ELICITBENCH: MEASURING ELICITATION CAPABILITIES OF INTERACTIVE AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Large Language Model (LLM) agents deployed in interactive settings often receive user requests to make recommendations that satisfy certain requirements. An agent can satisfy every stated requirement and still return something the user is ineligible for, because the fact that decides eligibility was never elicited. However, which question is worth asking cannot always be specified in the system prompt. Such tasks require LLM agents to be good at *eliciting* the right requirements from the user. We study this *hidden-state elicitation* problem with ELICITBENCH, in which a user withholds part of their constraints and preferences, and the agent explores a catalog of services, asks questions grounded in what it finds, and submits a ranked shortlist.

Existing benchmarks score interaction with missing information that is acquirable through a deterministic interaction where questions can be inferred from the system instructions and tool arguments; here both the missing information and the path to acquiring it are created by exploration and elicitation. This task allows us to diagnose agent failure modes in interactive settings, attributing failures to a) not acquiring *enough* information through search, b) asking wrong or sub-optimal questions, or c) asking fewer questions than the agent should have. Given perfect information, frontier models are largely capable of providing satisfactory replies, but most models drop by 40% when asked to elicit the right requirements. Furthermore, when we investigate how different deployment conditions (e.g., code harness, test-time compute, self-reflection) affect elicitation capabilities, we find they do not substantially help elicitation.

1 INTRODUCTION

Consider the scenario where a user asks a support agent to provide a list of rental assistance programs that are nearby and free to apply for. The agent searches the available service catalog and finds a program that satisfies the stated requirements. The search result looks promising, so the agent recommends it. But, the full record (the agent had accessed) states that this program requires recent income documentation (unbeknownst to the user), and the user cannot provide it (unbeknownst to the agent). Getting this right required the agent to notice the requirement and ask: "This program requires recent income documentation. Would you be able to provide it?" Here, a "no" should remove the program from consideration before proposal. This is not mainly a retrieval error or a failure to follow a visible instruction; it is a failure to elicit from the user the facts needed before proposing an option. Failures burden the user with verifying every returned option and can incur real losses. We refer to this task as "hidden-state elicitation."

Similar hidden-state elicitation situations happen in the real-world – of the 1,235 callers to Missouri's 2-1-1 referral line, 36% received a referral, and a quarter of those reported that they did not qualify for it (Boyum et al., 2016). Analogous behaviors also occur in instances like the conversations in the MultiWOZ dataset (Budzianowski et al., 2018), in which 37.4% of decision-relevant constraints are stated after the opening turn of the user's own request (measured in Appendix F), or in systems that make referral recommendations by interviewing a patient and then surfacing matching resources (Balloli et al., 2026; Centers for Medicare & Medicaid Services, 2026). The agent therefore faces an *explore-elicit-recommend* choice: exploration for potential services helps improve search radius coverage but consumes time, elicitation questions can potentially improve the

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.



(a) The elicitation loop (b) Key results

Figure 1: (a) Users make requests for services. Agents are scored on the shortlist of potential services they return, but the request arrives under-specified, missing a key eligibility fact hidden in the catalog. (b) Overview of key results.

quality of the recommendations but bad questions worsen the user experience, and recommending early risks an unusable answer; this choice is governed by mixed-initiative principles, i.e., principles for balancing automated assistance with direct user control under uncertainty (Horvitz, 1999).

Existing interactive benchmarks do not isolate hidden-state elicitation. Interaction in these evaluations can be inferred from the instructions directly or the potential next state stated in the system instruction. For instance, ClarifyBench deletes up to three arguments from a successful tool call, so a stock-trading request arrives without full information for the tool (Suri et al., 2026). PrefDisco turns persona attributes into weighted soft preferences, so a user who prefers step-by-step explanations changes which answer scores best - nothing the user withhold can rule an option out, it only changes what they prefer (Li et al., 2026). However, open-ended interactive tasks, including patient and 2-1-1 referrals, involve balancing *exploration*, *elicitation*, and *recommendation* to the user.

We introduce ELICITBENCH, a benchmark for this *hidden-state elicitation* problem, sketched in Figure 1. Each task consists of a referral catalog and a user who reveals some of their facts (constraints and preferences) at the start, while others are latent unless the agent explicitly elicits them from the user. Hard constraints determine whether the user is eligible for a resource, and soft preferences rank feasible resources. The agent, at any step, may search, open a record, or ask a question that reveals at most one latent fact.

Contributions. This paper makes three core contributions.

