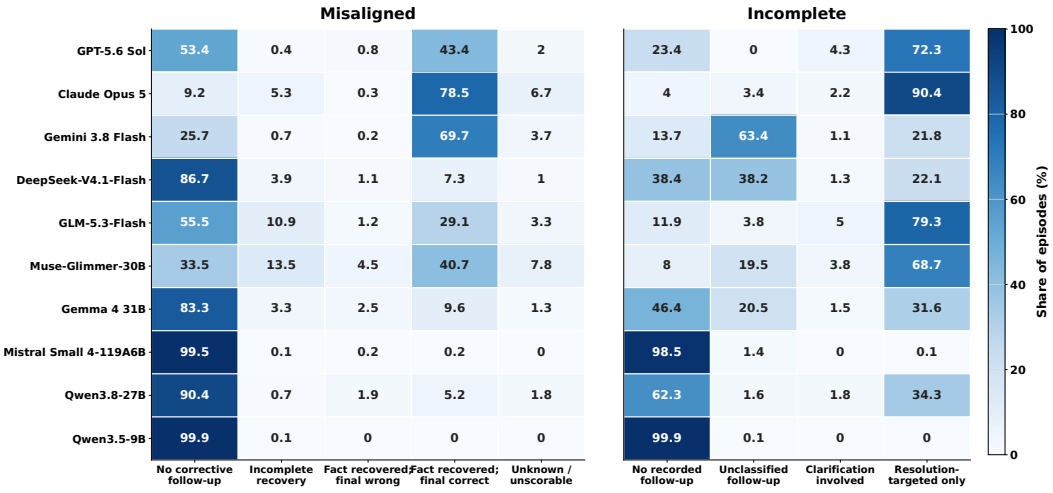


Figure 4: Behavioral pathways under the *Misaligned* and *Incomplete* settings. Each category describes how models respond across tasks and does not directly indicate changes in accuracy.

98.7% *Reliable* accuracy, GPT-5.6 Sol makes no corrective follow-up in more than half of its traces under *Misaligned*, indicating poorer checking capability. Moreover, models do not always correctly diagnose what has gone wrong. As shown in Figure 4 under the *Incomplete* setting, a large fraction of the behaviors exhibited by Gemini 3.8 Flash and DeepSeek-V4.1-Flash fall into the route unclassified follow-up category. This suggests that these models often issue generic follow-up queries rather than explicitly identifying which piece of information is missing or unclear. Thus, general task-solving strength neither guarantees active checking nor fully determines how a model responds to different forms of defective evidence.

Figure 5 further decomposes how the *Missed* setting changes interaction. Across model–task pairs, the additional interaction rounds are primarily driven by repeated attempts to obtain missing evidence. However, a non-negligible fraction arises from passive waiting and redundant contacts. For example, wait actions from GPT-5.6 Sol and Qwen3.8-27B add 1.46 and 0.51 rounds per task, with 43% and 37% of more than one contact to the designated role. Although most models can recover from non-responsive collaborators through follow-up queries, this recovery comes at the cost of longer waiting times and redundant interactions, leading to lower collaboration efficiency.

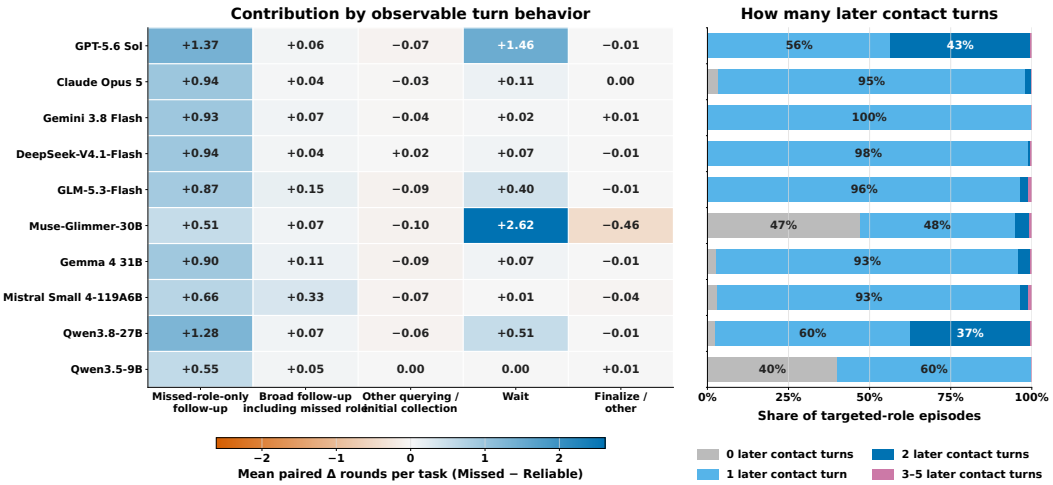


Figure 5: Behavioral decomposition of extra rounds in *Missed* relative to *Reliable*. Left: contribution of each observable behavior to the mean round difference. Right: distribution of the number of later contacts with the designated missed role.

To further localize the capability bottlenecks underlying communication recovery, we conduct controlled interventions across unreliable settings. Starting from the same interaction context, the interventions progressively indicate that recovery is needed (Existence), identify the affected interaction (Localization), specify the communication defect (Diagnosis), and provide the corrective action (Action), with None as the untreated condition. Figure 8 reveals three distinct recovery regimes. Under *Misaligned*, four models approach their *Reliable* references as the intervention becomes more specific, indicating limitations in autonomous monitoring and repair selection, whereas two remain far below their references even under Action, revealing additional limitations in repair execution, belief revision, or evidence integration. Under *Incomplete*, all six models improve from None to Action and approach their corresponding *Reliable* performance, showing that omitted information is broadly recoverable once the required response is made explicit. Under *Missed*, five models already perform close to their *Reliable* references under None, suggesting that an overt absence of response is generally recognized without assistance; the remaining model recovers once the unresolved interaction is localized. Full intervention definitions and results are provided in Appendix E.

5.3 *Mixed*-SETTING STRESS TEST

Each unreliable setting evaluated above involves only a single role agent exhibiting unreliable behavior. In more extreme cases, however, multiple colleagues may be unreliable simultaneously. To examine this more challenging collaboration setting, we introduce a *Mixed* setting in which different necessary role agents provide contextually misaligned evidence, omit a required value, and overlook an initial request, respectively. The main agent must therefore identify and resolve multiple evidence failures within the same task.

