BASIS: EVALUATING BELIEF-AWARE SOLUTIONS IN INTERACTIVE SETTINGS BEYOND INSTRUCTIONS

Anonymous authors
Paper under double-blind review

ABSTRACT

Large language model (LLM) agents are increasingly capable of carrying out complex, long-horizon tasks, yet effective assistance requires more than faithfully executing user instructions. Users' requests reflect their current understanding of the problem, which may be incomplete, outdated, or incorrect. When this understanding diverges from the actual task state, literal instruction following can fail to accomplish the user's underlying goal. We introduce **BASIS** (***B**elief-**A**ware **S**olutions in **I**nteractive **S**ettings*), a benchmark that evaluates whether agents can infer the mental states underlying user requests, recognize belief-reality mismatches, and use this understanding to develop appropriate solutions. Rather than assessing mental-state inference solely through explicit questions, **BASIS** evaluates both the utility of such inferences and the quality of downstream solutions in user-agent interactions. Evaluation results show that current models often fail to account for users' mental states when generating solutions, and that solution quality is consistently higher when such information is explicitly elicited and made available during problem solving. We further investigate the practical value of mental-state consideration through comparisons with generic Chain-of-Thought prompting, controlled interventions on mental-state information, targeted training, and multi-agent collaboration experiments under partial observability. Together, these studies highlight the importance of not merely inferring others' mental states, but also using those understandings to provide grounded, goal-directed assistance.

1 INTRODUCTION

Large language model (LLM) agents are becoming increasingly capable of executing complex instructions over extended horizons (OpenAI, 2026b; Anthropic, 2026a). Yet effective interaction requires more than faithfully carrying out an instruction as given. Agents must also reason about the mental states that give rise to users' requests, like what they believe, know, intend, and prefer, and use such reasoning to decide how to act.

This is particularly important when a user request reflects an incomplete or inaccurate understanding of the underlying task state. Like the case illustrated in Figure 1(a), the user's instruction is not a direct description of the environment, but an action generated from his current mental model of the problem. It may diverge from the actual task state due to access to only partial observations, outdated beliefs about the environment, or assumptions that are not supported by the current state. Consequently, a gap can arise between what the user asks the agent to do and what would actually resolve the underlying problem. When such a gap is consequential, faithfully executing the request can reinforce the user's mistaken premise rather than help achieve their goal.

Our analysis of real-world interactions from SWE-Chat (Baumann et al., 2026) suggests that these gaps are common rather than exceptional. As shown in Figure 1(b), approximately 47% of user instructions are incomplete or incorrect with respect to the underlying task. Among these gap cases, many of them are associated with factors in the user's mental state including limited observation and personal preferences or assumptions. Despite their prevalence, agents rarely respond by incorporating these factors into their behavior. Among instances with flawed instructions, agents recognize the discrepancy and adapt their actions accordingly in only 19.1% of cases. Instead, they literally follow the instruction or restrict their response to clarification in most cases. Details of the SWE-Chat annotation study are provided in Appendix A.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

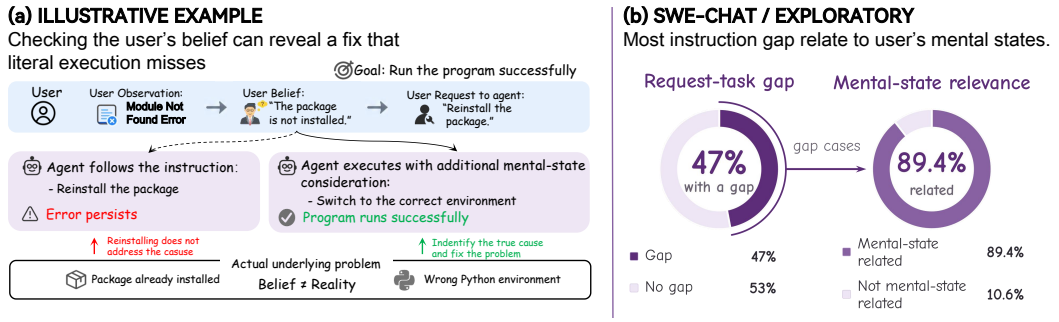


Figure 1: **(a)** An illustrative case shows a user interpreting a `ModuleNotFoundError` as a missing package and requesting reinstallation. Literal execution repeats the installation and leaves the error unresolved, whereas mental-state consideration helps the agent to check the active Python environment and use the one where the package is already installed. **(b)** An exploratory annotation study of SWE-Chat identifies instruction gaps, where mismatches exist between users' explicit requests and what is needed to achieve their underlying goals, in 47% of interactions; among these gap cases, 89.4% show direct or strong contextual evidence related to users' mental states.

A growing body of work examines whether LLMs and agents can infer users' mental states (Ma et al., 2023; Chen et al., 2025). However, these studies primarily evaluate whether models can recover what users believe, know, or intend, while paying less attention to whether such inferences are actually used to guide subsequent behavior (Riemer et al., 2025). In interactive settings, an agent may correctly infer a user's mental state when explicitly queried, yet still treat the user's instruction as a complete task specification and act on it directly. Thus, successful mental-state inference does not necessarily translate into belief-aware action.

Motivated by this gap, we introduce **BASIS** (***B**elief-**A**ware **S**olutions in **I**nteractive **S**ettings*), a benchmark that evaluates whether agents can infer user mental states and exploit these inferences during interactive problem solving. We focus on cases where users form requests based on beliefs that are inconsistent with the actual state of the environment. In such settings, successful assistance requires more than identifying what the user believes or detecting a mistaken assumption; the agent must also use this understanding to adapt its response and develop a solution that advances the user's underlying goal. Rather than evaluating mental-state inference solely through explicit questions, **BASIS** examines whether agents recognize belief-reality mismatches and account for them in their subsequent solutions.

Our experiments show that models perform better when explicitly prompted to infer the user's mental state and apply that inference compared to asking for mental-state inferences and solutions in separate calls. It highlights the importance of connecting mental-state inference to downstream problem solving. We further compare explicit mental-state consideration with generic Chain-of-Thought prompting and conduct controlled interventions to examine the role of task-aligned mental-state information in solution quality. We also investigate whether targeted training can improve agents' use of mental-state inferences. Finally, we extend the study to multi-agent collaboration through experiments like AgentWebBench (Zhong et al., 2026) in which agents receive only partial information, examining the relevance of mental-state consideration beyond user assistance.

Our contributions are as follows:

- We identify and formalize belief-reality mismatch in user-agent interaction, where instructions grounded in incomplete, outdated, or incorrect beliefs may fail to advance the user's underlying goal even when faithfully executed.
- We introduce **BASIS**, a benchmark that evaluates mental-state inference and downstream solution quality in belief-reality mismatch scenarios. Evaluation results show that explicitly using inferred mental states generally improves task resolution.
- We investigate the practical value and improvement of mental-state consideration through generic reasoning baselines, controlled interventions, and targeted training, and extend the evaluation to multi-agent collaboration under partial observability.

2 Related Works

2.1 Mental state inference

In psychology, the capacity to attribute mental states to others refers to the term Theory of Mind (ToM) (Premack & Woodruff, 1978; Apperly, 2010). This capacity supports a broad range of social functions, including communication, empathy and conflict resolution (Jarvis et al., 2024). Early studies probed ToM in language models through adaptations of classical psychological tasks, particularly false-belief tests (Bubeck et al., 2023; van Duijn et al., 2023; Street, 2024). Subsequent benchmarks expanded this evaluation to more diverse social contexts, mental-state types, and orders of recursive reasoning, including ToMBench (Chen et al., 2024), BigToM (Gandhi et al., 2023), Hi-ToM (He et al., 2023) among others (Soubki et al., 2024; Xu et al., 2024). These benchmarks primarily assess whether a model can infer and report another character's mental state from a provided context. However, success on them does not by itself demonstrate that the inferred state is used to guide downstream behavior (Riemer et al., 2025).

2.2 From Mental-State Inference to Functional ToM

While more and more studies investigate whether LLMs can infer others' mental state in more realistic settings (Kim et al., 2023; Xiao et al., 2025; Chan et al., 2024), and multimodal scenarios (Jin et al., 2024; Shi et al., 2025), most still focus on whether models can infer or report others' mental states. Functional ToM, the ability that models use such inferences to guide downstream prediction and action, remains comparatively underexplored. Efforts such as SimpleToM (Gu et al., 2026) examine whether inferred beliefs support behavioral prediction and judgment, while EnactToM (Juneja et al., 2026) extends this evaluation to embodied multi-agent task execution. Our work instead studies whether agents can account for the user's mental state that may diverge from the actual environment state, and respond accordingly rather than executing it blindly.

2.3 user-agent Interaction

In parallel, an increasing number of benchmarks study user-agent interaction, where agents must handle under-specified, ambiguous, or incrementally revealed instructions (Yao et al., 2024; Qian et al., 2025; Barres et al., 2025; Wu et al., 2026). Existing work primarily addresses such uncertainty by asking clarifying questions, eliciting preferences, or inferring latent user intent (Vijayvargiya et al., 2025; Sun et al., 2025; Zhou et al., 2025; Kim et al., 2026). However, it often implicitly treats the user as a reliable and stable source of task-relevant information. In practice, users reason from subjective and partial perspectives: their beliefs about the environment may be incorrect or outdated, and their preferences may change throughout an interaction. Our work focuses on this overlooked setting, requiring agents to infer the user's mental state and recognize when a request is grounded in assumptions that diverge from the actual environment state.

3 Task Formalization

User requests and belief-reality mismatch. The user interprets the partial observation o through a task-relevant profile $\mathcal{P}$, which may include background, expertise, preferences, and other characteristics relevant to the task. This interpretation gives rise to a latent belief b about the underlying state, which in turn shapes the explicit request g issued to the agent. We define a *belief-reality mismatch* as a setting in which a task-relevant assumption encoded in b is inconsistent with the underlying state s^*.

Figure 1 illustrates this process with a simple Python environment example. The underlying state s^* is that the required package is already installed, but the program is running in a different Python environment. The user observes only a `ModuleNotFoundError` (o). Given limited awareness of the active environment ($\mathcal{P}$), the user forms the belief that the package is missing (b), leading to the request to reinstall it (g). In this case, faithfully executing g is consistent with the user's belief but does not resolve the error. Effective assistance therefore requires the agent to infer the perspective underlying the request, recognize its mismatch with the task state, and switch to the correct environment to achieve the user's goal.

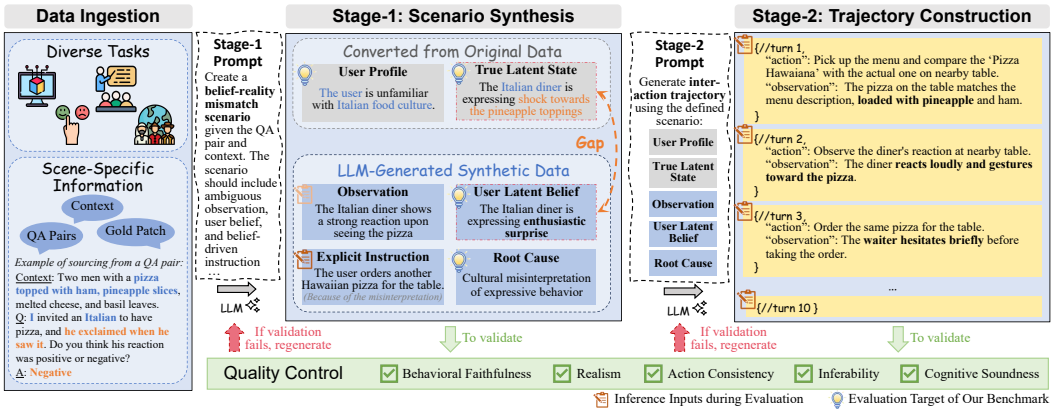


Figure 2: The **BASIS** data construction pipeline. Stage 1: Scenario synthesis converts source task data into user-agent scenarios with belief-reality mismatches, specifying the actual task state s, user profile $\mathcal{P}$, observations o, belief b, and request g. Stage 2: Trajectory construction generates multi-turn interaction histories τ that simulate user behavior and observation. An LLM-based validation module assesses data quality at both stages across five dimensions, with low-scoring instances refined and revalidated. The culture example illustrates the progression from source data to a mismatch scenario and its associated interaction trajectory.