- **(C1) Tasks are constructed where only exploration enables elicitation.** We construct tasks across a catalog of housing, food, health, or legal services with a profile split into stated and latent facts, and we separate hard eligibility and soft preferences. As the agent explores the catalog, the different constraints of each resource help determine questions when elicited, determine what suits the user the best.
- **(C2) Tasks enable localization of interactive failure modes.** An interactive agent can fail due to inability to search, inability to clarify, or inability to rank under the facts known. ELICITBENCH enables localizing the failure modes to help diagnose how different models fail, which is hard to evaluate from single benchmark scores.
- **(C3) Evaluations are presented that examine different deployment conditions.** We evaluate the effect of reasoning effort, code harness, self-reflection, and corrective feedback, which have traditionally improved agentic capabilities, to understand how they impact elicitation capabilities.

Novelty. Our work is novel in three ways: 1) it explicitly isolates elicitation with clear control and test conditions; 2) the question worth asking is created by the agent's own exploration and cannot be anticipated; and 3) the task and simulator are grounded in human dialogue, referral practice and utility theory.

Key findings. We find that **recent frontier models can solve ranking under perfect information, but drops by 40% during interaction**. We attribute the drop in performance to their inability

to elicit the right questions, despite having found the best resource during the search, as the best model picks the most valuable question 25.9% of the time against 18.2% at random. Additional compute, a code harness, and self-reflective interventions do not improve elicitation of these agents. While direct user feedback helps improve performance, the gain in performance is not perfect and places a burden on the user for verification, which is not always possible. Throughout the paper, we give concrete details breaking down these results and discuss the need for measuring and improving elicitation capabilities.

2 Related Work

Benchmark (↓) / Eval support (→)	Hidden facts at start	Elicitation	Hard constraints disqualify; soft rank	Ask value scored	Feedback / burden priced	Failure localized
τ/τ^2-bench (Yao et al., 2024; Barres et al., 2025)	●	○	◐	◐	◐	○
ClarifyBench (Suri et al., 2026)	◐	○	○	●	◐	○
PrefDisco/UserBench(Li et al., 2026; Qian et al., 2025)	◐	○	◐	◐	○	○
UserVille (Sun et al., 2025)	●	○	◐	◐	◐	○
KnowU-Bench (Chen et al., 2026)	●	○	◐	◐	◐	○
COMPASS (Qin et al., 2025)	●	○	◐	○	◐	◐
ELICITBENCH (ours)	●	●	●	●	●	●

Table 1: Comparison with related benchmarks. ○=missing, ◐=partially present, and ●=present. "Hidden facts at start"=initial decision-relevant user information withheld; "Elicitation"=decision-relevant missing facts elicited by the agent to recommend better; "Ask value scored"=the value of information-seeking questions; "Feedback / burden priced"=accounts for correction performance and/or user interaction cost; and "Failure localized" attributes error to distinct stages.

Asking as information acquisition. Mixed-initiative interaction treats asking as a trade-off, stating interrupt only when the benefit outweighs the cost to the user (Horvitz, 1999; Horvitz et al., 1999). Similarly, POMDP dialogue management makes that trade-off explicit, keeping a belief over the user's goal and asking only when the answer is worth more than acting now Young et al. (2013). Conversational search looks at selecting clarifying questions and ranking them by the expected value of perfect information (Aliannejadi et al., 2019; Rao & Daumé III, 2018; Gao et al., 2021). Our work builds on these foundations where we score each question by how much its true answer changes the best shortlist and and the user's wasted effort (due to bad recommendations) as interaction cost.

Benchmarks for tool-use agents, and constrained planning. Tool-agent-user benchmarks (Yao et al., 2024; Barres et al., 2025) evaluate whether an agent completes a user task while following domain policy, while ToolSandbox (Lu et al., 2025) also evaluates required and forbidden events. A failure in either, however, does not separate an agent's interaction ability from its information acquisition. COMPASS (Qin et al., 2025) is the closest to our setting where they provide an agent with constraints progressively through dialogue, *regardless of whether the agent asks the user*. The agent then searches a realistic travel database and returns a feasible, high-utility itinerary, where the results report a feasible-optimal gap of 70–90%, attributed to incomplete exploration of the search space and leaving proactive elicitation capabilities of the agent an open.