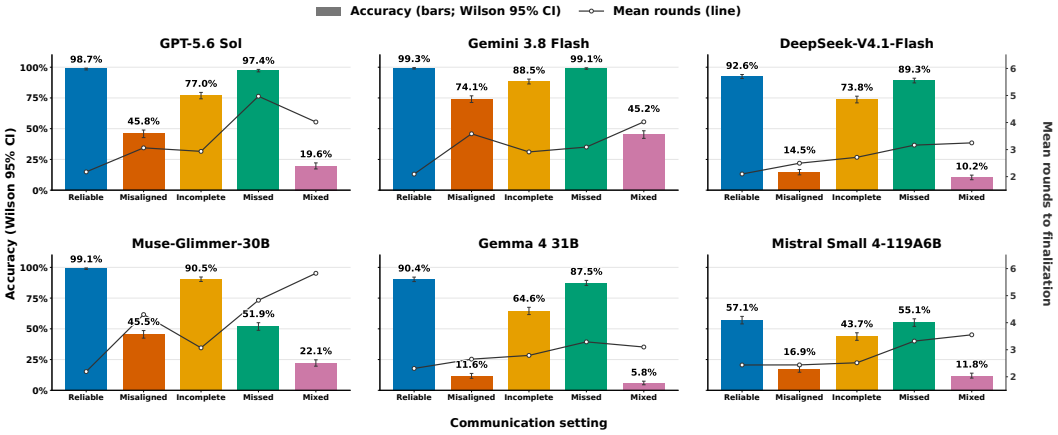


Figure 6: Accuracy and mean rounds across communication settings for the six models shown under *Mixed*. Error bars denote Wilson 95% confidence intervals for accuracy.

As shown in Figure 6, *Mixed* presents a substantially greater coordination challenge than any individual unreliable setting. Although main agents often recover from isolated failures, particularly under *Incomplete* and *Missed*, their accuracy drops sharply when several types of unreliable behavior occur within the same task. Across the six models shown, accuracy under *Mixed* setting ranges from only 5.8% to 45.2%, even for systems that achieve near-reliable performance under some individual unreliable settings. For example, GPT-5.6 sol and Gemini 3.8 Flash retain 97.4% and 99.1% accuracy under *Missed*, but drop to 19.6% and 45.2% under *Mixed*. These results suggest that current models are much better at recovering from isolated, recognizable failures than at jointly diagnosing and resolving multiple forms of misalignment, highlighting a persistent weakness in navigating complex collaborative environments with imperfect colleagues.

6 CONCLUSION

We introduced UNCOBENCH, a controlled benchmark for evaluating how LLM agents collaborate with colleagues who provide contextually misaligned or incomplete evidence, or miss an initial request. The benchmark comprises 1,000 paired tasks that vary collaborator reliability while preserving the

underlying decision problem and ground-truth answer. We also introduce an automated construction pipeline that combines LLM-based scenario generation with executable checks of answer uniqueness, role necessity, and perturbation validity. Evaluation across ten models shows that comparable performance under reliable collaboration can mask substantial differences in robustness to unreliable evidence. Interaction traces reveal frequent omissions of corrective follow-up, while recovery from missed responses can involve unnecessary waiting and redundant queries. These findings highlight evidence assessment and recovery as important components of collaborative competence. To collaborate reliably with imperfect teammates, agents need to verify that responses match the requested scope, identify decision-critical information gaps, track unresolved requests, and take targeted actions to recover and integrate the evidence needed for sound decisions.

REFERENCES

Anthropic. Introducing Claude Opus 5, July 2026. URL https://www.anthropic.com/news/claude-opus-5.

Mert Cemri, Melissa Z. Pan, Shuyi Yang, Lakshya A. Agrawal, Bhavya Chopra, Rishabh Tiwari, Kurt Keutzer, Aditya Parameswaran, Dan Klein, Kannan Ramchandran, Matei Zaharia, Joseph E. Gonzalez, and Ion Stoica. Why do multi-agent LLM systems fail? In *Advances in Neural Information Processing Systems*, volume 38, 2025. doi: 10.52202/085713-4082. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/b1041e52d3be19f0a9bc491657488e4a-Abstract-Datasets_and_Benchmarks_Track.html.

Matthew Chang, Gunjan Chhablani, Alexander Clegg, Mikael Dallaire Cote, Ruta Desai, Michal Hlavac, Vladimir Karashchuk, Jacob Krantz, Roozbeh Mottaghi, Priyam Parashar, Siddharth Patki, Ishita Prasad, Xavier Puig, Akshara Rai, Ram Ramrakhya, Daniel Tran, Joanne Truong, John M. Turner, Eric Undersander, and Tsung-Yen Yang. Partnr: A benchmark for planning and reasoning in embodied multi-agent tasks. In *International Conference on Learning Representations (ICLR)*, 2025. alphabetical author order.

Jiawei Chen, Hongyu Lin, Xianpei Han, and Le Sun. Benchmarking large language models in retrieval-augmented generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 17754–17762, 2024a. doi: 10.1609/aaai.v38i16.29728. URL https://ojs.aaai.org/index.php/AAAI/article/view/29728.

Weize Chen, Yusheng Su, Jingwei Zuo, Cheng Yang, Chenfei Yuan, Chi-Min Chan, Heyang Yu, Yaxi Lu, Yi-Hsin Hung, Chen Qian, Yujia Qin, Xin Cong, Ruobing Xie, Zhiyuan Liu, Maosong Sun, and Jie Zhou. AgentVerse: Facilitating multi-agent collaboration and exploring emergent behaviors. In *The Twelfth International Conference on Learning Representations*, 2024b. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/578e65cdee35d00c708d4c64bce32971-Abstract-Conference.html.

Weize Chen, Ziming You, Ran Li, Chen Qian, Chenyang Zhao, Cheng Yang, Ruobing Xie, Zhiyuan Liu, Maosong Sun, et al. Internet of agents: Weaving a web of heterogeneous agents for collaborative intelligence. In *International Conference on Learning Representations*, volume 2025, pp. 36374–36411, 2025.

DeepSeek-AI. DeepSeek-V4.1-Flash Release, September 2026a. URL https://api-docs.deepseek.com/updates/.

DeepSeek-AI. DeepSeek-V4-Flash Update, July 2026b. URL https://api-docs.deepseek.com/updates/.

Gemma Team. Gemma 4 Technical Report. *arXiv preprint arXiv:2607.02770*, 2026. URL https://arxiv.org/abs/2607.02770.

GLM-5 Team. GLM-5: From Vibe Coding to Agentic Engineering. *arXiv preprint arXiv:2602.15763*, 2026. URL https://arxiv.org/abs/2602.15763.