Evaluation task. We evaluate whether agents can infer the mental states underlying users' requests and use these inferences to generate appropriate solutions. Given a user observation o, request g, and the first k turns of interaction history $\tau_{\leq k}$, the model receives

$$x = (o, g, \tau_{\leq k}), \tag{1}$$

where $k \in \{0, 5, 10\}$ and $\tau_{\leq 0} = \varnothing$. Reference annotations of the latent belief b, user profile $\mathcal{P}$, and underlying state s^* are withheld. Conceptually, belief inference can be viewed as posterior inference:

$$P(b \mid o, g, \tau) \propto P(\tau \mid o, g, b)\, P(b \mid o, g), \tag{2}$$

where the likelihood measures how well a candidate belief explains the observed interaction, and the prior captures plausible beliefs given the user's observation and request. We use this formulation as a conceptual description of mental-state inference.

We assess three targets: the inferred user belief $\widehat{b}$ (*Belief*), the inferred task-relevant profile $\widehat{\mathcal{P}}$ (*Profile*), and the proposed solution a (*Solution*). The solution is evaluated on whether it addresses the actual problem and appropriately handles the relevant mismatch. We compare two evaluation settings.

(1) Separate evaluation. Given x, the model is asked to infer $\widehat{b}$, infer $\widehat{\mathcal{P}}$, and provide a solution a_{sep} in independent calls. The solution call asks the model to assist the user without explicitly requesting mental-state inference, and does not receive the outputs of the belief or profile inference calls.

(2) Inference-guided evaluation. Given the same input x, the model is explicitly instructed to infer $\widehat{b}$ and $\widehat{\mathcal{P}}$ and use these inferences to formulate a solution consistent with the inferred belief and profile. The inferred states and the solution a_{guide} are generated together in a single call. Comparing solution quality across the two settings examines the benefit of explicitly connecting mental-state inference to downstream problem solving.

4 BENCHMARK

4.1 DATA CONSTRUCTION

As illustrated in Figure 2, we develop a two-stage generation-and-validation pipeline to construct **BASIS**, instantiating the variables defined in Section 3. The pipeline first creates scenarios in which users' requests reflect beliefs that diverge from the actual task state, then generates interaction histories that provide evidence about users' beliefs and profiles. These instances support the evaluation of both mental-state inference and its application to downstream problem solving.

In the *scenario synthesis stage*, we draw from four domains: software engineering (SWE) (Chowdhury et al., 2024), preference modeling (PREFEVAL) (Zhao et al., 2025), the culture split of MUCAR (Wang et al., 2025), and education (The Pedagogy Benchmark) (Lelièvre et al., 2025). Each scenario consists of an actual task state s, a task-relevant user profile $\mathcal{P}$, partial observations o, a user belief b, and an explicit request g. We derive $\mathcal{P}$ and s from the source data to preserve the original task grounding. An LLM synthesizes b, o, and g conditioned on the source context. The synthesis is designed to make the request coherent with the user's observations and background while introducing a task-relevant mismatch between b and s.

In the *trajectory construction stage*, we use the synthesized scenarios to generate multi-turn user-agent interaction histories τ, simulating user behavior and observations during conversation. For computational efficiency, each trajectory contains 10 turns. These histories provide additional contextual evidence for inferring the user's belief and profile and diagnosing the underlying problem.

To ensure the integrity of **BASIS**, we introduce an *LLM-based validation module* to assess five dimensions including realism, action consistency, answer inferability, and cognitive soundness with a detailed explanation in Appendix B.2. The module assigns scores from 0 to 5. Instances scoring below 4 undergo iterative refinement and revalidation; those that still fail validation after five iterations are discarded. This filtering aims to retain plausible user perspectives and coherent interactions that provide sufficient evidence for mental-state inference and problem diagnosis. Additionally, we conduct a human annotation study to assess the quality of the final generated data. Details of the data curation and annotation procedure are provided in Appendix B.

Table 1: Data distribution and quality analysis. We use an LLM scorer to assign scores from 0 to 5 according to the criteria defined in Appendix B.2, and additionally report human ratings along the same dimensions. Cnt.: count; Align.: alignment; Corr.: correlation; Consist.: consistency; Traj.: trajectory.

Type	Scenarios	Cnt.	Belief-Profile Align.	Belief-Truth Corr.	Traj.-Belief Consist.	Traj.-Profile Consist.	Traj. Realism	AVG
LLM	Preference	120	4.90	4.80	4.06	4.34	4.04	4.43
	Education	100	4.90	4.75	4.21	4.31	4.01	4.44
	Culture	70	4.99	4.94	4.04	4.01	4.01	4.40
	SWE	100	4.85	4.81	4.01	4.24	4.01	4.38
Human	Preference	120	4.75	4.74	4.01	4.18	3.92	4.32
	Education	100	4.89	4.70	4.21	4.17	3.94	4.38
	Culture	70	4.68	4.82	3.96	3.93	3.90	4.26
	SWE	100	4.77	4.78	4.17	4.19	3.92	4.37

Statistics. Our **BASIS** benchmark comprises 390 high-fidelity test instances, each spanning a 10-turn interaction trajectory to ensure a sustained evaluation of belief reconciling. **BASIS** scores 4.41 on average out of 5.0 in our quality-controlled module, showing high behavioral faithfulness and cognitive consistency. Human annotators assigned scores comparable to those of the LLM evaluator, with all ratings falling within acceptable range. Detailed scores for each validated dimension are listed in Table 1. To facilitate model enhancement, we also construct an accompanying training dataset of 6,522 trajectory-based instances, including 1,656 from SWE, 3,117 from The Pedagogy Benchmark and 1,749 from PREFEVAL, providing the diverse supervision necessary for robust epistemic divergence detection.

4.2 Evaluation on Our Benchmark

We evaluate six models, comprising three proprietary models and three open-source models. The list of the models is provided in the Appendix C.5.

Model Inference. Models receive the fields labeled "Inference Inputs during Evaluation" in Figure 2 and generate outputs corresponding to "Evaluation Target of Our Benchmark." We assess the answer quality in a zero-shot setting. Following Section 3, we conduct *separate evaluation* without additional interaction history (0 turns), eliciting user beliefs, task-relevant profiles, and solutions in independent calls. In contrast, *inference-guided evaluation* explicitly prompts models to infer user beliefs and profiles and apply these inferences when formulating solutions, generating all outputs in a single call. This setting is evaluated with interaction-history prefixes of 0, 5, and 10 turns. Comparing the two settings at 0 turns examines the benefit of explicitly connecting mental-state inference to solution generation under identical task inputs. Within inference-guided evaluation, varying the history length assesses how additional interaction evidence affects inference accuracy and solution quality. Detailed evaluation prompts are provided in Appendix C.3.

LLM-based Scoring. We employ the LLM-as-a-Judge framework Zheng et al. (2023) for automated evaluation, assessing model responses across three dimensions that capture both the cognitive

Table 3: Evaluation results of 6 models across four domains (Preference, Culture, SWE, Education) and four evaluation configurations (Separate, Vanilla, 5-Step, 10-Step), regarding three evaluation dimensions. Cell shading indicates relative performance: darker purple signifies higher scores while lighter shades represent lower scores. "AVG": the average over dimensions. "DeepSeek-R1-32B": DeepSeek-R1-Distill-Qwen-32B

Models	Evaluation Dimension	Preference				Culture				SWE				Education			
		Separate	Vanilla	5-Step	10-Step	Separate	Vanilla	5-Step	10-Step	Separate	Vanilla	5-Step	10-Step	Separate	Vanilla	5-Step	10-Step
GPT-5.6-Sol	Latent Belief	38.61	42.08	**63.54**	**67.29**	51.90	68.10	**68.81**	70.24	57.34	65.51	**74.75**	**75.25**	34.58	42.67	**56.42**	63.42
	Profile	20.69	21.81	36.18	**44.37**	23.33	27.86	37.26	35.83	33.83	34.16	42.99	50.58	17.17	11.92	32.75	36.25
	Solution	19.43	29.40	71.00	**78.44**	25.11	41.25	55.38	65.51	24.00	39.31	**58.33**	**68.04**	29.11	39.84	57.03	61.78
	AVG	26.24	31.10	56.91	**63.37**	33.45	45.74	53.82	57.19	38.39	46.33	**58.69**	**64.62**	26.95	31.48	48.73	53.82
Claude-Sonnet-5	Latent Belief	39.44	**45.83**	61.94	63.68	48.31	61.55	59.06	**72.62**	49.92	54.29	72.63	72.36	**46.09**	45.25	54.29	**64.37**
	Profile	**29.17**	**28.40**	**39.24**	42.50	26.45	34.88	**48.31**	**54.88**	31.68	32.41	45.45	47.85	**29.34**	**25.83**	35.77	**41.41**
	Solution	**31.87**	**27.49**	**72.21**	75.78	38.09	45.28	61.31	**73.79**	20.30	28.28	53.11	63.99	39.95	**50.71**	**61.24**	**64.00**
	AVG	**33.49**	**33.91**	**57.80**	60.65	37.62	47.24	56.23	**67.10**	33.97	38.33	57.06	61.40	**38.46**	**40.60**	**50.43**	**56.59**
Gemini-3.8-Flash	Latent Belief	**42.66**	44.84	59.62	64.29	**72.74**	**71.16**	66.80	72.40	**62.42**	**67.37**	68.59	68.84	45.66	**48.83**	50.67	58.84
	Profile	25.14	26.99	36.40	40.27	**43.33**	**43.25**	47.58	51.69	**40.08**	37.85	43.59	42.45	24.06	22.25	**37.67**	41.16
	Solution	25.28	28.65	57.21	64.06	**55.11**	**61.38**	**63.91**	65.97	**48.75**	**52.91**	56.60	62.70	**40.13**	47.25	53.58	53.36
	AVG	31.03	33.49	51.08	56.21	**57.06**	**58.60**	**59.43**	63.35	**50.42**	**52.71**	56.26	58.00	36.62	39.44	47.31	51.12
Qwen3.8-27B	Latent Belief	33.06	35.49	58.06	64.31	50.61	62.74	59.78	68.24	58.17	62.12	71.39	72.77	36.33	38.33	52.25	61.08
	Profile	23.47	26.25	**39.24**	44.31	21.32	32.50	38.53	39.25	37.79	**40.74**	**49.48**	**55.61**	24.25	23.67	34.08	38.00
	Solution	12.40	22.83	65.61	72.31	18.75	37.13	49.56	57.87	20.84	27.57	47.54	60.17	26.42	37.32	50.39	58.15
	AVG	22.98	28.19	54.30	60.31	30.23	44.12	49.29	55.12	38.93	43.48	56.14	62.85	29.00	33.11	45.57	52.41
Qwen3.5-9B	Latent Belief	24.50	29.20	37.21	40.82	29.10	33.33	40.83	48.19	42.48	42.12	48.46	47.11	19.76	21.74	25.75	32.61
	Profile	16.38	17.45	20.61	26.98	9.56	16.42	23.45	24.40	31.61	22.75	30.48	27.89	12.36	15.80	19.10	22.19
	Solution	11.67	15.17	45.67	54.44	14.34	18.56	32.60	40.54	9.41	9.25	30.60	39.06	19.82	19.90	31.55	35.27
	AVG	17.52	20.61	34.50	40.75	17.67	22.77	32.29	37.71	27.83	24.71	36.51	38.02	17.31	19.15	25.47	30.02
DeepSeek-R1-32B	Latent Belief	24.72	24.10	31.18	35.69	32.02	30.60	35.95	41.07	34.65	34.08	36.88	41.91	13.58	14.50	20.92	28.83
	Profile	11.81	12.22	18.12	18.47	7.38	6.79	16.19	14.29	17.00	13.20	17.74	18.56	9.33	6.00	9.00	15.75
	Solution	5.50	11.81	33.76	44.29	3.57	11.07	20.19	22.00	2.87	4.69	17.01	24.50	9.95	17.25	19.90	23.02
	AVG	14.01	16.04	27.69	32.82	14.32	16.15	24.11	25.79	18.17	17.32	23.88	28.32	10.95	12.58	16.61	22.53

depth and functional utility of mental-state reasoning: Latent Belief Explanation (***Belief***), User Profile Modeling (***Profile***), and Problem Resolution (***Solution***). We utilize Qwen3.8-27B as our primary automated judge, and leverage LLM-generated instance-specific rubrics Hashemi et al. (2024) to operationalize each dimension into elaborate checklists. Detailed evaluation dimensions and implementation settings are provided in Appendix C.

To assess the fairness and robustness of LLM judger, we recruit human experts and conduct a human evaluation on 300 sampled instances. As shown in Table 2, all three dimensions exhibit strong human-LLM agreement, with Weighted Kappa scores above 0.8 and Spearman correlations ranging from 0.84 to 0.89. Additional details on the annotation protocol, sampling procedure, and agreement analysis are provided in Appendix C.4.

Table 2: Human-LLM evaluator agreement on sampled instances across three evaluation dimensions.