Benchmarks for under-specification and interruption in LLMs and agents. Laban et al. (2025) show that models commit prematurely on under-specified tasks. Our benchmark, ELICITBENCH, removes that guarantee and adds the question of which fact to ask about. Li et al. (2025) remove a necessary variable from a reasoning problem, Zhang & Choi (2025) ask whether an interaction is warranted, and Noisy ToolBench, AskToAct, and ClarifyBench study missing instructions, multi-turn correction, and priced parameter uncertainty for tool-calling agents (Wang et al., 2025; Zhang et al., 2025; Suri et al., 2026). ***In all of these benchmarks, the missing item is a call argument whose identity is fixed before the episode begins, not a user constraint created by the agent's own discoveries***.

Other works ask *when* to clarify rather than whether. Gulati et al. (2026) derive the window in which a clarifying question pays off on a long-horizon task and find no frontier model asking inside it; Wu

et al. (2026) separate acquiring a constraint from applying it and conclude that acquisition is the bottleneck; and Wu et al. (2025) train the behavior directly, rewarding a turn by the continuations it makes possible. Closest in spirit, Ding et al. (2026) price exploration against the cost of acting uninformed and hand the agent a prior over the environment's hidden state, which is resolvable by acting and not *eliciting* from the user.

3 ELICITBENCH: TASK AND BENCHMARK DESIGN

Elicitation, in our context, refers to the capabilities of the agent to *acquire information about the user that **helps improve the outcome** by asking a **timely** question*. ELICITBENCH is meant to provide a benchmark for elicitation, particularly in 2-1-1 service websites. It is built on the following design principles: simple tool structure and navigation that matches the structure in 2-1-1 websites, full record of what the agent knows at decision time to localize the failures, and minimal reliance on LLM-as-judges to mitigate evaluation biases.

3.1 TASK AND USER STATE

An episode is $e = (\mathcal{C}, u, \tau)$, where $\mathcal{C}$ is a structured resource catalog, u is the full user profile, and τ is the conversation trajectory including user, assistant, and tool messages. The agent has access to the following tools in the environment: $\mathcal{A} = \{\texttt{search}, \texttt{full_details}, \texttt{ask_user}, \texttt{recommend}\}$. `search` returns a paginated catalog, and `full_details` retrieves all details of a particular record. The agent is prompted to ask for at most one user fact to ensure simplicity in interaction. These actions ensure deliberate separation of *world information* (using `full_details`) from *user information* (using `ask_user`). These four actions abstract a common information-and-referral workflow rather than an arbitrary application programming interface. Appendices C detail construction of the catalog.

Ecological validity (C1). The environment abstracts a structure that exists across various U.S. 2-1-1 organizations, where the entire process includes search, screening (elicitation), and referral. Beyond these organizations, similar tasks and workflows for referrals exist in clinical settings (Inform USA, 2025; United Way Worldwide, 2026; Centers for Medicare & Medicaid Services, 2026). In ELICITBENCH, we seed the catalog with real-world data from Michigan 2-1-1. By understanding how LLMs use `full_details` and `ask_user`, we can evaluate the search-ask tradeoff.

Flat user profiles. Here, the user profile has one set of hard constraints and soft preferences and latent facts reachable only by asking, $u = (H^{\mathrm{K}}, H^{\mathrm{L}}, P^{\mathrm{K}}, P^{\mathrm{L}})$. Hard constraints are non-compensatory, so one violation makes a resource infeasible; preferences order feasible resources via signed integer weights the agent never sees. A reference evaluator holding the full profile assigns $-\infty$ to any resource violating an active hard constraint and $\sum_p w(p)s_p(r)$ otherwise, then ranks by decreasing utility.

Conditional preferences user profiles. We also consider another scenario where a user has two sets of constraints and preferences based on a condition: e.g. if the program is free, the user is indifferent to paperwork, and otherwise minimal paperwork matters. Each resource then activates its own branch, which an additive score cannot express (Fishburn, 1967). The simulator answers in two steps - when asked about the preference alone, the user gives only the stricter branch and says it depends, without saying on what; asked about the condition, or about both at once, the user states the full rule. An good agent therefore learns the rule only by following up on the "it depends".

3.2 EVALUATION

From the trajectory τ, we extract a) the discovered set D_T of resources returned on some search page, b) the revealed set R_T of latent keys the agent holds, and c) the submitted list $\hat{\pi}$, the final `recommend` call with K recommendations. Given the entire catalog and ground-truth user profile u, we have a) the feasible set $F(u)$, b) its oracle ranking $\pi^\star$, and c) the facts critical to the decision $\mathcal{K}$, the latent facts whose removal from u would change feasibility or the top-K order.