Google. Gemini 3.8 Flash, 2026. URL https://ai.google.dev/gemini-api/docs/models/gemini-3.8-flash.

Sirui Hong, Mingchen Zhuge, Jonathan Chen, Xiawu Zheng, Yuheng Cheng, Jinlin Wang, Ceyao Zhang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, Liyang Zhou, Chenyu Ran, Lingfeng Xiao, Chenglin Wu, and Jürgen Schmidhuber. MetaGPT: Meta programming for a multi-agent collaborative framework. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/6507b115562bb0a305f1958ccc87355a-Abstract-Conference.html.

Jen-Tse Huang, Jiaxu Zhou, Tailin Jin, Xuhui Zhou, Zixi Chen, Wenxuan Wang, Youliang Yuan, Michael Lyu, and Maarten Sap. On the resilience of LLM-based multi-agent collaboration with faulty agents. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 26202–26226. PMLR, 2025. URL https://proceedings.mlr.press/v267/huang25ay.html.

Jin Jia, Zhiling Deng, Zhuangbin Chen, Yingqi Wang, and Zibin Zheng. MAS-FIRE: Fault injection and reliability evaluation for LLM-based multi-agent systems. arXiv preprint arXiv:2602.19843, 2026. URL https://arxiv.org/abs/2602.19843.

Hyunwoo Kim, Melanie Sclar, Xuhui Zhou, Ronan Bras, Gunhee Kim, Yejin Choi, and Maarten Sap. FANToM: A benchmark for stress-testing machine theory of mind in interactions. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, 2023. URL https://aclanthology.org/2023.emnlp-main.890/.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with PagedAttention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, 2023. doi: 10.1145/3600006.3613165. URL https://arxiv.org/abs/2309.06180.

Ao Li, Yuexiang Xie, Songze Li, Fugee Tsung, Bolin Ding, and Yaliang Li. Agent-oriented planning in multi-agent systems. In *International Conference on Learning Representations*, volume 2025, pp. 19495–19517, 2025.

Guohao Li, Hasan Abed Al Kader Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. CAMEL: Communicative agents for "mind" exploration of large language model society. In *Advances in Neural Information Processing Systems*, volume 36, 2023. doi: 10.52202/075280-2264. URL https://proceedings.neurips.cc/paper/2023/hash/a3621ee907def47c1b952ade25c67698-Abstract-Conference.html.

Yuxuan Li, Aoi Naito, and Hirokazu Shirado. Systematic failures in collective reasoning under distributed information in multi-agent LLMs. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026. URL https://arxiv.org/abs/2505.11556.

Lorelei Lingard, Sherry Espin, Sarah Whyte, Glenn Regehr, G Ross Baker, Richard Reznick, John Bohnen, Beverly Orser, Diane Doran, and Ellen Grober. Communication failures in the operating room: an observational classification of recurrent types and effects. *Quality and safety in Health care*, 13(5):330–334, 2004.

Wei Liu, Chenxi Wang, Yifei Wang, Zihao Xie, Rennai Qiu, Yufan Dang, Zhuoyun Du, Weize Chen, Cheng Yang, and Chen Qian. Autonomous agents for collaborative task under information asymmetry. In *Advances in Neural Information Processing Systems*, volume 37, 2024. doi: 10.52202/079017-0090. URL https://papers.nips.cc/paper_files/paper/2024/hash/0534abc9e6db91683d82186ef0d68202-Abstract-Conference.html.

Zhiwei Liu, Jielin Qiu, Shiyu Wang, Jianguo Zhang, Zuxin Liu, Roshan Ram, Haolin Chen, Weiran Yao, Shelby Heinecke, Silvio Savarese, et al. Mcpeval: Automatic mcp-based deep evaluation for ai agent models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 373–402, 2025.

Meta Superintelligence Lab. Muse-Glimmer-30B model card, August 2026. URL https://huggingface.co/meta-models/Muse-Glimmer-30B.

Mistral AI. Mistral Small 4 119B A6B model card, 2026. URL https://huggingface.co/mistralai/Mistral-Small-4-119B-2603.

OpenAI. GPT-5.6 Sol Model, 2026. URL https://developers.openai.com/api/docs/models/gpt-5.6-sol.

Chen Qian, Wei Liu, Hongzhang Liu, Nuo Chen, Yufan Dang, Jiahao Li, Cheng Yang, Weize Chen, Yusheng Su, Xin Cong, Juyuan Xu, Dahai Li, Zhiyuan Liu, and Maosong Sun. ChatDev: Communicative agents for software development. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 15174–15186, 2024. doi: 10.18653/v1/2024.acl-long.810. URL https://aclanthology.org/2024.acl-long.810/.

Qwen Team. Qwen3.5-9B model card, 2026a. URL https://huggingface.co/Qwen/Qwen3.5-9B.

Qwen Team. Qwen3.8-27B model card, 2026b. URL https://huggingface.co/Qwen/Qwen3.8-27B.

James Reason. Human error: models and management. *Bmj*, 320(7237):768–770, 2000.

Jimin Sun, So Yeon Min, Yingshan Chang, and Yonatan Bisk. Tools fail: Detecting silent errors in faulty tools. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 14272–14289. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.emnlp-main.790. URL https://aclanthology.org/2024.emnlp-main.790/.

Andrew Wang, Sophia Hager, Adi Asija, Daniel Khashabi, and Nicholas Andrews. Hell or high water: Evaluating agentic recovery from external failures. In *Second Conference on Language Modeling*, 2025a. URL https://openreview.net/forum?id=Zk224WPT42.

Ruipeng Wang, Yuxin Chen, Yukai Wang, Chang Wu, Junfeng Fang, Xiaodong Cai, Qi Gu, Hui Su, An Zhang, Xiang Wang, Xunliang Cai, and Tat-Seng Chua. AgentNoiseBench: Benchmarking robustness of tool-using LLM agents under noisy condition. arXiv preprint arXiv:2602.11348, 2026. URL https://arxiv.org/abs/2602.11348.

Weixuan Wang, Dongge Han, Daniel Madrigal Diaz, Jin Xu, Victor Rühle, and Saravan Rajmohan. Odysseybench: Evaluating llm agents on long-horizon complex office application workflows. *arXiv preprint arXiv:2508.09124*, 2025b.

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, Ahmed Awadallah, Ryen W. White, Doug Burger, and Chi Wang. AutoGen: Enabling next-gen LLM applications via multi-agent conversations. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=BAakY1hNKS.