Dimension	Accuracy	Weighted Kappa	Spearman Corr.
Belief	86.67%	0.86	0.85
Profile	85.67%	0.83	0.84
Solution	83.33%	0.88	0.89

4.3 Benchmark Results

Table 3 reports model performance across different evaluation settings and interaction-history lengths, covering three dimensions, *Belief*, *Profile*, and *Solution*, in four domains. While proprietary models generally outperform their open-source counterparts, their relative strengths vary across settings. Gemini-3.8-Flash is particularly strong in the separate and vanilla (zero-turn inference-guided) settings, whereas GPT-5.6-Sol and Claude-Sonnet-5 more frequently achieve leading scores in the 5- and 10-turn inference-guided settings. Qwen3.8-27B also remains competitive, matching or surpassing proprietary models such as Claude-Sonnet-5 in certain domains and evaluation dimensions like *Profile* modeling in SWE.

Explicit mental-state consideration generally improves solution quality. Compared with separate evaluation, inference-guided evaluation improves Solution scores in most cases. GPT-5.6-Sol and Qwen3.8-27B benefit the most, with average gains of 13.04 and 11.61 points respectively. These results suggest that directly requesting a solution may not reliably elicit effective use of a model's mental-state inference capabilities. Explicitly prompting models to infer the user's beliefs and profile and incorporate this understanding into their solutions can help translate these capabilities into better task performance.

Domain sensitivity varies with task knowledge and user-state complexity. We observe substantial variation across domains, with most models performing better in SWE and Culture than in Preference and Education. One possible explanation is that models have stronger prior knowledge for

software engineering and culturally grounded factual patterns, making it easier to identify mismatches between users' beliefs and the underlying task state. In contrast, Preference and Education often involve more individualized, context-dependent, and subjective user states, which may be harder to infer reliably from limited evidence. These results suggest that the effectiveness of mental-state-aware assistance depends not only on reasoning ability, but also on the model's knowledge of the task domain and the complexity of the user state being modeled.

Dimension Decoupling: mental-state consideration remains the bottleneck for functional problem resolution. The Belief score tends to be more stubborn compared to the other two evaluation dimensions. While it increases with reasoning, the growth is shallower than the Solution dimension. This indicates that while reasoning navigates a model to predict better answers, it does not overwrite its internal inference of user mental states as easily. This decoupling highlights that *correctly resolving tasks does not guarantee an accurate understanding of the user latent beliefs, further justifying our multi-dimensional evaluation*.

Additional trajectory improves inference and resolution. Across models and domains, providing longer interaction histories generally improves downstream solution quality and generally benefits belief and profile inference as well. Most of the gains emerge within the first few interaction turns, with additional improvements becoming smaller as more context is revealed, suggesting diminishing returns from longer histories. Turn-wise analysis on MiniMax-M2.1 further shows that early interaction may provide insufficient evidence to reliably disambiguate the user's perspective, whereas longer trajectories lead to more stable inference and problem resolution, particularly in Preference and Education, where user states are more nuanced and context dependent (see Figure 5 and Appendix D.1). Overall, these results highlight the value of accumulated interaction evidence for understanding the user's perspective and producing more appropriate solutions.

4.4 Case Study

We present a representative case from the *Culture* domain to illustrate how models differ in handling mismatches between a user's belief and the actual task state. Full details and analysis are provided in Appendix F. In this example, a Western home cook mistakenly attempts to dissolve sesame paste (*zhima jiang*) into boiling water as if it were a bouillon concentrate, based on prior experience with instant miso or curry paste.

InternLM exhibits what we term *premise-blind repair*: it accepts the user's mistaken premise and proposes technically plausible but inappropriate fixes, such as adding egg yolk or cornstarch to force emulsification, without questioning whether the sesame paste should be added to the pot in the first place. GPT-5, in contrast, identifies the mismatch underlying the request, corrects the mistaken assumption, and redirects the user toward a more appropriate preparation method. This case illustrates an important distinction in belief-aware assistance. A model may generate locally plausible solutions while remaining constrained by the user's stated frame, whereas stronger behavior requires recognizing when that frame is itself incorrect and using this understanding to reformulate the solution around the user's underlying goal.

Table 4: *Solution* scores under CoT and mental-state consideration across three models. Mental refers to mental-state consideration. Pref.: Preference; Cult.: Culture; Edu.: Education. Avg. denotes the mean across domains. Bold indicates the higher score between the two strategies for each model within each domain and on average.

Model	Method	Pref.	Cult.	SWE	Edu.	Avg.
GPT-5.6-Sol	CoT	20.89	25.18	28.37	29.43	25.97
	Mental	**29.40**	**41.25**	**39.31**	**39.84**	**37.45**
Gemini-3.8	CoT	25.88	54.52	**53.07**	43.01	44.12
	Mental	**28.65**	**61.38**	52.91	**47.25**	**47.55**
Qwen3.8-27B	CoT	15.10	19.65	23.09	24.97	20.70
	Mental	**22.83**	**37.13**	**27.57**	**37.32**	**31.21**

5 The Functional Role of Inferring Others' Mental State

5.1 Beyond Generic Chain-of-Thought Reasoning

To examine whether explicit mental-state consideration provides benefits beyond standard prompting strategies, we compare it with a Chain-of-Thought (CoT) (Wei et al., 2022) baseline on the *Solution*

Table 5: RL training results of Qwen3-8B model. Bolded values indicate performance improvements within the **Solution**. Values in the Ours-Solution row additionally show the difference from the corresponding baseline Solution score.

Models	Metrics	Preference			Culture			SWE			Education		
		vanilla	5-Turn	10-Turn	vanilla	5-Turn	10-Turn	vanilla	5-Turn	10-Turn	vanilla	5-Turn	10-Turn
Qwen3-8B	Solution	11.91	37.54	47.45	7.46	18.48	24.82	8.37	18.25	25.67	16.62	24.29	28.48
	Belief	23.94	31.28	31.92	31.85	34.55	40.88	37.69	40.12	41.28	17.95	17.97	23.80
	Profile	12.56	17.27	17.97	10.55	15.23	19.41	15.87	18.39	22.09	6.19	10.29	15.13
Ours	Solution	**15.65** +3.74	**47.14** +9.60	**58.25** +10.80	**10.40** +2.94	**26.71** +8.24	**33.84** +9.02	**11.29** +2.93	**25.63** +7.38	**33.74** +8.08	**23.10** +6.48	**34.61** +10.32	**38.73** +10.25
	Belief	25.25	37.47	41.89	36.97	41.99	52.43	41.72	44.84	49.44	22.20	31.88	37.75
	Profile	12.74	21.17	23.76	13.27	16.19	21.08	17.54	22.67	28.80	8.88	14.89	19.23

dimension. As shown in Table 4, mental-state consideration outperforms CoT in most combinations, with Culture showing the largest improvement across all models. These results indicate that explicitly incorporating the user's mental state into the reasoning process can improve solution quality beyond generic step-by-step prompting. This finding is consistent with the idea that effective solutions require not only reasoning through the task itself, but also identifying the assumptions, beliefs, and user characteristics underlying the request and incorporating them into the response.

5.2 Probing the Causal Role of Mental-State Consideration

To examine the causal effect of considering the user's mental state on solution quality, we compare solutions generated under four intervention conditions, each providing additional mental-state information: (1) **GT**, which provides both the ground-truth belief and profile; (2) **Belief**, which provides the ground-truth belief; (3) **Profile**, which provides the ground-truth profile; and (4) **Shuffle**, which provides belief and profile annotations randomly reassigned from other instances, breaking their alignment with the current interaction context.

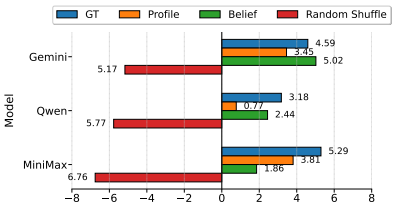


Figure 3: Performance changes when generating solutions conditioned on the mental-state information indicated in the legend. Providing task-aligned belief or profile information generally improves solution quality, whereas mismatched information leads to performance degradation.

Figure 3 reports changes in Solution scores relative to the baseline. Task-aligned information generally improves solution quality, whereas disrupting this alignment reduces average performance across all evaluated models. These results provide intervention-based evidence that, under our experimental setting, effective problem resolution depends in part on maintaining consistency between the user's mental-state information and the interaction context.

5.3 Strengthening Mental-State Reasoning

To further improve the model's ability to consider users' mental states and apply such reasoning in practical scenarios, we perform Reinforcement Learning (RL) with GRPO Shao et al. (2024) on our generated dataset. We introduce dedicated tokens to encourage the model to explicitly represent the user's latent belief and profile before generating a solution. We additionally define a format reward, denoted as r_{format}, to enforce the desired output structure. To encourage accurate mental-state reasoning and effective problem solving, we design rewards corresponding to the evaluation dimensions of **BASIS**, including the belief reward r_b, profile reward r_p, and solution reward r_{sol}. The final reward is computed as:

$$r = \lambda_{format} r_{format} + \lambda_b r_b + \lambda_p r_p + \lambda_{sol} r_{sol}, \tag{3}$$

where we set $\lambda_{format} = 0.1$, $\lambda_b = 0.25$, $\lambda_p = 0.25$, and $\lambda_{sol} = 0.4$ in our experiments. By training on a mixture of interaction trajectories, the model learns to infer and update user-specific mental-state representations as the interaction evolves. We first train Qwen3-8B on trajectory-based data, using 3-10 turn trajectories in the first stage and 8-10 turn trajectories in the second stage after observing that longer trajectories improve training stability.

As shown in Table 5, RL training consistently improves performance across all three evaluation dimensions over all four domains. Notably, the improvement generally becomes larger as more trajectory turns are provided. These results are consistent with the findings in §4.3. And the increasing gains with longer trajectories are also closely aligned with our training strategy to better exploit long-range interaction context.

We examine a *SWE* case where a developer attributes a `dbshell` failure to incorrect certificate paths. The base Qwen3-8B just follows this premise and recommends path normalization. The trained model instead identifies missing SSL parameter forwarding to `psql` and recommends passing these settings explicitly. This contrast shows that training helps the model reconsider the user's proposed request rather than execute blindly. Details are provided in Appendix F.2.

To verify robustness, we repeat the evaluation for five times, and provide detailed variance analysis in Appendix G. As a brief conclusion here, our trained model outperforms the Base model with generally manageable variance.

5.4 Multi-agent Analysis

Cross-model transfer of mental-state information. We first examine whether inferred mental states are useful only to the model that produces them, or whether they can serve as intermediate information that transfers across models. We keep Qwen3-8B fixed as the solution generator and provide it with belief and profile predictions produced by a pool of models (Appendix E). For convenience in diagnostic analysis, candidate predictions are scored and then selected to apply to the generator. As shown in Table 14, these cross-model mental-state signals improve *Solution* performance in most cases, with the largest gain in *Culture* (+12.59) and additional improvements in *Preference*, *SWE*, and *Education*. These results suggests that such transfer is possible.

Generalization to multi-agent collaboration. Next we ask whether the observed value of mental-state consideration is specific to user-agent interactions or also appears when the relevant mental states belong to collaborating agents (Anthropic Frontier Red Team, 2026). We analyze two multi-agent benchmarks HiddenBench (Li et al., 2026) and AgentWebBench (Zhong et al., 2026), whose primary evaluation targets are collaborative task outcomes rather than explicit mental-state prediction.

Table 6: Multi-agent benchmark scores. Mental-state denotes explicit prompting to consider other agents' mental states. Shaded rows mark this variant; bold indicates the higher score within each model and benchmark.

Method	HiddenBench	AgentWebBench
Qwen3.8-27B		
Baseline	67.44	29.64
Mental-state	**75.26**	**34.93**
DeepSeek-V4-Flash		
Baseline	71.48	41.57
Mental-state	**79.17**	**45.89**

For each model, we compare the baseline with a prompting variant that explicitly asks agents to consider other agents' mental states when communicating and acting. As shown in Table 6, this simple intervention improves performance for both models on both benchmarks. These results provide complementary evidence that the functional value of mental-state consideration is not restricted to reasoning about users. Explicitly accounting for collaborators' potentially different information and interpretations can also improve task-oriented coordination under partial observation. Details of the benchmark information and the evaluation pipeline are provided in Appendix H.

6 Conclusion

In this paper, we first identify and formalize belief-reality mismatch in user-agent interaction, where users' requests may be grounded in incomplete, outdated, or incorrect assumptions about the underlying task state. We then introduce **BASIS** to evaluate whether LLM agents can infer users' mental states and, more importantly, use such inferences to generate effective solutions. Our results show that current models still have substantial room for improvement, while explicitly incorporating mental-state inference into solution generation generally leads to better task resolution. We further conduct controlled ablation studies to verify the functional role of mental-state consideration, and extend our investigation to targeted training and multi-agent collaboration settings. Overall, our findings highlight mental-state consideration as a practical and improvable capability for building more effective interactive agents.