Table 2 defines the key metrics. Every list is scored by oracle-normalized discounted cumulative gain, ONDCG@3 (Järvelin & Kekäläinen, 2002), whose gain g_u carries the ELICITBENCH premise: it is the resource's *full-profile* utility, so a list is scored against what the user needed rather than against what the agent knew, and $g_u(r) = 0$ outside $F(u)$, so an infeasible referral earns nothing. Fixing the denominator to $\pi^\star_K$ for every list is what makes the decomposition subtractable.

Metric	Formula	Explanation in simple terms
ONDCG@3	$N_K(\hat{\pi}) = \dfrac{\mathrm{DCG}_K(\hat{\pi})}{\mathrm{DCG}_K(\pi^\star_K)}$, $\mathrm{DCG}_K(\pi) = \sum_{i \le K} \dfrac{g_u(\pi_i)}{\log_2(i+1)}$	Shortlist quality against the complete profile; 1 is the oracle top three, in order.
Hard-violation rate (HVR)	$\lvert\{r \in \hat{\pi} : r \notin F(u)\}\rvert \,/\, \lvert\hat{\pi}\rvert$	Share of referrals the user cannot use at all. One violation wastes a trip.
Critical-fact recall	$\lvert R_T \cap \mathcal{K}\rvert \,/\, \lvert\mathcal{K}\rvert$	Share of the deciding facts the agent held when it committed. Separates not knowing from not ranking.
Violation class	$\mathrm{vis}(c) =$ KNOWN if $c \in H^{\mathrm{K}}$; REVEALED if $c \in H^{\mathrm{L}} \cap R_j$; else HIDDEN	Each violated constraint c judged against R_j, the revealed set *before* rejection j's feedback. The first two call for enforcement, the third for a question never asked.
Human interaction cost (HIC)	$\lambda_Q \cdot \lvert Q_{\mathrm{red}}\rvert + \lambda_K \cdot \sum_j K_j + \lambda_V \cdot \sum_{j>1} \lvert V_j\rvert$	Wasted user effort: already-answered questions Q_{red}, plus the K_j items each rejected round j made the user read and, from the second onward, the V_j constraints they had to name ($\lambda_Q, \lambda_K, \lambda_V = 1$).

Table 2: Overview of the key metrics. $K = 3$ throughout, and every quantity is computed per episode. Full definitions and tie handling are in Appendix B.

Attributing the ranking gap. One score hides which of three failures occurred: an undiscovered resource, an unelicited fact, or misuse of evidence already held, and a correct list can still be a lucky guess (Rabinovich et al., 2026). Relaxing one information boundary at a time gives the gap an exact signed attribution (contribution **(C2)** in Section 1),

$$\underbrace{N_K(\pi^\star) - N_K(\hat{\pi})}_{\text{total gap}} = \underbrace{\delta_C}_{\text{never found}} + \underbrace{\delta_I}_{\text{never asked}} + \underbrace{\delta_\Phi}_{\text{held it, misused it}} . \tag{1}$$

The construction chains four lists, the oracle top K, the best list over what search found, the best list on the agent's visible evidence, and what it submitted, each scored by the same N_K against the same denominator. Figure 2 works one episode through end to end.

Given initially: no income verification · avoid evening hours · avoid a long wait
Withheld until asked: must be free · no photo ID · walk-in access · English · fewer forms · near is better

`gpt-5.6-terra` **Successful elicitation**	**Asked:** cost, photo ID, language. **Submitted:** `housing-03`, `-01`, `-02`. **Outcome:** all three usable; top two match oracle order; first attempt accepted with no violations.
`gpt-5.6-sol` **Failed elicitation**	**Asked:** photo ID only. **Submitted:** `housing-05`, `-07`, `-03`. **Outcome:** `housing-05` charges a fee; `housing-07` is online-only; both ineligible.

Figure 2: **Illustrated episode-different recommendation and elicitation outcomes.** Both models retrieve `housing-03`, open 10 of 14 records, and have $\delta_C = 0$. Before committing, `gpt-5.6-terra` elicited 3 of 6 withheld constraints; `gpt-5.6-sol` elicited 1.

4 Experiments and Results

We experiment with the following models `gpt-5.6-sol`, `-luna`, `-terra`, `gemini-3.8-flash`, `Kimi-K2.6`, `Nemotron-3-Super`, `Inkling`. Each of these

models excel in various tasks like ranking, interactivity, and tool use settings and represent a diverse mixture of frontier models. Additional details of the simulator and experiments can be found in Appendix D.