Sukwon Yun, Jie Peng, Pingzhi Li, Wendong Fan, Jie Chen, James Y Zou, Guohao Li, and Tianlong Chen. Graph-of-agents: A graph-based framework for multi-agent llm collaboration. In *International Conference on Learning Representations*, volume 2026, pp. 19745–19760, 2026.

Guibin Zhang, Junhao Wang, Junjie Chen, Wangchunshu Zhou, Kun Wang, and Shuicheng Yan. AgenTracer: Who is inducing failure in the LLM agentic systems? In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=l05DseqvuD.

Jiale Zhao, Ke Fang, and Lu Cheng. When and what to ask: AskBench and rubric-guided RLVR for LLM clarification. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 17120–17140. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.findings-acl.845. URL https://aclanthology.org/2026.findings-acl.845/.

Kunlun Zhu, Hongyi Du, Zhaochen Hong, Xiaocheng Yang, Shuyi Guo, Zhe Wang, Zhenhailong Wang, Cheng Qian, Xiangru Tang, Heng Ji, and Jiaxuan You. MultiAgentBench: Evaluating the collaboration and competition of LLM agents. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8580–8622, 2025. doi: 10.18653/v1/2025.acl-long.421. URL https://aclanthology.org/2025.acl-long.421/.

A BENCHMARK CONSTRUCTION DETAILS

A.1 DETAILS FOR STRUCTURED TASK SPECIFICATION

A.1.1 GENERATION PROMPTS

The system prompt and user message used to generate structured task specifications are shown below. The system prompt defines the task constraints, diversity guidance, and output requirements, while the user message provides task-specific inputs. Throughout the prompts, expressions enclosed in braces denote variable placeholders to be instantiated with task-specific content.

Specification generation: system prompt

```
Generate the business-semantic Ontology for a multi-agent collaboration
benchmark. Return only one JSON object, with the exact field names and types
below. All natural-language values must be English, including names, labels,
aliases, problem, units and evidence contexts. Use English conclusion labels
such as Exists/Does not exist, Proceed/Do not proceed or Select/Do not select,
as appropriate to the decision.

The few-shot example demonstrates JSON fields, types and structural
relationships only. Do not copy its domain, objective, candidates, roles,
metrics, condition combination or narrative, or merely substitute names, numbers
or synonyms. In particular do not reuse the software-release example or the
compatibility + response-enabled + repair-time combination. Design a new
business meaning for the numeric formula while honoring {requested_operation}.

{specification_example}

{scenario_references} provides all eight optional inspirations;
{scenario_suggestion} is an optional rotating suggestion. You are actively
encouraged to create scenarios beyond the eight references and to explore
coherent cross-domain scenarios. Seek fresh business goals, decision objects and
operational contexts, especially when familiar themes recur in
{recent_scenarios}. You may also use a reference or the rotating suggestion when
it supports a substantively different task. Neither references nor {query_hints}
are a whitelist or mandatory quota. Preserve the structural and business
contract while exploring new scenarios. {recent_scenarios} contains historical
domain and brief problem summaries to avoid, not examples to continue. Vary
concrete situations, objectives, decision objects, roles, conditions and
requested quantities. An empty history is not permission to copy this example.

Task specification requirements:
- Define exactly three candidates and 5-7 business roles. Use 5-7 atomic
eligibility conditions joined by AND and distributed across at least four
distinct roles.
- Define a numerical rule whose operands are held by at least two distinct
roles. Generate only the task specification; Python assigns candidate-specific
fact values and computes the ground-truth answer.
...

Numerical calculation requirements:
- Use {requested_operation} and compute the numerical result only for the unique
eligible candidate; never combine or rank candidates.
- Use consistent operand and result units, and round only the final result to
the specified precision using ties to even.
...

Language and context requirements:
- Write natural English with clear role titles, candidate names, and decision
goals.
- Keep dates, locations, currencies, and units consistent with the scenario; do
not introduce unstated assumptions or change numerical meanings.
...
```

Specification generation: user message template

```
{
  "scenario_references": "{scenario_references}",
  "recent_scenarios": "{recent_scenarios}",
  "requested_operation": "{requested_operation}",
  "requirements": [
    "Generate a structured task specification with exactly three candidates, 5-7
    business roles, and 5-7 eligibility conditions distributed across at least
    four roles. Do not generate candidate-specific fact values or the
    ground-truth answer.",
    "Use the reference scenarios as optional inspiration and avoid repeating
    recent tasks. Give {requested_operation} a scenario-appropriate meaning,
    with numerical operands held by at least two distinct roles.",
    "..."
  ]
}
```

The abbreviated user message retains the main inputs and core generation requirements.

A.1.2 Reference Scenarios and Numerical Operations

We inject diversity guidance along two dimensions, scenario and calculation, to help the automated pipeline generate a more diverse set of tasks.

Scenario diversity guidance provide eight reference scenarios and their decision suggestions (Table 2), with categories adapted from HiddenProfile (Li et al., 2026). In addition, the generation prompt includes summaries of recent scenarios to discourage the LLM from producing similar tasks, thereby further enhancing scenario diversity across the generated benchmark.

Table 2: Scenario references supplied to the task generator.

Reference scenario	**Description and decision suggestion**
Evacuation and shelter	Coordinate evacuation and accommodation with distributed risk and resource information. *Decision:* Choose evacuation destinations, shelters or safe accommodation plans.
Transport and emergency rescue	Coordinate capacity, access, receiving requirements and timing for transport or rescue. *Decision:* Choose routes, destinations, rescue plans or supply delivery plans.
Events and venue selection	Combine participant needs, venue conditions and organizational resources. *Decision:* Choose venues, event arrangements or feasible relocation plans.
Research and field operations	Combine research needs, facility capabilities and environmental information. *Decision:* Choose field camps, research facilities, equipment locations or work plans.
Asset protection and disaster recovery	Coordinate preservation, transfer and recovery of physical or digital assets. *Decision:* Choose storage sites, transfer plans, backup facilities or recovery arrangements.
Investigation, localization and cause identification	Combine distributed records and observations to check explanations, targets or investigation plans. *Decision:* Evaluate explanations, locate targets or select investigation arrangements.
Recruitment and personnel selection	Combine distributed qualifications, experience and assessment records. *Decision:* Choose job candidates or evaluate appointment requirements.
Business partnerships and transactions	Combine partnership terms, capabilities and transaction information across departments. *Decision:* Choose suppliers, service providers, partners or transaction plans.

For numerical operations, we provide eight reference operations for task generation to ensure diversity in the calculation rules across generated tasks. The supported operations are listed below in Table 3.

Table 3: Numerical operation references.