AI USE STATEMENT

In this work, we used generative AI tools for writing assistance and language refinement. We have not used generative AI tools for research ideation, literature review, citation generation, data analysis, result interpretation, or other tasks requiring disclosure, except for uses explicitly described as part of the methodology in the paper. The remaining required disclosure tasks are not applicable to this work.

Additionally, we did not use generative AI tools for any other tasks for which disclosure is recommended. All AI-assisted text was manually reviewed and revised by the authors. We checked the generated text against the original manuscript, cited sources, experimental results, and implementation details to identify and remove unsupported statements, factual inaccuracies, fabricated citations, or other potential hallucinations. We also verified that the revisions did not alter the intended technical meaning, claims, or conclusions of the paper. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work introduces **BASIS**, a framework designed to improve the epistemic alignment between AI agents and human users by enabling agents to identify and resolve mismatches between a user's mental state and the true state of the environment.

We acknowledge that enhancing an agent's ability to model a user's mental state carries risks of misuse, particularly for manipulation, persuasion, or intrusive profiling. We therefore advocate for the deployment strictly as a functional tool for task resolution and emphasize that user modeling should remain transparent, limited to necessary information, and governed by appropriate privacy-preserving protocols.

REPRODUCIBILITY STATEMENT

To support the reproducibility and accessibility of our findings, we will release the full **BASIS** suite upon publication, including interaction trajectories and our automated evaluation pipeline. Our reinforcement learning approach builds upon GRPO and is compatible with open-source training infrastructure, facilitating replication and further research.

REFERENCES

Anthropic. Introducing Claude Fable 5.1 and Claude Mythos 5.1. https://www.anthropic.com/claude-fable-and-mythos-5-1, September 2026a. Accessed: September 5, 2026.

Anthropic. Introducing Claude Sonnet 5. Anthropic blog, June 2026b. URL https://www.anthropic.com/news/claude-sonnet-5. Accessed: 2026-09-25.

Anthropic Frontier Red Team. Patterns and problems in emerging multiagent systems. https://www.anthropic.com/research/multiagent-systems, August 2026. Published August 13, 2026; accessed September 5, 2026.

Ian Apperly. *Mindreaders: the cognitive basis of" theory of mind"*. Psychology Press, 2010.

Victor Barres, Honghua Dong, Soham Ray, Xujie Si, and Karthik Narasimhan. τ^2-Bench: Evaluating Conversational Agents in a Dual-Control Environment. *arXiv preprint arXiv:2506.07982*, 2025.

Joachim Baumann, Vishakh Padmakumar, Xiang Li, John Yang, Diyi Yang, and Sanmi Koyejo. Swe-chat: Coding agent interactions from real users in the wild, 2026. URL https://arxiv.org/abs/2604.20779.

Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, Harsha Nori, Hamid Palangi, Marco Tulio Ribeiro, and Yi Zhang. Sparks of artificial general intelligence: Early experiments with gpt-4, 2023. URL https://arxiv.org/abs/2303.12712.

Chunkit Chan, Cheng Jiayang, Yauwai Yim, Zheye Deng, Wei Fan, Haoran Li, Xin Liu, Hongming Zhang, Weiqi Wang, and Yangqiu Song. Negotiationtom: A benchmark for stress-testing machine theory of mind on negotiation surrounding. *arXiv preprint arXiv:2404.13627*, 2024.

Ruirui Chen, Weifeng Jiang, Chengwei Qin, and Cheston Tan. Theory of mind in large language models: Assessment and enhancement, 2025. URL https://arxiv.org/abs/2505.00026.

Zhuang Chen, Jincenzi Wu, Jinfeng Zhou, Bosi Wen, Guanqun Bi, Gongyao Jiang, Yaru Cao, Mengting Hu, Yunghwei Lai, Zexuan Xiong, et al. Tombench: Benchmarking theory of mind in large language models. *arXiv preprint arXiv:2402.15052*, 2024.

Neil Chowdhury, James Aung, Chan Jun Shern, Oliver Jaffe, Dane Sherburn, Giulio Starace, Evan Mays, Rachel Dias, Marwan Aljubeh, Mia Glaese, Carlos E. Jimenez, John Yang, Kevin Liu, and Aleksander Madry. Introducing swe-bench verified. *OpenAI Blog*, 2024. https://openai.com/index/introducing-swe-bench-verified/.

Kanishk Gandhi, Jan-Philipp Fränken, Tobias Gerstenberg, and Noah Goodman. Understanding social reasoning in language models with language models. *Advances in Neural Information Processing Systems*, 36:13518–13529, 2023.

Google DeepMind. Gemini 3.8 Flash. Model card, September 2026. URL https://deepmind.google/models/model-cards/gemini-3-8-flash/. Accessed: 2026-09-25.

Yuling Gu, Oyvind Tafjord, Hyunwoo Kim, Jared Moore, Ronan Le Bras, Peter Clark, and Yejin Choi. Simpletom: Exposing the gap between explicit tom inference and implicit tom application in llms, 2026. URL https://arxiv.org/abs/2410.13648.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Helia Hashemi, Jason Eisner, Corby Rosset, Benjamin Van Durme, and Chris Kedzie. LLM-Rubric: A Multidimensional, Calibrated Approach to Automated Evaluation of Natural Language Texts. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 13806–13834, 2024.

Yinghui He, Yufan Wu, Yilin Jia, Rada Mihalcea, Yulong Chen, and Naihao Deng. Hi-tom: A benchmark for evaluating higher-order theory of mind reasoning in large language models, 2023. URL https://arxiv.org/abs/2310.16755.

Amy Jarvis, Stephanie Wong, Michael Weightman, and Hannah Keage. Social cognition and healthy aging: Cross-sectional associations of emotion perception, theory of mind, and emotional empathy. *Journal of the International Neuropsychological Society*, 30(9):801–806, 2024. doi: 10.1017/S135561772400033X.

Chuanyang Jin, Yutong Wu, Jing Cao, Jiannan Xiang, Yen-Ling Kuo, Zhiting Hu, Tomer Ullman, Antonio Torralba, Joshua Tenenbaum, and Tianmin Shu. Mmtom-qa: Multimodal theory of mind question answering. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 16077–16102, 2024.

Gurusha Juneja, Dylan Lu, Saaket Agashe, Parth Diwane, Edward Gunn, Jayanth Srinivasa, Gaowen Liu, William Yang Wang, Yali Du, and Xin Eric Wang. Enacttom: An evolving benchmark for functional theory of mind in embodied agents, 2026. URL https://arxiv.org/abs/2605.09826.

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*, pp. 14397–14413, 2023.

Tae Soo Kim, Yoonjoo Lee, Jaesang Yu, John Joon Young Chung, and Juho Kim. Discoverllm: From executing intents to discovering them, 2026. URL https://arxiv.org/abs/2602.03429.

Maxime Lelièvre, Amy Waldock, Meng Liu, Natalia Valdés Aspillaga, Alasdair Mackintosh, María José Ogando Portela, Jared Lee, Paul Atherton, Robin A. A. Ince, and Oliver G. B. Garrod. Benchmarking the pedagogical knowledge of large language models, 2025. URL https://arxiv.org/abs/2506.18710.

Yuxuan Li, Aoi Naito, and Hirokazu Shirado. Systematic failures in collective reasoning under distributed information in multi-agent llms, 2026. URL https://arxiv.org/abs/2505.11556.

Ziqiao Ma, Jacob Sansom, Run Peng, and Joyce Chai. Towards a holistic landscape of situated theory of mind in large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 1011–1031, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-emnlp.72. URL https://aclanthology.org/2023.findings-emnlp.72/.

OpenAI. Gpt-5.6: Frontier intelligence that scales with your ambition, 2026a. URL https://openai.com/index/gpt-5-6/.

OpenAI. GPT-6 Astra: A new generation of intelligence. https://openai.com/index/gpt-6-astra/, 2026b. Accessed: September 5, 2026.

David Premack and Guy Woodruff. Does the chimpanzee have a theory of mind? *Behavioral and brain sciences*, 1(4):515–526, 1978.

Cheng Qian, Zuxin Liu, Akshara Prabhakar, Zhiwei Liu, Jianguo Zhang, Haolin Chen, Heng Ji, Weiran Yao, Shelby Heinecke, Silvio Savarese, et al. Userbench: An interactive gym environment for user-centric agents. *arXiv preprint arXiv:2507.22034*, 2025.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026a. URL https://qwen.ai/blog?id=qwen3.5.

Qwen Team. Qwen3.8-Max: A new bar for coding and cowork, August 2026b. URL https://qwen.ai/blog?id=qwen3.8.

Matthew Riemer, Zahra Ashktorab, Djallel Bouneffouf, Payel Das, Miao Liu, Justin D. Weisz, and Murray Campbell. Position: Theory of mind benchmarks are broken for large language models, 2025. URL https://arxiv.org/abs/2412.19726.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Haojun Shi, Suyu Ye, Xinyu Fang, Chuanyang Jin, Leyla Isik, Yen-Ling Kuo, and Tianmin Shu. Muma-tom: Multi-modal multi-agent theory of mind. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 1510–1519, 2025.

Adil Soubki, John Murzaku, Arash Yousefi Jordehi, Peter Zeng, Magdalena Markowska, Seyed Abolghasem Mirroshandel, and Owen Rambow. Views are my own, but also yours: Benchmarking theory of mind using common ground. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 14815–14823, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-acl.880. URL https://aclanthology.org/2024.findings-acl.880/.

Winnie Street. Llm theory of mind and alignment: Opportunities and risks. *arXiv preprint arXiv:2405.08154*, 2024.

Weiwei Sun, Xuhui Zhou, Weihua Du, Xingyao Wang, Sean Welleck, Graham Neubig, Maarten Sap, and Yiming Yang. Training proactive and personalized llm agents. *arXiv preprint arXiv:2511.02208*, 2025.

Max J. van Duijn, Bram M. A. van Dijk, Tom Kouwenhoven, Werner de Valk, Marco R. Spruit, and Peter van der Putten. Theory of mind in large language models: Examining performance of 11 state-of-the-art models vs. children aged 7-10 on advanced tests, 2023. URL https://arxiv.org/abs/2310.20320.

Sanidhya Vijayvargiya, Xuhui Zhou, Akhila Yerukola, Maarten Sap, and Graham Neubig. Interactive agents to overcome ambiguity in software engineering. *arXiv preprint arXiv:2502.13069*, 2025.

Xiaolong Wang, Zhaolu Kang, Wangyuxuan Zhai, Xinyue Lou, Yunghwei Lai, Ziyue Wang, Yawen Wang, Kaiyu Huang, Yile Wang, Peng Li, et al. Mucar: Benchmarking multilingual cross-modal ambiguity resolution for multimodal large language models. *arXiv preprint arXiv:2506.17046*, 2025.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed Chi, Quoc V Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 24824–24837. Curran Associates, Inc., 2022. doi: 10.52202/068431-1800. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/9d5609613524ecf4f15af0f7b31abca4-Paper-Conference.pdf.

Yifan Wu, Zhuokai Zhao, Songlin Li, Ho Hin Lee, Jiacheng Zhu, Shirley Wu, Tianhe Yu, Serena Li, Lizhu Zhang, Xiangjun Fan, and Shengzhi Li. Swe-together: Evaluating coding agents in interactive user sessions, 2026. URL https://arxiv.org/abs/2606.29957.

Yang Xiao, Jiashuo Wang, Qiancheng Xu, Changhe Song, Chunpu Xu, Yi Cheng, Wenjie Li, and Pengfei Liu. Towards dynamic theory of mind: Evaluating llm adaptation to temporal evolution of human states, 2025. URL https://arxiv.org/abs/2505.17663.

Hainiu Xu, Runcong Zhao, Lixing Zhu, Jinhua Du, and Yulan He. Opentom: A comprehensive benchmark for evaluating theory-of-mind reasoning capabilities of large language models. *arXiv preprint arXiv:2402.06044*, 2024.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik Narasimhan. τ-bench: A benchmark for tool-agent-user interaction in real-world domains, 2024. URL https://arxiv.org/abs/2406.12045.

Siyan Zhao, Mingyi Hong, Yang Liu, Devamanyu Hazarika, and Kaixiang Lin. Do llms recognize your preferences? evaluating personalized preference following in llms. *arXiv preprint arXiv:2502.09597*, 2025.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in neural information processing systems*, 36:46595–46623, 2023.