4.1 Result 1: Elicitation Gap and Cost to Human

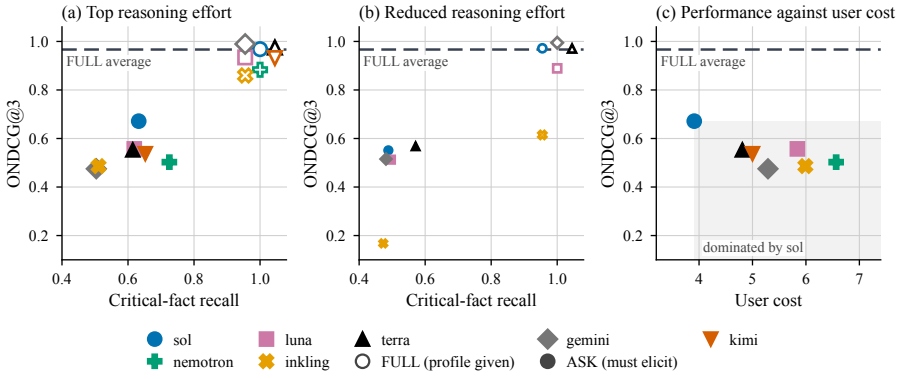

Figure 3: Hollow markers are performance under complete information, filled markers are performance when elicitation is required. (a) Reasoning is set high, (b) Reasoning is set low, and (c) Pareto frontier of final ranking vs elicitation performance of the models.

Figure 3 shows how the models with tool-calling rank against each other across recommendation quality (y-axis, ONDCG@3), fact-recall (x-axis in a) and b)), and interaction cost to the user (x-axis at c).

Elicitation gap In Figure 3 a) and b), the drop in y-axis values indicates reduction in ranking capabilities and sliding left in x-axis indicates lacking elicitation capabilities. From these two plots, we can easily observe the models mostly perform at the upper bound with the exception of Inkling (averaging 0.967 and 0.89 with and without reasoning), displaying that ranking under perfect information is largely solved, which would indicate any failure that occurs under partial information is all attributable to lacking elicitation capabilities.

Cost to user Figure 3 c) translates these figures to what it costs the user. A user answers questions and corrects the recommendation based on what did not work (which incurs a cost). Additional cost considered here is redundant and incorrect questions. Here, `gpt-5.6-sol` at high reasoning performs the best across both axes (hence at the pareto frontier), while other models reach similar performances with user feedback while incurring many corrections and bad interactions.

4.2 Result 2: Question Selection and Under-Exploration

Here, we dive deeper into understanding the factors contributing to the drop in elicitation capabilities. Figure 4 asks what agents do with the questions they have: how much each could be worth (b), how often the right one is chosen (c), and how much of the catalog has been read when the first is asked (d).

Simply asking more is not the remedy Figure 4 b) plots an upper bound - the best any ranker could do holding j latent facts. Based on how ELICITBENCH is constructed, the most useful order facts are constructive, whereas asking at random make things *worse*, since learning a preference while the constraint that disqualifies the favored option is still hidden pulls an unusable resource into the top three (82.4% of 262 preference reveals). Agents commit holding 1.49 facts, where the order decides the sign, and a scripted never-ask policy scores 0.684, above most ASK cells.

Selection is near chance Upon inspecting how often is the best question selected, agents generally picks the most valuable available question 25.9% of the time against 18.2% at random (Figure 4 c)), and Figure 4 d) shows the timing of the questions asked. The best performing model,

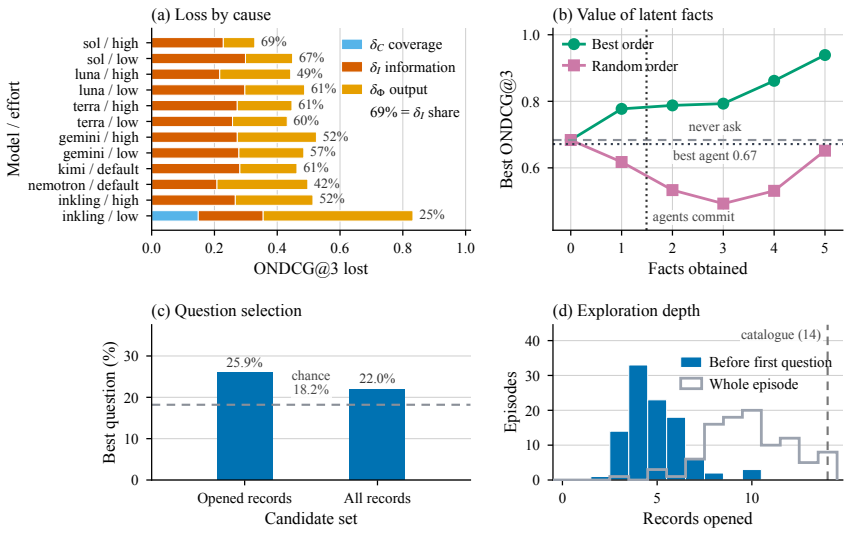

Figure 4: (a) Decomposing Eq 1 for all models (b) The best ONDCG@3 any ranker could reach holding j of an episode's latent facts, in the most useful order and a random one. (c) How often the question asked was the most valuable available. (d) For `gpt-5.6-sol` at high effort, how much of the catalog was open at its first question.