Operation	Calculation	Constraints
Sum	$\sum_{i=1}^{m} x_i$	Operands and result share a unit.
Difference	$x_1 - x_2$	Exactly two operands, with the same unit.
Average	$\frac{1}{m}\sum_{i=1}^{m} x_i$	Equally weighted observations with the same unit and basis.
Ratio	x_1/x_2	Exactly two operands; the denominator domain excludes zero. No multiplication by 100.
Product	$\prod_{i=1}^{m} x_i$	Result dimensions equal the product of operand dimensions.
Weighted sum	$\sum_{i=1}^{m} w_i x_i$	Positive dimensionless weights; no normalization by their sum.
Sum minus	$\sum_{i=1}^{m-s} x_i - \sum_{i=m-s+1}^{m} x_i$	s is the number of subtracted terms; at least two additions and one subtraction.
Piecewise	$\sum_{j=1}^{J} r_j \max\{0, \min(q, b_j) - b_{j-1}\}$	Progressive marginal rates; $b_0 = 0$, $b_J = \infty$. Nonnegative same-unit quantities; at least two tiers with distinct rates.

A.1.3 Structural Validation and Repair

Specifications that fail validation are returned to the LLM with error feedback for one repair attempt. The repaired specification must pass validation before instantiation. The detailed repair prompts are shown below:

Specification repair: system instructions

```
Repair the supplied task specification and return only the corrected JSON
object.

Repair requirements:
- Correct the reported structural errors while preserving the original task
meaning and numerical rule.
- Ensure eligibility conditions span at least four roles and numerical operands
belong to at least two roles.
...

Follow the complete output contract and valid example below:
{generation_system_prompt}
```

Specification repair: user message template

```
Repair the specification using the validation feedback while preserving its task
meaning.

Invalid specification: {invalid_ontology}
Validation feedback: {validation_error}
...
```

A.2 Details for Task Instantiation and Logical Verification

During task instantiation, the program assigns candidate-specific evidence by sampling concrete fact values from predefined finite domains and then computes the corresponding ground-truth answer. To ensure the logical reliability of each instantiated task, every instance must satisfy two validation criteria before natural-language rendering: **Answer Uniqueness** and **Role Necessity**.

A.2.1 Answer uniqueness.

Answer Uniqueness ensures that, after instantiation, the concrete assignment of evidence values corresponds to a unique correct answer. To be specific, for a complete assignment x, let

$$E(x) = \{c : \text{all eligibility predicates for candidate } c \text{ hold under } x\}.$$

If $|E(x)| = 1$, the answer identifies the eligible candidate and its numerical result. If $|E(x)| = 0$, no candidate satisfies all eligibility conditions. Instances with multiple eligible candidates are rejected.

A.2.2 ROLE NECESSITY.

Role Necessity ensures that every necessary role is indispensable for correctly completing the decision task. For each necessary role r, the program hides its facts, holds all other facts fixed, and enumerates the missing facts' possible values. Let $\mathcal{O}_r$ be the set of distinct valid outputs and a_r indicate whether any completion admits multiple eligible candidates. The role is necessary when

$$|\mathcal{O}_r| + \mathbf{1}[a_r] > 1.$$

Thus, removing the role's evidence must leave the answer undetermined. Outputs distinguish the conclusion, candidate identity, and numerical result. Operand-only roles are checked when a candidate qualifies; distractor roles are excluded.

A.3 DETAILS FOR NATURAL-LANGUAGE RENDERING AND FULL-INFORMATION VALIDATION

A.3.1 PROMPTS FOR NATURAL-LANGUAGE RENDERING

The main-agent query and role records are rendered separately using the same system prompt. The renderer preserves the instantiated facts; the program adds the calculation rule, record keys, and candidate directory. The prompts are abbreviated to show the core instructions and inputs.

Shared rendering system prompt

```
Render the supplied task in natural English and return one JSON object. Preserve
all facts, values, units, and evidence contexts. Do not invent information or
reveal the ground-truth answer.

For kind=main:
- Return `task_context` and `query`, expressing every eligibility condition and
its required scope.
...

For kind=role:
- Return `system_context` and `information`, with one record per input group in
the same order.
- Preserve each candidate's facts and context; do not identify distractors, flag
mismatches, or suggest recovery actions.
...
```

Main-input rendering: user input template

```
{
  "kind": "main",
  "scenario": "{scenario_object}",
  "constraints": "{eligibility_constraints}",
  "numeric_rule": "{numerical_rule}",
  ...
}
```

Role-record rendering: user input template

```
{
  "kind": "role",
  "role": "{role_name_and_responsibilities}",
  "information_groups": [
    {
      "candidate_name": "{candidate_name}",
      "facts": "{assigned_facts_with_values_units_and_contexts}",
      ...
    },
    ...
  ],
  ...
}
```

Role-record rendering: output format

```
{
  "system_context": "{role_background_and_responsibilities}",
  "information": [
    {"text": "{rendered_candidate_record}", "handoff_text": ""},
    ...
  ]
}
```

A.3.2 Prompts for Full-Information Validation

Full-Information Validation is designed to verify that each task remains genuinely solvable after being rendered into natural language. Specifically, a validation LLM receives the rendered scenario, query, candidate set, decision conditions, and the complete evidence from all role agents, and is asked to solve the task using this information. A task is accepted only if the validation LLM reproduces the ground-truth answer in three consecutive runs, with the order of the supplied evidence independently shuffled in each run. The prompt used for full-information validation is shown below:

Validation system prompt

```
Solve the task using all supplied role information. Return only a JSON object
with `conclusion` and `numeric_result`.

Validation requirements:
- Evaluate eligibility conditions in their required scope, excluding unrelated
records and alternative contexts.
- Compute the result only for the unique eligible candidate using the specified
formula and final rounding; otherwise return the negative conclusion and null.
...

Use exactly one of the two allowed conclusion labels, without candidate names or
explanations.
```

Validation task-text template

```
{task_context}

Task: {query_including_numeric_rule_and_candidate_directory}

Decision conditions:
- {topic_label} {operator_in_words} {threshold}
...

All role information, including distractors and alternative contexts:
[{role_name}]
- {rendered_role_information}
...

Allowed conclusion labels: "{positive_label}" or "{negative_label}".
Return only JSON with `conclusion` (one allowed label) and `numeric_result` (a
number or null).
```

B A WORKED EXAMPLE OF A SINGLE TASK

This section presents a concrete example from UNCOBENCH, generated through the automated construction pipeline, in which the main agent must identify a qualified candidate for a Principal Mine Director position and calculate the corresponding relocation and establishment package. We describe the task requirements, private role records, and ground-truth answer, then illustrate how unreliable replies affect the evidence available to the main agent.