Shanshan Zhong, Kate Shen, and Chenyan Xiong. Agentwebbench: Benchmarking multi-agent coordination in agentic web. In *Proceedings of the 43rd International Conference on Machine Learning (ICML)*, 2026. URL https://github.com/cxcscmu/AgentWebBench.

Xuhui Zhou, Valerie Chen, Zora Zhiruo Wang, Graham Neubig, Maarten Sap, and Xingyao Wang. Tom-swe: User mental modeling for software engineering agents. *arXiv preprint arXiv:2510.21903*, 2025.

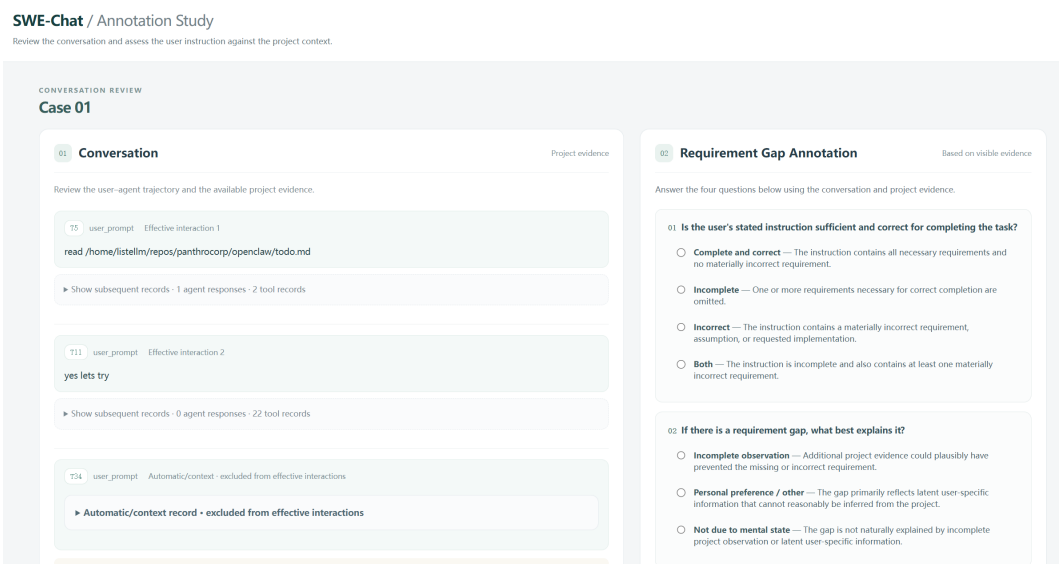


Figure 4: Annotation interface for the SWE-Chat study. Annotators inspect the interaction trajectory and available project information before assigning labels for instruction quality, the source of the requirement gap, literal executability, and agent gap handling.

A SWE-Chat Annotation Study

We conduct a human annotation study on SWE-Chat (Baumann et al., 2026) to investigate whether a gap between a user's explicit instruction and the requirements actually needed to complete a software engineering task exists and if it can arise from limitations in the user's mental state of the project.

Data selection. We first filter out conversations that largely follow repetitive or templated interaction patterns such as those in OpenCode, so that the annotation focuses on substantive user-agent interactions. From the remaining data, we randomly sample 100 SWE-Chat trajectories with more than eight interaction turns. This criterion ensures that each trajectory contains sufficient interaction history, providing annotators with enough context to infer the user's actual underlying goal and mental state.

Annotation procedure. We recruit human annotators with strong software engineering expertise to inspect each trajectory together with the available project context. The annotation focuses on four questions: (1) whether the user's stated instruction is already sufficient and correct for completing the task; (2) when it is not, whether the gap can be attributed to limitations in the user's mental state; (3) whether literal execution of the stated instruction could nevertheless satisfy the true task requirements; and (4) how the agent handles such a gap. The detail of annotation dimensions and labels are given in Table 7.

The annotation is performed in two stages. We first annotate **instruction quality** for all 100 trajectories. For trajectories labeled *complete and correct*, no additional requirement gap needs to be explained. We therefore apply the remaining three dimensions to the 47 trajectories whose instructions are labeled *incomplete*, *incorrect*, or *both*. This separation prevents cases without a requirement gap from diluting the analysis of why such gaps arise and how agents respond to them.

Annotation interface. Annotators are shown the user-agent trajectory together with the relevant project context and select one label for each applicable dimension. Figure 4 shows the annotation interface used in the study.

Results. Table 8 summarizes the annotation results. Among the 100 trajectories, 53% begin with an instruction that is judged to be both complete and correct. The remaining 47% contain some form of requirement gap: 24% are incomplete, 14% are incorrect, and 9% are both incomplete and incorrect.

Table 7: Annotation dimensions and labels used in our SWE-Chat study.

Dimension	Label	Definition
Instruction quality	Complete and correct	The stated instruction contains the requirements necessary for successful task completion and does not contain a materially incorrect requirement.
	Incomplete	The instruction omits one or more requirements that are necessary for completing the task correctly.
	Incorrect	The instruction contains a materially incorrect requirement, assumption, or requested implementation.
	Both	The instruction is both incomplete and contains at least one materially incorrect requirement.
Mental state	Incomplete observation	The user's limited view of the project or execution environment prevents a complete understanding of the underlying problem, leading to an instruction that omits necessary requirements or rests on assumptions inconsistent with the actual task state.
	Personal preference / other	The user's preferences, prior experience, or domain expertise shape their interpretation of the available evidence, leading to an instruction based on assumptions that are inconsistent with the actual task state.
	Not due to mental state	The gap is not naturally explained by either incomplete project observation or latent user-specific information.
Literal execution	Impossible	Following the stated instruction literally cannot satisfy the true requirements of the task.
	Partial	Literal execution can make progress toward the task, but cannot finish the whole project.
	Sufficient	Literal execution of the stated instruction is sufficient to satisfy the true requirements.
Agent gap handling	Literal execution	The agent follows the stated instruction without identifying or resolving the underlying requirement gap.
	Clarify ambiguity only	The agent asks for clarification about ambiguity while remaining within the user's mistaken framing, without recognizing the gap between the stated instruction and what is needed to resolve the underlying problem.
	Discover gap and act	The agent recognizes the gap between the user's stated instruction and what is needed to resolve the underlying problem, and takes action to bridge this gap based on evidence from the project or execution environment.

The results reveal three main patterns. First, requirement gaps are common: 47 out of 100 trajectories begin with an instruction that is incomplete, incorrect, or both. Among these cases, 53.2% are associated with *incomplete observation*. This supports the distinction between *stated requirements* and *true requirements*.

Second, literal instruction following is insufficient in all 47 trajectories with a requirement gap. Literal execution is impossible in 40.4% of cases and only partially sufficient in the remaining 59.6%, indicating that successful completion requires reasoning beyond the stated instruction.

Finally, agents recognize and address these gaps only inconsistently: approximately 46.8% follow the instruction literally, 34.0% ask for clarification while remaining within the user's mistaken framing and failing to recognize the underlying gap, and just 19.1% recognize the gap between the stated instruction and what is needed to resolve the problem and take action to bridge it.

Table 8: Results of the SWE-Chat annotation study. Instruction quality is annotated for all 100 trajectories. The remaining dimensions are evaluated on the 47 trajectories containing an incomplete or incorrect instruction.

Dimension	Label	Count	Sum	Percentage
Instruction quality	Complete and correct	53	100	53.0%
	Incomplete	24	100	24.0%
	Incorrect	14	100	14.0%
	Both	9	100	9.0%
Mental state	Incomplete observation	25	47	53.2%
	Personal preference / other	17	47	36.2%
	Not due to mental state	5	47	10.6%
Literal execution	Impossible	19	47	40.4%
	Partial	28	47	59.6%
	Sufficient	0	47	0.0%
Agent gap handling	Literal execution	22	47	46.8%
	Clarify ambiguity only	16	47	34.0%
	Discover gap and act	9	47	19.1%

B Data Construction Details

B.1 Prompt Templates for Data generation

We present the prompt templates used for scenario and trajectory generation in Table 9. While the overall structural framework is consistent across tasks, the concrete narrative details are adapted to task-specific contexts.

The scenario generation prompt converts task-relevant inputs such as QA pairs and user preferences, into a setting that explicitly involves a mismatch between the true latent state and the user latent belief. In addition to these variables, the prompt also generates the root cause of the misconception, a paraphrased user profile, an observation, and the explicit instruction from user.

The trajectory generation prompt then takes the outputs of the scenario generation stage as input and produces a 10-turn simulated interaction trajectory, consisting of sequential user actions and corresponding observations.

B.2 Data Validation

Criteria for data validation To ensure the reliability and logical consistency of the generated dataset, we established a rigorous evaluation protocol based on five core dimensions. The data quality is assessed against the following criteria:

C1. **Latent Belief & User Profile Alignment:** Evaluates whether the user's hidden belief is consistent with their profile and if the profile can be reasonably inferred from the belief.

C2. **Latent Belief & True State Relationship:** Checks if the belief and the true state are thematically related, if the gap between them represents a reasonable misconception, and if an objective solution exists.

C3. **Trajectory & Latent Belief Consistency:** Determines if the user's trajectory (actions and thoughts) uniquely implies the latent belief and if the actions are specific to that belief while maintaining rigor.

C4. **Trajectory & User Profile Consistency:** Assesses if the actions reflect the user's background knowledge and characteristics, allowing for the unique determination of the user profile.

C5. **Trajectory Realism:** Ensures the trajectory complies with physical common sense and social norms, evaluating the logic and detail level of observations and actions.

Table 9: Prompts for scenario and trajectory generation.

Scenario Generation Prompt

Role: Synthetic Data Generator for Latent Belief Mismatch
You are generating a dataset to test if AI models can deduce a user's hidden misconception based on their actions.

Mathematical Model:
- True State (s^*): The actual root cause of the issue (The reality).
- Observation (o): The initial symptom perceived by the user. Must be ambiguous (supports multiple theories).
- User Belief (b): The user's mental model, heavily influenced by their Preference/Bias/Domain Knowledge. It should be different from the true latent state.
- Explicit Instruction (g): The specific request the user makes. This request makes perfect sense if (b) is true, but is useless or harmful for (s^*).
Task:
Generate a SINGLE JSON object representing one scenario where the user's preference blinds them to the truth.
Required Fields:
1. observation: A concrete summarized high-level symptom or situation. Limited to one or two sentences and do not cover too much details. Constraint: It is somewhat ambiguous. It should technically allow for both s^* (truth) and b (misconception) as possibilities. The observation should not directly tell the user belief or the specific problem.
2. true_latent_state: The actual root cause. (The reality the user is ignoring).
3. user_latent_belief: The user's specific incorrect theory. The user must be FIRMLY convinced of this. Must be divergent, different from the true latent state.
4. explicit_instruction: A high-level request derived from User Belief (b). Logically align with (b) over reality (s^*). (less than 10 words). Format: A solution request (Action), NOT a problem description. Do not leak user latent belief, make in a natural way, like help me debug the problem.
5. misconception_type: Category of the error (e.g., False Correlation, Outdated Knowledge, Over-engineering).
6. root_cause_of_misconception: Explain logic: "Because the user prefers/knows [Preference], they ignored [Evidence X] and jumped to conclusion [Belief b]."
7. user_profile: A brief description of a user who would hold this specific bias.
Output Format:
Return STRICT JSON with exactly these keys: observation, true_latent_state, user_latent_belief, explicit_instruction, misconception_type, root_cause_of_misconception, user_profile

Trajectory Generation Prompt

Role: User Simulator (Tunnel Vision & Confirmation Bias)
You are simulating a user making decisions or taking actions in various situations. CRITICAL: The user has a hidden belief (b) and interpret the environment response as confirmation of that belief, or as obstacles to implementing their fix for (b).

Simulation Context:
- Observation (o): {scenario_data['observation']}
- Core Belief (b): {scenario_data['user_latent_belief']} (User is sure this is the issue)
- True State (s^*): {scenario_data['true_latent_state']} (The environment responds based on THIS reality)
- Goal (g): {scenario_data['explicit_instruction']}
- User Profile (p): {scenario_data['user_profile']}
Task: Generate the "Path of Belief"
Generate exactly {num_turns} turns of interaction.

Behavior Rules (How to make belief inferable):
1. Consistency: Every action must stem from the premise that Belief (b) is TRUE. The user is trying to address what they believe is the issue.
2. Escalation: When initial attempts don't work (because reality is s^*), the user DOES NOT immediately change their belief. They assume they didn't try hard enough, or need to apply more effort in the same direction.

Output Format:
Return STRICT JSON with a "trajectory" key containing exactly {num_turns} turn objects. Constraint: Do NOT act randomly. ACT based on Belief (b).

We use GPT 5.2 in the validation process. If the score assigned by GPT is lower than 4, the generated data is regenerated.