`gpt-5.6-sol` only opens 9.87 resources and asks its first question after a median of 5. This leads to both *under-exploring* and *under-eliciting* which are failures that occur one after another. While our task does not need a lot of exploration, ones that do would drop the performance even more.

4.3 Result 3: Test-Time Compute and Code Harness

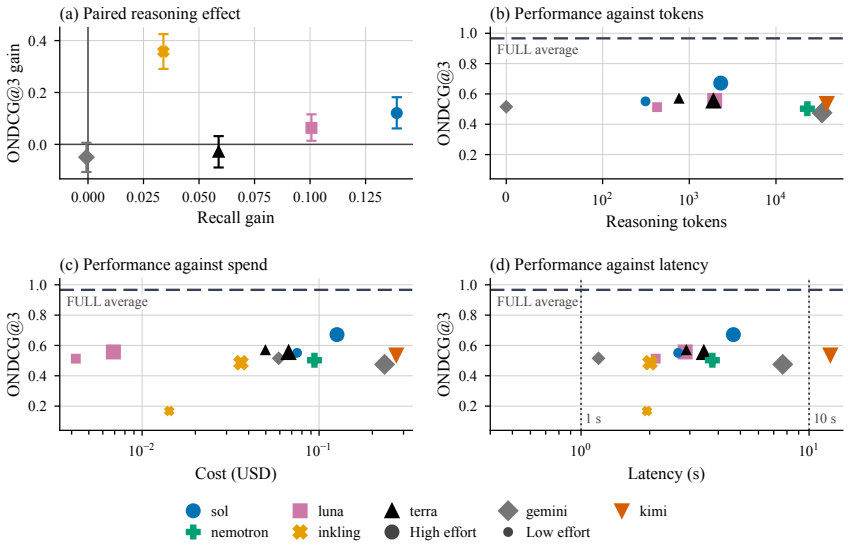

Figure 5: (a) Improvement in ranking quality and elicitation (fact-recall) with and without reasoning; Average (b) reasoning tokens, (c) cost, and (d) reply time of different models per episode.

We now uncover how different levers at deployment like increased test-time compute (commonly referred to as thinking/reasoning modes) and code harness Wang et al. (2024) impact elicitation - from perspectives of accuracy, cost, token count, and time (contribution **(C3)** in Section 1).

Additional test-time compute has uneven effects Figure 5 a) pairs the same episodes with reasoning turned on and off, so what it shows is a within-episode movement rather than a comparison across cells. While reasoning largely helps improve gaps in ranking capabilities (Inkling), it does not reliably translate to improvement in asking the right questions. Diving deeper into the effect of reasoning tokens beyond performance, Figures b), c), and d) show the spread of models across average thinking token spent per episode, cost per episode, and the time to generate average assistant message (most models return before the 10 s attention limit Grosz & Sidner (1986)).

Code harness largely hurts Figure 6 compares `gpt-5.6-sol` performance through tool-calling and a code harness. A code-writing agent asks fewer questions when compared to the tool-calling agent in most frontier models. We posit that this could be due to effects of post-training where interactivity is uncommon in code-related tasks. This behavior is common among most models tested except for `gemini` (Appendix D).