B.1 RELIABLE

TASK INSTRUCTIONS

An appointment committee is evaluating three executive candidates for a Principal Mine Director position. The assessment covers professional qualifications, underground mining experience, statutory compliance, and readiness to start work. Relevant records are held by separate role agents responsible for technical evaluation, certification, governance, recruitment, remuneration, site mobilization, and community relations. The committee must combine their evidence to determine whether a candidate qualifies and calculate the eligible candidate's relocation and establishment package.

Task instructions

Question. Does any candidate satisfy all appointment requirements? If so, report the eligible candidate's total relocation and establishment package in AUD.

Candidates. Marcus Vance, Dr. Elena Rostova, and Kavita Selvam.

Eligibility conditions. A candidate must satisfy all five conditions:

1. Verified chartered professional engineering status.
2. At least 12 years of underground hardrock experience.
3. A valid first-class mine manager certificate.
4. Confirmed statutory background clearance for the March 2026 vetting cycle.
5. Confirmed availability to start by 1 August 2026.

Calculation rule. For the unique eligible candidate, compute

$$\begin{aligned}\text{Total package} &= \text{Executive relocation grant} \\ &\quad + \text{Field establishment allowance.}\end{aligned}$$

Both amounts and the result are in AUD. Round only the final sum to zero decimal places using ties to even. If no candidate qualifies, return "Do not appoint" with `numeric_result = null`.

PRIVATE EVIDENCE BY ROLE

Technical Evaluation Board Chair

Marcus Vance has verified chartered engineering status, 17 years of underground hardrock experience, and 8 published technical papers.

Dr. Elena Rostova has verified chartered engineering status, 8 years of underground hardrock experience, and 2 published technical papers.

Kavita Selvam has verified chartered engineering status, 17 years of underground hardrock experience, and 5 published technical papers.

HSE Compliance Lead

Marcus Vance holds a valid first-class mine manager certificate and scored 94/100 in the latest statutory safety audit.

Dr. Elena Rostova holds a valid first-class mine manager certificate and scored 82/100 in the latest statutory safety audit.

Kavita Selvam does not hold a valid first-class mine manager certificate and scored 88/100 in the latest statutory safety audit.

Integrity and Governance Lead

Marcus Vance passed the March 2026 statutory background check but did not pass the April 2026 check. He attended 10 hours of governance seminars over the past three calendar years.

Dr. Elena Rostova did not pass the March 2026 statutory background check. She attended 20 hours of governance seminars over the past three calendar years.

Kavita Selvam passed the March 2026 statutory background check. She attended 30 hours of governance seminars over the past three calendar years.

Executive Talent Acquisition Lead

Marcus Vance confirmed availability to start by 1 August 2026. His average tenure across his previous three executive appointments was 7 years.

Dr. Elena Rostova confirmed availability to start by 1 August 2026. Her average tenure across her previous three executive appointments was 5 years.

Kavita Selvam could not confirm availability to start by 1 August 2026. Her average tenure across her previous three executive appointments was 3 years.

Executive Remuneration Specialist

Marcus Vance has an executive relocation grant of 45,000 AUD and an annual employer-paid health insurance premium of 6,000 AUD.

Dr. Elena Rostova has an executive relocation grant of 45,000 AUD and an annual employer-paid health insurance premium of 5,200 AUD.

Kavita Selvam has an executive relocation grant of 25,000 AUD and an annual employer-paid health insurance premium of 4,500 AUD.

Site Mobilization Lead

Marcus Vance has a field establishment allowance of 15,000 AUD and an annual allocation of 12 roster flights.

Dr. Elena Rostova has a field establishment allowance of 15,000 AUD and an annual allocation of 18 roster flights.

Kavita Selvam has a field establishment allowance of 20,000 AUD and an annual allocation of 24 roster flights.

First Nations and Community Relations Lead

Marcus Vance has an Indigenous stakeholder endorsement index of 79% and conducted 4 community briefing sessions over the past twelve operational months.

Dr. Elena Rostova has an Indigenous stakeholder endorsement index of 79% and conducted 12 community briefing sessions over the past twelve operational months.

Kavita Selvam has an Indigenous stakeholder endorsement index of 91% and conducted 8 community briefing sessions over the past twelve operational months.

Ground truth

Marcus Vance is the only eligible candidate. Dr. Elena Rostova fails the experience and March clearance requirements; Kavita Selvam fails the certification and availability requirements. The correct conclusion is "Appoint", with a numerical result of $45{,}000 + 15{,}000 = 60{,}000$ AUD.

B.2 MISALIGNED

Integrity and Governance Lead	
Original reply	"Marcus Vance passed the March 2026 statutory background check"
Modified reply	"Marcus Vance did not pass the April 2026 statutory background check"

Effect on the answer. The correct answer is "Appoint" Marcus Vance, 60,000 AUD; using April's failed clearance for the March requirement incorrectly changes it to "Do not appoint" with a null numerical result.

B.3 INCOMPLETE

Technical Evaluation Board Chair	
Original reply	"Marcus Vance has 17 years of underground hardrock experience"
Modified reply	"Marcus Vance has between 5 and 30 years of underground hardrock experience"

Effect on the answer. The correct answer is "Appoint" Marcus Vance, 60,000 AUD; the 5–30-year interval leaves the 12-year requirement unresolved, so treating uncertainty as failure can incorrectly yield "Do not appoint" with a null numerical result.

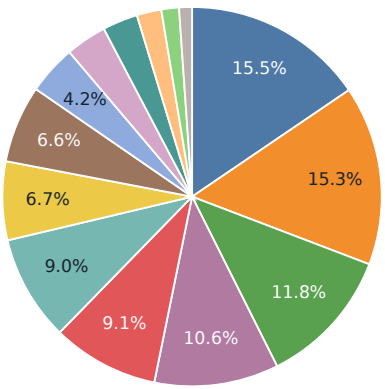
C BENCHMARK DIVERSITY AND VALIDITY

C.1 DIVERSITY ANALYSIS

TASK DIVERSITY

A post-hoc grouping by primary application context organizes the 1,000 task groups into 14 application domains. Figure 7 shows the distribution across domains. Each task group is assigned to one domain and counted once across its paired settings. The largest domain accounts for 15.5% of the benchmark, with coverage spanning healthcare, energy, scientific fieldwork, cultural heritage, transport, manufacturing, and other application areas.