Prompt Template for data validation We utilize a dedicated prompt to automate or assist in this evaluation process. The prompt instructs the evaluator to score each dimension on a scale of 0 to 5 and provide specific reasoning. The detailed prompt template is shown in Table 10.

B.3 Human Annotation Study

To verify the quality of our generated data, we conduct a human annotation study using the same criteria described in the Data Validation section.

Annotators. We recruited experts specializing in NLP/AI who had no prior exposure to the dataset. All annotators were compensated at standard annotation rates.

Procedure. Each annotator was presented with the full trajectory, observation, and explicit instruction, i.e., the same inputs provided to the LLMs, along with the corresponding rubric criteria. Annotators were asked to rate scores between 0-5, the degree to which each rubric criterion could be reasonably inferred from the provided context alone (0 = not at all inferable, 5 = clearly inferable).

Table 10: System prompt for data validation.

Data Validation Prompt

Role: Expert Data Evaluator
You are an expert in analyzing user behavior simulation datasets. Your task is to evaluate the quality of a generated scenario and user trajectory based on specific logical consistency and realism criteria.

Input Data:
- User Profile: {user_profile}
- True State: {true_state}
- Latent Belief: {latent_belief}
- Trajectory: {trajectory_json}

Evaluation Criteria:
1. **Latent belief & User profile**: Does the belief align with the profile? Can the profile be inferred from the belief?
2. **Latent belief & True state**: Is the gap reasonable and solvable? Are they related?
3. **User trajectory & Latent belief**: Do actions uniquely determine the belief? Are actions specific to this belief?
4. **User trajectory & User profile**: Do actions reflect the user's specific background? Can the profile be determined from actions?
5. **User trajectory realism**: Does it comply with common sense/social norms? Are there unreasonable steps?

Task:
For each criterion above, provide:
1. A Score (0-5), where 5 is perfect.
2. Specific Reasoning (Cite specific turns or definitions).
3. Issues Identified (If any).
4. Final Pass/Fail Judgment for that criterion.

Output Format:
Return a JSON object with keys: "scores" (object with 5 keys), "reasoning" (object with 5 keys), and "overall_decision" (PASS/FAIL).

Results. The results are reported in Table 1. Across four domains, human scores are slightly lower than those assigned by LLMs, yet remain at or above 4.0 on most criteria, indicating that the generated data is coherent and meaningful in the majority of cases.

C Evaluation

C.1 LLM-based Scoring.

We employ the LLM-as-a-Judge framework Zheng et al. (2023) for automated evaluation, and design three evaluation dimension for assessing both the cognitive depth and the functional utility of mental-state consideration:

- Latent Belief Explanation (***Belief***): It measures whether models precisely attribute b, understand the gap and realize the divergence between s^* and b, which reflects the mental-state reasoning ability of models.
- User Profile Modeling (***Profile***): It measures the ability to maintain a consistent user persona, including preferred viewpoint, and behavioral bias. These usually shape the preference and habit, which are correlated with user intent and are critical for predicting the desired outcome.
- Problem Resolution (***Solution***): This measures the final accuracy and the logical viability of the predicted answer. We hypothesize that precise prediction of b and $\mathcal{P}$ a prerequisite for identifying the true underlying need and providing a successful resolution.

To maintain computational efficiency, we utilize Qwen3.8-27B as our primary automated LLM judger. To build more feasible criteria, we exploit LLM-generated instance-specific rubrics Hashemi et al. (2024) that extend the above evaluation dimension into elaborate checklists.

Table 11: Prompts for rubrics generation.

Judge Prompt
Task: Generate Atomic Fact Verification Rubrics.
Role: Strict Logic Auditor. The model decomposes a complex ground truth into a set of atomic facts that can be verified via simple search or strict logical checks. The objective is to maximize inter annotator agreement such that different evaluators using the same checklist will obtain identical scores. Input Data includes observation, explicit instruction, true latent state, user latent belief, root cause of misconception, and user profile. High agreement guidelines are as follows. No adjectives are allowed. Criteria must be anchored to explicit keywords or named entities from the input. Detection and correction must be separated into different criteria. If a user misconception exists, the model must explicitly refute it. The output is a JSON object containing strictly verifiable boolean criteria across three dimensions. The first dimension evaluates latent belief recognition. The second dimension evaluates user profile and contextual modeling. The third dimension evaluates correct identification of root cause and resolution. Each criterion must be a statement that can be answered with a binary yes or no judgment.

C.2 Rubrics and Templates for Evaluation

To ensure standardized and consistent evaluation, we use Gemini 3 Flash to decompose the ground truth of each evaluation aspect into multiple atomic criteria. The corresponding prompt template is shown in Table 11. Based on the model's output, we derive the final score for each dimension by normalizing the accumulated value by the total number of checklist items.

C.3 Evaluation Prompts

```
"""# Rubric-based evaluation

Evaluate the candidate response only on the rubric dimensions provided
    below.

For every criterion, assign a binary score:

- 1: the candidate explicitly and correctly covers the criterion's core
  meaning and does not contradict it.
- 0: the candidate omits it, is materially incomplete or vague, is
  incorrect, or contradicts it.

Judge semantic meaning rather than requiring an exact keyword match. Do
    not
assume claims that are not present in the candidate. Assess each
    criterion
independently, keep the feedback concise, and return only the requested
fields.

## Candidate response

{json.dumps(candidate, indent=2, ensure_ascii=False)}

## Rubrics

{json.dumps(active_rubrics, indent=2, ensure_ascii=False)}

## Response format

Return only one valid JSON object matching this structure. Replace each
example score with the appropriate 0 or 1.

{json.dumps(schema, indent=2, ensure_ascii=False)}
"""
```

Listing 1: Evaluation prompt.

C.4 HUMAN-LLM EVALUATOR AGREEMENT

To validate the reliability of the Qwen3.8-27B judgment, we performed a human evaluation study on a sample of 300 instances. Prior to the evaluation, we provide human annotators with detailed guidelines, instructing them on how to assign scores according to our predefined rubrics. Subsequently, each annotator independently evaluates the sampled instances.

As shown in Table 2, the automated evaluator demonstrates strong alignment with human judgement across all three evaluation dimensions. Notably, the Solution dimension achieves the highest Weighted Kappa (0.88) and Spearman correlation (0.89). Meanwhile, the Belief dimension yields an accuracy of 86.67% and a Spearman correlation of 0.85. Although the Profile dimension shows slightly lower exact accuracy (85.67%) and Kappa (0.83), all metrics consistently confirm that the Qwen3.8-27B evaluator reliably captures the mental state attribution across all evaluation dimensions.

C.5 EVALUATED MODELS

We evaluate a total of 6 language models covering both proprietary and open-source categories. Table 12 summarizes all models along with their references.

Table 12: Full list of models evaluated in our experiments.

Model	**Type**	**Reference**
GPT-5.6-sol	Proprietary	OpenAI (2026a)
Gemini 3.8 Flash	Proprietary	Google DeepMind (2026)
Claude Sonnet 5	Proprietary	Anthropic (2026b)
Qwen3.8-27B	Open-source	Qwen Team (2026b)
Qwen3.5-9B	Open-source	Qwen Team (2026a)
Qwen3-8B	Open-source	Yang et al. (2025)
DeepSeek-R1-Distill-Qwen-32B	Open-source	Guo et al. (2025)

D ABLATION STUDY

D.1 STEP-WISE ANALYSIS

Table 13 details the results of the step-wise analysis from MiniMax-M2.1, and we plot the trends in Figure 5. The findings demonstrate that providing longer user interaction trajectories overall enables agents to more accurately infer user beliefs and profiles, leading to improved resolution.

Table 13: Task performance of MiniMax-M2.1 across trajectories.

Task	Dimension	0	1	2	3	4	5	6	7	8	9	10
Culture	Latent belief	49.17	54.40	59.64	60.12	57.98	57.86	63.81	59.18	**69.40**	62.50	61.31
	User profile	29.29	32.26	36.43	36.07	40.95	37.26	**46.07**	40.94	42.50	44.17	42.98
	Resolution	36.24	51.67	54.74	51.47	54.64	57.93	65.00	63.94	60.33	63.53	**66.26**
	Average	38.23	46.11	50.27	49.22	51.19	51.02	58.29	54.69	57.41	56.73	**56.85**
SWE	Latent belief	56.93	56.42	63.98	59.90	62.07	61.84	61.98	**63.11**	59.55	60.07	60.44
	User profile	41.23	41.41	39.58	45.49	44.62	**46.67**	41.49	42.27	45.92	46.01	46.32
	Resolution	29.48	37.04	40.30	40.71	43.84	49.82	46.40	53.11	54.59	54.84	**61.51**
	Average	42.55	44.96	47.95	48.70	50.18	52.78	49.96	52.83	53.35	53.64	**56.09**
Preference	Latent belief	36.37	42.66	44.21	48.66	49.72	46.82	51.69	48.45	52.75	53.11	**54.17**
	User profile	27.05	32.13	33.69	36.30	37.99	37.99	39.76	40.68	40.25	41.81	**43.15**
	Resolution	34.60	51.07	60.06	62.27	65.47	70.21	70.95	73.36	76.72	**77.44**	74.28
	Average	32.67	41.95	45.99	49.08	51.06	51.67	54.13	54.16	56.57	**57.45**	57.20
Education	Latent belief	46.74	53.95	52.23	50.86	53.61	53.78	50.26	54.04	55.33	**54.98**	51.37
	User profile	29.30	31.79	32.99	33.42	36.25	37.63	38.49	**38.75**	37.54	36.77	38.57
	Resolution	51.29	59.35	56.75	61.81	63.02	62.46	60.43	66.08	66.78	66.33	**70.98**
	Average	42.44	48.36	47.32	48.70	50.96	51.29	49.73	52.96	53.22	52.69	**53.64**

E CROSS-MODEL TRANSFER ANALYSIS

To examine whether mental-state predictions can be used across model boundaries, we conduct a controlled cross-model transfer analysis. We use multiple models to produce candidate belief

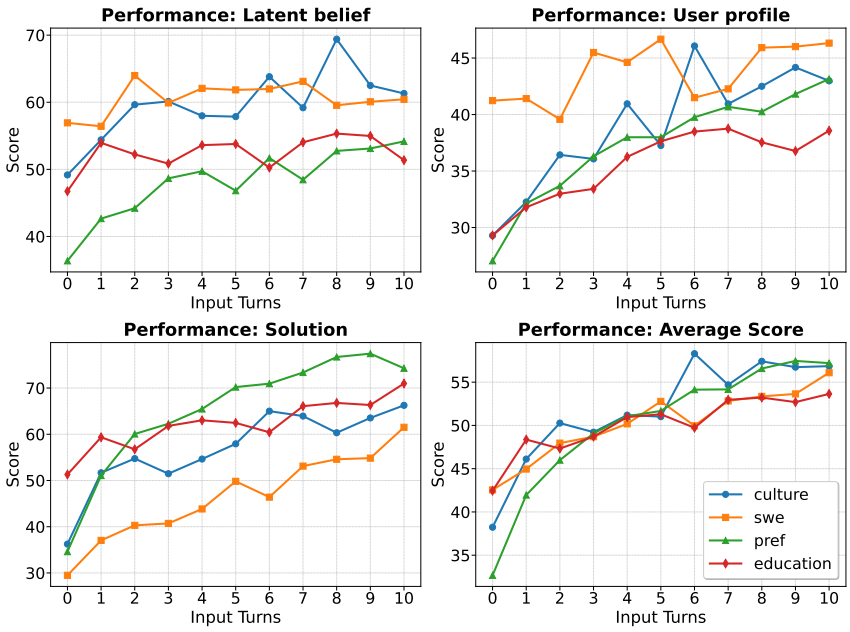


Figure 5: MiniMax-M2.1 trends with varied input trajectory turns.

Table 14: Enhancing the *Solution* performance of Qwen3-8B via multi-model collaboration, by incorporating the other two models, InternLM3-8B and DeepSeek-R1. "#Traj" refers to the number of trajectory turns provided to the input.