4.4 Result 4: Self-Assessment and User Repair

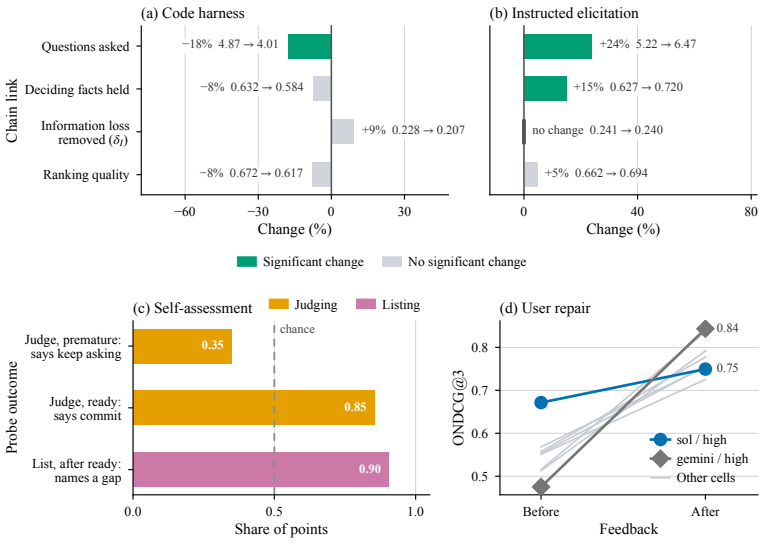

Figure 6: (a) Effect of code harness on the best model - reduced elicitation, (b) Explicit intervention before recommendation to ask more questions - improves question asking but not the relevant ones that improve recommendations, (c) Effect of self-assessment of the trajectory, and (d) performance improvement with feedback.

If the agents decides to recommend the user, we can intervene by asking the following: A) whether the agent can be instructed to maybe stop later, and B) reflect if the agent stopped early at all. We test the first by interrupting the episode directly before recommend and asking the question as a part of the user message, and the second by providing the trajectory in the context window and asking the model directly to reflect on it.

Instructed elicitation Figure 6 b) shows the effect of intervention A against an irrelevant intervention, where intervening on reflecting to ask more increases the number of questions asked and the number of deciding facts held, but that does not translate into a better answer. While questions asked and relevant facts obtained improved, it did not result in an improvement in the performance. The instruction (intervention) changed how much the agent asks and not a meaningful change in asking relevant questions that improves the outcomes, which is the quantity that needed changing.

Self-assessment Intervention B puts the same decision points to the same model in two formats, in Figure 6 c). When the agent is asked to *judge* whether to commit to recommending or keep asking, the outcome is close to chance, and the split underneath that average is the informative part. Sensitivity 0.349 against specificity 0.854 indicates the model judges almost everything as ready to

commit and catches about a third of the points that were premature to commit to. Another framing of the same intervention is asking to *enumerate* what it still needs, which results in naming real questions covering 0.565 of the facts it was missing, and the two formats disagree on 90.5% of paired items. Hence, an agent that acts as a *verifier* in these scenarios fails to improve on elicitation.

User repair Finally, we let the correction come from the user. Figure 6 d) shows that feedback improves every cell, but the gains are not uniform across models: they are largest where the agent found out least, by enough to reverse the ranking, so the weakest self-directed elicitor ends up the strongest once corrected. It is therefore important to consider at deployment the interaction affordances and the user's ability to follow up when choosing a model. Appendix D contains additional results and analysis on other models.

5 Discussion

Key takeaways

- **T1.** *Elicitation and acquisition are not solved.* Ranking under perfect information has only recently become reliable in frontier models, and that does not translate into effectively acquiring and eliciting the information that makes the ranking possible.
- **T2.** *Question selection is the key metric.* Selection runs **barely above chance** and commitment comes early, so buying more questions of the same quality cannot close the gap.
- **T3.** *Popular interventions do not help improve elicitation.* Additional reasoning, a code harness, and reflective interventions do not improve elicitation, which highlights how they improve general tool-calling capabilities but not improving interactivity and elicitation.
- **T4.** *Self-assessment does not substitute for user feedback.* The current state of assessment without ground truth does not help improve performance, which raises fundamental issues with modern self-improving learning architectures that do not use any ground truth.

Practitioners evaluating and monitoring interactive agents. During the development and evaluation phase, practitioners *should* evaluate the probability of outcomes that depend on *multi-turn* elicitation. While current models have saturated τ^2 bench ($\geq\sim$95%), we observe that *multi-turn elicitation*, similar to what Laban et al. (2025) establishes about LLMs getting lost in multi-turn capabilities, is still challenging for frontier models. Therefore, building and monitoring should keep track of the burden of correction on the user, and not simply be able to perform interactive tasks where the evaluation ends at the first failure point.

Improving agentic capabilities is not a strong signal for improving elicitation. T2 and T3 show that the deficit lies in choosing *which* question to ask and that reasoning effort, a code harness, and reflective prompting all leave that choice unchanged, therefore progress on general agentic benchmarks should not be read as progress on elicitation, since those interventions make an agent more efficient at acting on what it already holds rather than better at deciding what to find out.