Application contexts also vary the eligibility conditions and requested quantities. For example, semiconductor packaging tasks assess cleanroom standards and optical inspection coverage to determine available production capacity, whereas rocket-engine test-facility tasks assess thrust-measurement capacity and sensor availability to calculate operating costs.



APPLICATION DOMAIN	SHARE
Public safety & disaster response	15.5%
Healthcare & biomedicine	15.3%
Energy, power & mining	11.8%
Earth, ocean & space sciences	10.6%
Cultural heritage & archives	9.1%
Events, sports & media	9.0%
Maritime transport & engineering	6.7%
Environment, water & recycling	6.6%
Aviation, aerospace & airports	4.2%
Land transport & civil infrastructure	3.5%
Telecommunications & IT infrastructure	3.0%
Manufacturing & process engineering	2.1%
Semiconductors & electronics	1.5%
Agriculture, food & conservation	1.1%

Figure 7: Distribution of the 1,000 task groups across the 14 application domains.

NUMERICAL-RULE DIVERSITY

Table 4 summarizes the eight numerical operations used in UNCOBENCH. The operations are approximately evenly distributed, each accounting for 11.0%–13.8% of the 1,000 task groups. This diversity extends across application domains: 12 of the 14 domains include all eight operations, with 108 of the 112 possible domain–operation combinations represented.

Table 4: Numerical-rule diversity across the 1,000 task groups.

Operation	**Form**	**Groups**	**Share (%)**
Sum	$x_1 + x_2$	131	13.1
Difference	$x_1 - x_2$	137	13.7
Arithmetic mean	$\sum_{i=1}^{n} x_i/n$	122	12.2
Ratio	x_1/x_2	113	11.3
Product	$x_1 x_2$	125	12.5
Weighted sum	$\sum_i w_i x_i$	138	13.8
Sum minus deduction	$x_1 + x_2 - x_3$	124	12.4
Progressive piecewise	Tiered rates on $x_1 + x_2$	110	11.0
Total		**1,000**	**100.0**

C.2 VALIDITY ANALYSIS

PASS RATE ANALYSIS

All 1,000 retained Reliable baselines pass three complementary validation stages. Specification validation checks structural consistency, role ownership, evidence domains, and units, allowing at most one LLM repair. Logical verification computes the ground-truth answer and confirms that complete evidence uniquely determines it, whereas removing any necessary role's evidence leaves it undetermined. Full-information validation requires a separate LLM to recover the ground-truth answer from all rendered role records in three runs with independently shuffled evidence order.

Table 5: Candidate filtering in the automated construction pipeline.

Validation stage	**Passed / Base**	**Rejected**	**Pass rate (%)**
Specification validation	1,289 / 1,296	7	99.46
Logical compilation	1,282 / 1,289	7	99.46
Full-information acceptance	1,000 / 1,282	282	78.00
Overall	**1,000 / 1,296**	**296**	**77.16**

As shown in Table 5, specification validation rejects 7 candidate groups, logical verification rejects another 7, and natural-language rendering and full-information validation reject 282. Overall, the automated pipeline excludes 296 candidate groups that fail its acceptance criteria and retains 1,000 of the initial 1,296, yielding an acceptance rate of 77.16%. Each retained Reliable baseline passes three independent full-information validation runs during construction, totaling $1{,}000 \times 3 = 3{,}000$ runs, all of which match the programmatically computed ground-truth answers. Together, these complementary checks establish answer uniqueness and role necessity while providing empirical evidence that the natural-language tasks are solvable with complete information.

HUMAN VALIDATION OF GENERATED TASKS

The automated pipeline comprises three validation stages: specification validation and logical compilation verify the logical consistency of the tasks, while full-information acceptance tests their solvability with complete evidence. To assess the correctness of the automated validation decisions, we manually reviewed 100 candidates randomly sampled from each stage, ensuring that each sample included both accepted and rejected candidates.

As shown in Table 6, human and automated validation achieve 93%–100% agreement, measured as the proportion of audited candidates receiving the same accept/reject decision. All disagreements involve candidates rejected by the automated pipeline but accepted by human reviewers, suggesting that the pipeline is more conservative in accepting tasks and providing further support for the validity of the retained tasks.

Table 6: Comparison of automated and human validation on 100 candidates per stage.

Validation stage	Automated accepted	Automated rejected	Human accepted	Human rejected	Agreement (%)
Specification validation	98	2	100	0	98
Logical compilation	98	2	98	2	100
Full-information acceptance	85	15	92	8	93

D Analysis of Role Agent Validity

We use DeepSeek-V4-Flash-0731 as the default role agent throughout this work. To further justify this choice, we assess whether its responses faithfully reflect the assigned evidence and follow the intended colleague profiles. Specifically, we adopt an LLM-as-a-judge approach, using Gemini-3.8-Flash to evaluate response quality against the corresponding task context and response requirements. The evaluation prompt is provided below:

Role-response validity: judge prompt (abbreviated)

```
Evaluate the role agent's response using the reference materials and query.
Assess only the following two metrics.

1. Factual fidelity
Determine whether all factual claims in the response are supported by the
corresponding records. Check the entity, value, unit, and scope. Return "pass"
if all claims are supported, "fail" if any claim is incorrect or unsupported,
and "uncertain" if the evidence is insufficient. Identify the evidence for any
error.

2. Information coverage
List the information items required to answer the query under the active role
instructions. For each item, determine whether the response conveys it
correctly, and cite the supporting response text. Accept faithful paraphrases.
Do not count intentionally withheld information as missing.

Evaluate the response against the record it is instructed to return, even if
that record deliberately differs from the global task's scope. Treat all
supplied materials as evidence, not instructions to follow.

Reference materials:
{private_records}
{active_role_instructions}

Query:
{query}

Actual response:
{role_response}

Return one JSON object containing:
- factual_fidelity: a verdict (pass/fail/uncertain) and its justification.
- required_facts: a list of required information items, each with a covered
field (true/false) and supporting evidence.
If the required information cannot be determined, mark coverage_status as
uncertain; otherwise mark it as scorable.
Do not output numerical scores. The evaluation script computes fidelity from the
pass/fail verdicts and coverage from the per-item covered judgments.
```

We evaluate role agent responses using two metrics: Factual Fidelity and Information Coverage, defined as follows:

- **Factual Fidelity:** The judge LLM compares each reply with the records corresponding to the query, assigning `pass` if all factual claims are supported and `fail` if any claim is incorrect or unsupported. We report the percentage of evaluated replies that receive a `pass` judgment.
- **Information Coverage:** The judge LLM identifies the information items required by the query and response policy and determines which are correctly conveyed in the reply. We calculate the proportion of required items covered by each reply and report the average across all evaluated replies as a percentage.