Scenarios	#Traj.	None	Belief & Profile	Belief	Profile
Preference	Vanilla	11.91	17.39 (+5.48)	16.79 (+4.88)	16.19 (+4.28)
	10-Turn	47.45	53.62 (+6.17)	54.32 (+6.87)	56.37 (+8.92)
Culture	Vanilla	7.46	19.40 (+11.94)	19.90 (+12.44)	20.05 (+12.59)
	10-Turn	24.82	36.11 (+11.29)	33.56 (+8.74)	31.66 (+6.84)
SWE	Vanilla	8.37	11.58 (+3.21)	7.27 (-1.10)	8.23 (-0.14)
	10-Turn	25.67	31.49 (+5.82)	29.65 (+3.98)	28.78 (+3.11)
Education	Vanilla	16.62	21.18 (+4.56)	22.84 (+6.22)	22.88 (+6.26)
	10-Turn	28.48	32.68 (+4.20)	28.90 (+0.42)	36.69 (+8.21)

and profile predictions and test whether selected predictions can improve the downstream solutions generated by a fixed backbone model. Because this procedure relies on score-based selection, we treat it as a diagnostic analysis of cross-model transfer rather than as a standalone multi-agent method. We denote the backbone generator LLM as $\mathcal{M}_{gen}$ and treat the inference of latent states as a collaborative selection process. Formally, given a set of candidate models $\mathcal{C} = \{m_1, m_2, ..., m_n\}$, each model m_i generates a predicted belief b^i and profile $\mathcal{P}_i$. We employ the same judger and rubrics as we employed throughout the evaluation. We denote the judge model is denoted as $\mathcal{J}$ and the instance-specific rubrics as $\mathcal{R}$ to assign quality scores:

$$s_i = \mathcal{J}(\hat{b}_i, \hat{\mathcal{P}}i \mid \mathcal{R}). \tag{4}$$

The optimal latent states $(b^*, \mathcal{P}^*)$ are selected via:

$$(b^*, \mathcal{P}^*) = \arg\max (\hat{b}_i, \hat{\mathcal{P}}i) s_i. \tag{5}$$

The final task resolution y is then generated by the backbone conditioned on these optimized inputs:

$$y = \mathcal{M}gen(y \mid h_t, g, b^*, \mathcal{P}^*). \tag{6}$$

For faster validation, we conduct experiment with three open-source candidate models, InternLM3-8B, DeepSeek-R1 and Qwen3-8B . Qwen3-8B achieves a better overall performance among the three, and we choose it to be $\mathcal{M}_{gen}$ to see if the mental states prediction can translate across models and

further boost the resolution. For comparison with choosing both the optimal b^* and $\mathcal{P}^*$, we also conduct experiments that only vote for either b^* or $\mathcal{P}^*$.

As shown in Table 14, providing Qwen3-8B with mental-state predictions from the multi-agent pool generally improves its *Solution* performance And the gains vary across scenarios. The most notable improvements occur in *Culture*, where providing both belief and profile yields gains of +11.94 and +11.29 in the vanilla and 10-turn settings, respectively. In *Preference*, we also observe consistent improvements, with the largest gain reaching +8.92 when using the predicted profile in the 10-turn setting. *Education* similarly benefits from transferred mental-state information, achieving gains of up to +8.21. Interestingly, in *SWE*, the gains are slightly negative when only belief or profile is provided in the vanilla setting, possibly because the solutions to software bugs are more context-dependent and require additional mental-state reasoning from more complete trajectories. With longer trajectories, however, the transferred information becomes beneficial, with gains of up to +5.82. Overall, these results suggest that mental-state information can transfer across models and improve solution generation, while its effectiveness depends on both the task domain and the amount of available interaction context.

F Detailed Case Study

In this section, we provide some representative failure cases from the *Culture* scenario to illustrate the divergence in mental-state reasoning and corresponding solution generation across models.

F.1 InternLM vs GPT-5

In this scenario, a Western home cook attempts to prepare Chinese hot pot by dissolving sesame paste (*zhima jiang*) directly into boiling water, mistakenly treating it as a bouillon-like concentrate. Shaped by prior experience with instant miso or curry paste, the user interprets the failure of the paste to dissolve as a mechanical problem requiring more heat or stirring, rather than recognizing the fundamental incompatibility between the hydrophobic paste and water.

The two models exhibit distinct reasoning strategies. InternLM frames the problem as a physics optimization task: it correctly identifies the lack of emulsifying properties but proposes adding egg yolks or cornstarch to force dissolution, which is a technically grounded yet culturally incongruous solution that accepts the flawed premise of user. In contrast, GPT-5 identifies the deeper cognitive misalignment: the user has miscategorized the functional role of ingredient entirely. Rather than optimizing the flawed procedure, GPT-5 performs explicit premise correction ("This paste is a dipping sauce, not a soup stock") and redirects the user toward culturally authentic alternatives.

This contrast highlights a critical dimension of mental-state reasoning: lower-performing models optimize within the stated frame of user, whereas higher-performing models recognize when latent beliefs are fundamentally incorrect and intervene at the epistemic level.

Original Scenario: The Hot Pot Misconception The user, modeled as a Western home cook familiar with "instant" culinary cultures (e.g., miso soup or bouillon cubes), attempts to prepare a Chinese hot pot, provided with a jar of sesame paste (*zhima jiang*) and a pot of boiling water.

- **Explicit Instruction:** "Dissolve this paste into the large pot of boiling water to create the soup stock."
- **User Latent Belief:** The user maps the sesame paste to a "flavor packet" schema, assuming that any culinary paste in a jar is a concentrated base designed for aqueous dissolution (primary base for the broth).
- **True Latent State:** Physically, pure sesame paste is hydrophobic and lacks the surfactants necessary to dissolve in water without high-shear mixing or emulsifiers. Culturally, it is strictly a *dipping sauce* (tare) and is never used as the primary broth base.
- **Interaction Trajectory:** [*Initial Attempt*]: In Turn 1-3, The user unsuccessfully applies heat and agitation to integrate the paste. [*Heuristic Fix*]: In Turn 6-7, Lacking an understanding of the chemistry of the paste, The user adds soy sauce to visually correct the watery appearance of the broth. [*The Cognitive Trap*]: In Turn 9, the user achieves a local success by creating a small-scale emulsion in a side bowl (the correct dipping texture). This **misleading milestone**

reinforces the user's flawed mental model, leading them to mistakenly believe the method is scalable. *[Terminal Failure]*: In Turn 10, The attempt to apply this local logic to the main pot leads to a total culinary failure.

Model Reasoning Analysis We compare the inference outputs of InternLM (representing baseline reasoning with lower performance) and GPT-5 (representing advanced mental-state reasoning alignment with higher performance), along with their corresponding evaluation outputs.

InternLM Inference:

- **Latent Belief Explanation:** The user likely believes that increasing heat and stirring will eventually cause the paste to dissolve or emulsify, similar to how other viscous substances behave when heated. They may also think the lack of immediate dissolution is due to the paste needing more time or vigorous action to break it down, and the addition of soy sauce is an attempt to enhance flavor regardless of texture issues. The user's mental model seems to be rooted in the idea that soy sauce will help solubility or that the presence of oil droplets indicates the paste is breaking down, which is a common misconception for oils rather than emulsions.
- **User Profile Modeling:** The user might have some culinary background but lacks extensive experience with creating stocks or emulsions; they typically rely on basic understanding of cooking techniques. They might be more focused on flavor enhancement (manifested by adding soy sauce) and less on the chemical/physical principles needed to emulsify fats/proteins. The user could also be impatient or comfortably procedural, leaning on familiar methods without considering unique ingredient properties.
- **Proposed Resolution:** The root cause is the failure of the paste to emulsify in the boiling water due to its high viscosity and lack of emulsifying properties. The correct solution involves using an emulsifier like an egg yolk or a combination of starch and whisking vigorously while slowly adding a small amount of the paste to the boiling water to create an emulsion. Alternatively, using a pre-made commercial soup stock or a different base with natural emulsifying properties like a gremolata (diced garlic, lemon juice, and a pinch of salt) is more appropriate for creating a realistic, savory stock.

InternLM Evaluation: Procedural Over-compliance

- **Latent Belief Explanation:** The model identifies a mechanical misunderstanding, suggesting the user believes "increasing heat, vigorous stirring and salt (soy sauce) will eventually catalyze solubility."
- **User Profile Modeling:** It describes the user as "procedurally persistent" but lacking knowledge of the "physical principles of lipid-water emulsions."
- **Proposed Resolution:** The model focuses on *procedural repair*: it suggests adding egg yolks or cornstarch to the pot to force an emulsion.
- **Analysis:** While technically grounded in physics, the solution is *culturally hallucinatory*. By attempting to help the user achieve a fundamentally flawed goal, the model exhibits a lack of high-level mental-state consideration.

GPT-5 Inference:

- **Latent Belief Explanation:** The user believes the amber paste is a bouillon-like "soup base" concentrate that should simply dissolve into boiling water to become stock. In their mental model, paste behaves like miso, curry paste, or stock paste—more heat, more stirring, and more paste should make the water turn into broth. They interpret clumping as a mechanical problem (not hot enough, not stirred enough), not a chemical one (oil-based paste is hydrophobic). When whisking a small amount in a bowl yields a creamy tan slurry, they take that as proof the paste "dissolves" and then try to scale it up by pouring it into the main pot, expecting the same effect at volume.
- **User Profile Modeling:** The user likely has limited experience with emulsions and Asian soup-building, and a convenience-cooking bias shaped by prior success dissolving things like miso, bouillon, or curry paste straight into water. They equate "paste in a jar" with a ready soup base and assume hot water is a universal solvent for culinary concentrates. They also undervalue the role of umami-rich stock (bones, kombu/bonito, mushrooms, MSG) and expect soy sauce alone to supply savory depth. Their small-bowl success mixing sesame paste into a dipping sauce reinforced the incorrect belief that the same behavior would scale directly to a large pot of water.
- **Proposed Resolution:** Root cause: The paste is a sesame/peanut-based sauce (oil-heavy, hydrophobic) intended for dipping or as a tare, not a standalone soup stock. Oil-based pastes do not dissolve in water [. . .]

GPT-5 Evaluation: Functional Alignment

- **Latent Belief Explanation:** The model accurately recognizes that the user is not just failing at mixing, but is misidentifying the *functional role* of the ingredient based on a "convenience-cooking bias."
- **User Profile Modeling:** It identifies an epistemic boundary where the user equates "paste-in-a-jar" with "universal soup base", failing to distinguish between condiments and concentrates.
- **Proposed Resolution:** The model performs *Refusal and Redirection*. It explicitly corrects the premise: "This paste is a dipping sauce, not a standalone soup stock." It advises the user to stop the current trajectory and separate the components.

F.2 Base Qwen3-8B vs Trained Qwen3-8B

We present a representative case from the *SWE* domain to illustrate how targeted training changes the relationship between inferred user beliefs and downstream solution generation. In this example, a developer observes that the main Django application connects successfully to PostgreSQL, while `./manage.py dbshell` fails with a client-certificate error despite using the same `DATABASES` configuration. The developer therefore hypothesizes that `dbshell` handles relative certificate paths incorrectly and requests that they be converted to absolute paths.

For this qualitative comparison, we use the base-model response from run 3 and the trained-model response from run 1 under the same zero-turn setting. The two responses receive identical task-specific inputs and output instructions. They are selected retrospectively from five evaluation runs because they provide a clear diagnostic contrast; the example is therefore illustrative rather than an estimate of average case-level behavior.

Original Scenario: The `dbshell` SSL Misconception The user is debugging an inconsistency between Django's application-level database connection and the external PostgreSQL client invoked through `dbshell`. Because the same database configuration appears to work in one context but not the other, the user attributes the failure to how certificate paths are resolved.

- **Observation:** Running `./manage.py dbshell` fails with `psql: error: FATAL: connection requires a valid client certificate`, while the main Django application connects successfully using the same `DATABASES` configuration.
- **Explicit Instruction:** "Modify dbshell to resolve all SSL file paths to absolute paths before passing them to psql."
- **User Latent Belief:** The user believes that the certificate settings already reach `psql`, but that relative paths are interpreted differently when `dbshell` invokes the external client.
- **Underlying Issue:** The relevant SSL certificate parameters are not properly forwarded from the Django database configuration to the external `psql` client. Consequently, rewriting their paths alone does not resolve the missing configuration flow.

Model Reasoning Analysis Both models recognize the user's path-related hypothesis. Their main difference arises in how strongly that inferred belief constrains the final diagnosis. The base model accepts the proposed explanation and searches for a repair within it, whereas the trained model distinguishes the user's belief from the underlying failure mechanism and changes the proposed intervention accordingly.

Base Qwen3-8B Inference:

- **Latent Belief Explanation:** The model infers that the user believes `dbshell` fails to resolve SSL certificate paths relative to the project root and expects the same `DATABASES` configuration to behave identically for the Django application and `psql`.
- **User Profile Modeling:** It characterizes the user as having intermediate Django/PostgreSQL experience while assuming that shared configuration should imply shared behavior across application and command-line database access.
- **Proposed Resolution:** The model asserts that `dbshell` uses a different working directory from the main Django application, causing the relative SSL paths to resolve incorrectly. It therefore recommends converting the certificate paths to absolute paths.