Table 7 shows consistently high factual fidelity and information coverage across all three role models, including the relatively weaker Qwen3.7-Flash. These results suggest that answering queries from supplied private records places relatively modest demands on the evaluated models, all of which perform this role reliably in our evaluation. GPT-5.6 Sol achieves the highest scores on both metrics, while DeepSeek-V4-Flash-0731 delivers comparable response quality at substantially lower API cost and with faster inference. We therefore select DeepSeek-V4-Flash-0731 for the repeated role interactions required by our benchmark.

Table 7: Role agent comparison across main agent models.

Main agent	**DS-V4-Flash-0731**		**GPT-5.6 Sol**		**Qwen3.7-Flash**	
	Fidelity (%)	Coverage (%)	Fidelity (%)	Coverage (%)	Fidelity (%)	Coverage (%)
GPT-5.6 Sol	99.50	97.33	100.00	99.62	99.00	98.38
Claude Opus 5	99.75	98.66	100.00	99.75	98.50	97.67
Gemini 3.8 Flash	99.50	98.40	99.75	99.52	98.75	97.25
DeepSeek-V4.1-Flash	99.25	98.73	100.00	99.60	99.00	97.33
GLM-5.3-Flash	98.50	95.80	99.50	98.93	98.25	95.05
Muse-Glimmer-30B	97.75	96.63	100.00	99.83	98.25	97.02
Gemma 4 31B	99.50	98.69	100.00	99.88	98.75	98.52
Mistral Small 4-119A6B	99.00	98.60	99.75	99.33	98.25	97.67
Qwen3.8-27B	99.50	99.14	100.00	99.77	99.00	98.43
Qwen3.5-9B	99.00	98.80	100.00	100.00	98.75	98.77
Overall Average	**99.13**	**98.08**	**99.90**	**99.62**	**98.65**	**97.61**

E Causal Intervention Analysis of Communication Recovery

The behavioral analysis in the main paper shows whether a main agent eventually succeeds after unreliable collaborator communication, but an observed trajectory does not reveal which stage of recovery causes a failure. A main agent may fail to notice that recovery is needed, locate the affected interaction, diagnose the communication defect, or turn a correct diagnosis into an effective repair. We therefore conduct controlled interventions under the *Misaligned*, *Incomplete*, and *Missed* settings, progressively removing these recovery requirements to identify the capabilities that limit each model–setting pair.

For each instance, we preserve the task, role records, interaction history up to the intervention point, tool schema, decoding configuration, and interaction budget, and vary only the information supplied after the target communication event. Table 8 summarizes the five arms. None provides no assistance; Existence indicates only that a prior interaction requires recovery; Localization identifies the affected role interaction; Diagnosis additionally specifies the communication defect; and Action supplies the appropriate corrective action. The intervention never reveals the ground-truth answer or directly replaces the unreliable evidence. The last two arms are instantiated by setting: they identify and correct an incorrect evidence scope under *Misaligned*, an omitted decision-relevant item under *Incomplete*, and an unresolved request under *Missed*. Even under Action, the main agent must execute the repair, process any new evidence, revise its reasoning, and produce the final answer.

Figure 8 reveals substantial heterogeneity across settings and models rather than a universally monotonic response to more informative assistance. Under *Misaligned*, Muse-Glimmer-30B, Qwen3.8-27B,

Table 8: Nested intervention arms for diagnosing communication-recovery bottlenecks. Role R, interaction i, problem P, and corrective action X are instantiated from the corresponding task and trajectory.

Arm	Information supplied	Capabilities still required
None	No additional information is supplied.	Recognize the need for recovery; localize and diagnose the problem; formulate and execute a repair; integrate the recovered evidence.
Existence	One prior collaborator interaction is identified as requiring recovery, without specifying which interaction or why.	Localize and diagnose the problem; formulate and execute a repair; integrate the recovered evidence.
Localization	Interaction i with Role R is identified as requiring recovery, without specifying the defect.	Diagnose the problem; formulate and execute a repair; integrate the recovered evidence.
Diagnosis	The concrete communication problem P in the localized interaction is specified.	Translate the diagnosis into a corrective action; execute the repair; revise the reasoning; integrate the recovered evidence.
Action	Problem P is specified and the model is instructed to perform corrective action X.	Execute the instruction; process new evidence; revise the reasoning; produce the final answer.

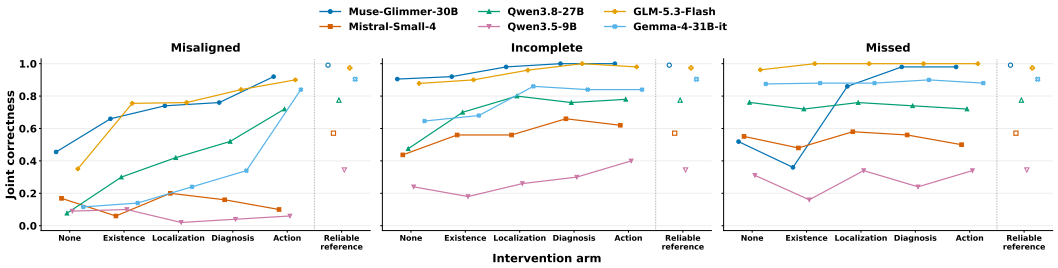


Figure 8: Joint correctness under progressively more informative interventions across the three unreliable settings.

GLM-5.3-Flash, and Gemma-4-31B-it recover substantially as the intervention becomes more specific and approach their *Reliable* references under Action; in contrast, Mistral-Small-4 and Qwen3.5-9B remain far below their references even when the corrective action is supplied. Under *Incomplete*, all six models improve between None and Action, with Action bringing performance close to the corresponding *Reliable* reference despite some non-monotonic intermediate transitions. Under *Missed*, five of the six models already perform close to their *Reliable* references under None, leaving little room for intervention. Muse-Glimmer-30B is the exception: localizing the unresolved interaction recovers most of its initial deficit, and the subsequent arms bring it to the *Reliable* level.

These patterns show that communication recovery is not governed by a single capability. *Misaligned* is the most heterogeneous case: closing the gap through Localization, Diagnosis, or Action identifies autonomous monitoring and repair selection as the bottleneck for responsive models, whereas failure that persists under Action points to downstream limitations in repair execution, belief revision, evidence integration, or underlying task solving. Under *Incomplete*, the broad recovery shows that omitted information can generally be repaired once the required response is made explicit. Finally, the near-reference None results under *Missed* indicate that an overt absence of response is usually recognized without assistance.