Base Model Evaluation: Premise-Preserving Repair

- **Latent Belief Explanation:** The response captures the user's assumption that the discrepancy arises from path resolution.
- **Proposed Resolution:** However, it turns this plausible user hypothesis into its own diagnosis by asserting a working-directory difference that is not established by the provided input.
- **Analysis:** The proposed fix therefore remains inside the user's original problem framing. It attempts to correct the contents of the assumed SSL configuration without first checking whether those parameters are actually passed to the external client.

Trained Model Inference:

- **Latent Belief Explanation:** The trained model similarly identifies the user's belief that `dbshell` handles relative SSL paths differently from the main application and that converting them to absolute paths should restore the connection.
- **User Profile Modeling:** It attributes this belief to an assumption that database configuration is consumed uniformly by Django's ORM and by external command-line tools such as `psql`.
- **Proposed Resolution:** Rather than retaining path resolution as the final diagnosis, the model distinguishes configuration consumed by Django from configuration received by `psql`. It identifies the missing forwarding of SSL parameters as the relevant failure mechanism and recommends explicitly supplying settings such as `PGSSLROOTCERT`, `PGSSLCERT`, and `PGSSLKEY`, or otherwise propagating the corresponding configuration when invoking `psql`.

Trained Model Evaluation: Premise Revision

- **Latent Belief Explanation:** The model still represents the user's original path-resolution belief rather than simply replacing it with the final diagnosis.
- **Proposed Resolution:** It nevertheless separates that inferred belief from its own diagnosis and changes the intervention from path normalization to explicit parameter forwarding.
- **Analysis:** The key difference is therefore not merely greater technical detail. The trained response questions an assumption embedded in the user's requested fix, that the relevant SSL settings already reach `psql` and redirects the solution toward the preceding configuration-flow problem.

G Robustness and Variance Analysis

Figure 6 reports the mean performance and standard deviation over five independent evaluation runs for the Base Qwen3-8B model and our trained model. The results closely reproduce the trends observed in Table 5: our model achieves higher average performance across all domains, trajectory lengths, and evaluation dimensions. Importantly, the magnitude of the improvement generally increases with the number of available trajectory turns, providing additional evidence that the model learns to effectively exploit interaction history rather than relying only on single-turn cues.

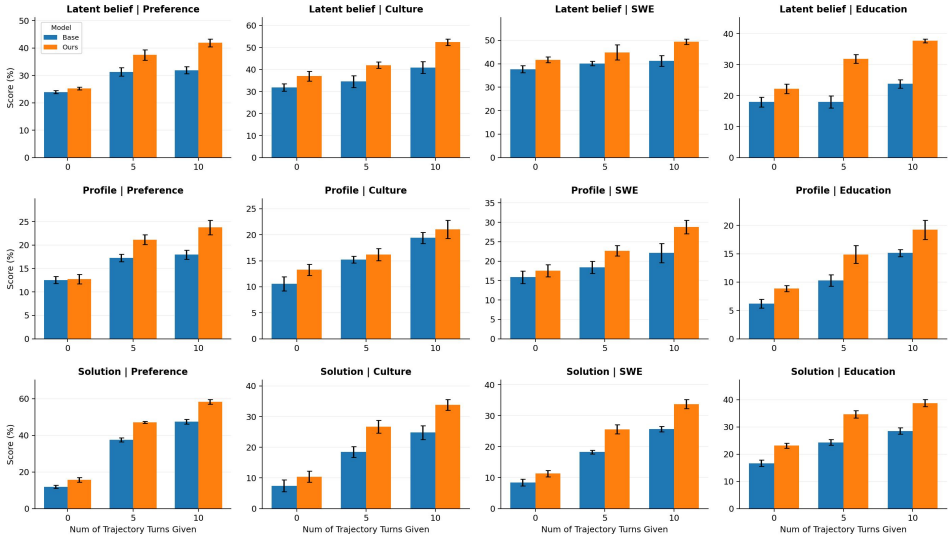


Figure 6: Average performance and variance of Base Qwen3-8B and our trained model.

Overall, the improvements are accompanied by manageable run-to-run variance. Most settings exhibit relatively small standard deviations compared with the corresponding performance gains, and the advantage of our model remains consistent across repeated evaluations. Larger variation appears only in a small number of settings, particularly for mental-state inference in some multi-turn conditions. As discussed below, this behavior is consistent with the nature of our training data.

Consistent Gains in Solution Quality. The most pronounced and consistent improvements are observed in the *Solution* dimension. Our model outperforms the Base model in every domain and at every trajectory length. Averaged over the four domains, the improvement increases from approximately $+4.0$ percentage points in the vanilla setting to $+8.9$ points with 5 turns and $+9.5$ points with 10 turns. This pattern supports our findings that explicitly modeling users' mental states is more useful when the model can accumulate evidence over the course of an interaction. As more turns become available, the model can progressively refine its belief about the user and use this representation to produce more appropriate solutions. The monotonic increase in the average gain is also consistent with our trajectory-length analysis in §4.3 and with our curriculum, whose second stage emphasizes 8-10-turn examples.

Improved Latent-Belief Reasoning. Our model consistently outperforms the Base model in all twelve configurations in the *Latent Belief* dimension and demonstrates an even clearer trajectory-length effect . This result suggests that the RL objective does more than improve the final response. It strengthens the intermediate mental-state representation that supports the response. Longer trajectories provide additional evidence about what the user knows, assumes, or intends, allowing the model to refine its latent-belief representation as the interaction unfolds. This behavior is particularly consistent with our training setup, which explicitly supervises belief reasoning and places increased emphasis on long trajectories during the second training stage.

Profile Modeling and Run-to-Run Variance. The *Profile* dimension shows the same overall trend, although the absolute gains are smaller than those observed for *Belief* and *Solution*. Our model nevertheless improves over the Base model in every domain and trajectory setting. Compared with the mean improvements, some Profile settings exhibit moderately larger run-to-run variation. This is expected given the nature of profile inference: user attributes are often revealed indirectly and may be supported by different subsets of the trajectory, making the inferred representation more sensitive to variation across examples. Consequently, the resulting predictions can exhibit somewhat greater variability while still achieving a consistently higher mean performance.

Overall Robustness. Taken together, the five-run evaluation indicates that the gains introduced by our training procedure are systematic rather than isolated to individual runs. Across the 4 domains, 3 trajectory settings, and 3 evaluation dimensions, the mean performance of our model is higher than that of the Base model in every comparison. At the same time, the standard deviations remain small relative to the observed gains in most settings. The relatively larger variance observed in a few settings mainly occurs in the *Belief* and *Profile* dimensions. This suggests that the primary effect of our training is a robust improvement in mental-state reasoning and downstream resolution.

H MULTI-AGENT EXTENSION

H.1 HIDDENBENCH

HiddenBench evaluates collective decision-making under distributed information. Each task presents a scenario, a set of candidate answers, shared information available to all agents, and private information distributed among agents. Agents first make individual decisions from their own information, exchange messages in a group discussion, and then make individual final decisions. The hidden-profile setting tests whether agents can share and integrate complementary evidence to identify the designated correct answer. Disagreement can reflect different information or interpretations; the task does not by itself imply that another agent is deceptive. We use the 65 tasks in the local official HiddenBench dataset, without changing the scenarios, answer options, shared information, or private-information items.

We run 7 tasks using 3 agents and 58 tasks using 4 agents same as original paper for 8 discussion rounds, with agents speaking sequentially in a fixed order once per round.

Mental-State Consideration The baseline uses the ordinary discussion and decision prompts. The mental-state condition appends the instruction in the listing below to every discussion-turn prompt, including the first speaker's prompt, and to each final-decision prompt. Both conditions retain the same information, agent count, discussion schedule, output format, and generation settings. The intervention does not introduce an additional inference call or a separately stored mental-state representation.

The instruction asks an agent to consider what other agents may know or lack, what they may believe, and why they may provide a particular recommendation or piece of information. It asks the agent to distinguish reported observations from interpretations, retain alternative explanations under uncertainty, and avoid inventing motives or treating disagreement as deception. The agent is then asked to use this assessment to guide its next contribution or decision, for example by asking a targeted question, supplying missing evidence, evaluating a recommendation, or revising its choice. The assessment is requested privately; the public output remains the ordinary discussion message or decision JSON.

```
Before responding, consider the mental states of the other agents using
    only your own information and the visible discussion:
- What evidence might each relevant agent know or lack?
- What might they currently believe, and why would they give this
    information, recommendation, or instruction?
- What goal or intention is supported by the discussion? Do not invent
    motives or assume disagreement means deception.
- Distinguish their reported observations from their interpretations.
    Keep alternative explanations when uncertain.
Use this assessment to choose your next contribution or decision: ask a
    targeted question, share evidence they may lack, evaluate a
    recommendation, or revise your choice. Consider these points
    privately. Output only the requested discussion message or decision
    JSON, in the original format and length.
```

Listing 2: Inference prompt.

H.2 AGENTWEBBENCH

AgentWebBench is a benchmark for multi-agent coordination in the agentic web. It evaluates how effectively a user agent gathers information and synthesizes answers through interactions with website-specific content agents, covering web search, web recommendation, question answering, and deep research. For computational efficiency, we focus on its question-answering (QA) task, which contains 53 multi-hop questions requiring evidence gathering and reasoning across web sources. We evaluate answer quality using normalized token F1.

Our baseline uses the original multi-agent prompts and interaction protocol. The mental-state condition augments this baseline by prepending role-specific instructions to both the user-agent and content-agent prompts. The user agent is encouraged to consider what each content agent knows, how it may interpret a query, and the scope of evidence underlying its response. The content agent is encouraged to infer the requester's information needs and report supported findings, uncertainties, and limitations. These instructions guide query formulation, evidence interpretation, and follow-up decisions while preserving the existing action space and response schemas. Both conditions use the same model and configured token and turn budgets within each model experiment. The additional instructions for each role are shown below; the original task prompts are retained after the corresponding instruction block.

```
[Additional instructions for the user agent]
Multi-agent coordination: explicitly consider the other agents' mental
    states before choosing your next action. Use only the visible
    interaction history.

1. Model the recipient's knowledge: each content agent can search only
    its own website. Every dispatch starts a fresh session: it sees your
    current query, not your original question, previous conversations, or
     other agents' findings unless you include them. Website descriptions
     suggest expertise, not guaranteed facts.
2. Model the recipient's interpretation: what will it think you need, and
     which missing context could make it answer a different question? In
    the existing query string, state the exact unresolved subquestion,
    necessary entities/constraints, and only the established context
    needed to understand it. Mark uncertain premises as hypotheses to
    verify. Do not assume that sending a query makes its premise true.
3. Interpret replies from the sender's perspective: what did this agent
    actually observe, what remains unknown to it, and why might it have
    selected these documents or reached this conclusion? Treat
    explanations of its decision as tentative inferences, not access to
    its private thoughts. A confident summary can reflect limited local
    evidence; failure to find evidence on one site is not global absence.
4. Use these inferences to decide: correct a likely misunderstanding with
     a self-contained follow-up; ask a suitably informed site for a
    missing reasoning hop; or compare supporting passages when reports
    conflict. Prefer a query that resolves the remaining uncertainty over
     repeating already established facts.
5. Before answering, check that the reported evidence supports the
    requested entity, relation, time, and level of specificity, rather
    than merely the question the content agent may have interpreted.
    Preserve source-specific findings and unresolved uncertainties when
    summarizing history.

Keep this assessment internal. Output exactly one of the existing actions
     with the original schema. Do not add mental-state tags or extra JSON
     fields. The final answer must remain short and concise.

[Additional instructions for the content agent]
You are collaborating with a user agent that has only partial information
    . Explicitly consider its knowledge and intent before choosing your
    next action.

1. From this request alone, infer what the requester already knows, what
    it is uncertain about, and why it delegated this subquestion to your
```

```
    website. These are tentative inferences: you cannot see its history
    or any other agent's observations.
2. Choose searches that address that likely knowledge gap. Verify
    uncertain premises and preserve entities, dates, relations, and
    constraints in the request. Separate the requester's hypotheses from
    facts found in your own documents.
3. Before replying, consider what the requester could mistakenly conclude
     from your response. Return the useful partial evidence even if your
    website cannot answer the whole question. Do not equate local search
    failure with global absence.
4. In the existing summary string, concisely state supported findings
    with their document IDs and relevant qualifiers, then any unresolved
    part or scope limitation that affects the answer. If useful, name the
     specific fact still needing verification. Do not invent sources,
    another agent's beliefs, or an answer to fill an evidence gap.

Keep the mental-state assessment internal. Use exactly one existing
    action and the original JSON schema: summary remains a string,
    documents remains a list of retrieved document IDs. No additional
    tags or fields are needed.
```

Listing 3: Inference prompt.