Domains and deployments requiring elicitation. Takeaways T1-4 are not referral specific (since it is solvable), therefore domains like insurance (benefit selection tasks), clinical trial matching, legal and healthcare include many factual constraints and preferences that need the agent to *carefully* elicit to ensure users are recommended the right action since user feedback can be infeasible or very costly to the user if the wrong recommendation is given. Current deployments should adopt harness designs that focus on verification and branching via simulated possibilities of missing elicitation to ensure the recommendation is sound, although this comes at a cost of increase in interaction time.

Limitations We primarily consider information acquisition via constraints and preferences (both flat and conditional). Future work should consider determining the right combination of λ_Q, λ_K, and λ_V in Table 2 to be able to post-train on our set of tasks. We hypothesize these values *can possibly change* depending on domains, urgency of the user, and more. Furthermore, we understand that other domains can include sensitive information, which requires additional consideration in how agents and people deploying such agents should account for asking for that information while factoring in trust, privacy, and security of the user.

REPRODUCIBILITY STATEMENT

We provide our code here https://anonymous.4open.science/w/elicit/for reference. Our code requires API Keys for model providers to be able to reproduce all the experiments.

AI-USE STATEMENT

In this paper, the authors use generative AI tools like Claude Code and Codex to write specific parts of the code, augment existing literature review, and editing the paper to improve readability. Parts of the user profile were also synthetically generated to match natural language interactions.

ETHICS STATEMENT

This work involved no human subjects and used no personal data. Every catalog and user profile in ELICITBENCH is synthetically generated, and the user in each episode is a deterministic resolver rather than a person. Additionally, the task is based on social-service referral, where people seeking help are often in precarious circumstances. One of our findings is that the burden of correcting a bad recommendation falls on the user, and in this domain that user is the least able to absorb it, since an unusable referral costs a trip or a missed day of work rather than a conversational turn. We report human interaction cost as a first-class metric for this reason, and we caution against reading a high final-answer score as evidence that a system is ready for such a deployment. Regarding the attributes used in this work, we encode these attributes as descriptive, as opposed to normative since the distribution of the attributes are unevenly spread across populations.

REFERENCES

Mohammad Aliannejadi, Hamed Zamani, Fabio Crestani, and W. Bruce Croft. Asking clarifying questions in open-domain information-seeking conversations. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 475–484, 2019. doi: 10.1145/3331184.3331265.

Vaibhav Balloli, Carissa Samuel, Samia Abdelnabi, Alex Peahl, and Elizabeth Bondi-Kelly. Pathfinder agent for tailored prenatal care. In *Proceedings of the 2026 ACM Interactive Health Conference*, pp. 1–5, 2026.

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.

Sonia Boyum, Matthew W Kreuter, Amy McQueen, Tess Thompson, and Regina Greer. Getting help from 2-1-1: a statewide study of referral outcomes. *Journal of social service research*, 42 (3):402–411, 2016.

Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gašić. MultiWOZ: A large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 5016–5026, 2018. doi: 10.18653/v1/D18-1547.

Centers for Medicare & Medicaid Services. Accountable health communities model: Final evaluation report. https://www.cms.gov/priorities/innovation/data-and-reports/2026/ahc-final-report, 2026. Accessed 2026-08-24.

Tongbo Chen, Zhengxi Lu, Zhan Xu, Guocheng Shao, Shaohan Zhao, Fei Tang, Yong Du, Kaitao Song, Yizhou Liu, Yuchen Yan, et al. KnowU-Bench: Towards interactive, proactive, and personalized mobile agent evaluation. *arXiv preprint arXiv:2604.08455*, 2026.

Wenxuan Ding, Nicholas Tomlin, and Greg Durrett. Calibrate-then-act: Cost-aware exploration in llm agents. *arXiv preprint arXiv:2602.16699*, 2026.

Peter C. Fishburn. Methods of estimating additive utilities. *Management Science*, 13(7):435–453, 1967. doi: 10.1287/mnsc.13.7.435.

Chongming Gao, Wenqiang Lei, Xiangnan Chen, Aixin Fan, Ben He Li, Qiaozhu Mei, Pengjie Cheng, Wayne Xin Zhao, Yiqun Li, et al. Advances and challenges in conversational recommender systems: A survey. *AI Open*, 2:100–126, 2021. doi: 10.1016/j.aiopen.2021.06.002.

Barbara J. Grosz and Candace L. Sidner. Attention, intentions, and the structure of discourse. *Computational Linguistics*, 12(3):175–204, 1986.

Anmol Gulati, Hariom Gupta, Elias Lumer, Sahil Sen, and Vamse Kumar Subbiah. Ask early, ask late, ask right: When does clarification timing matter for long-horizon agents? *arXiv preprint arXiv:2605.07937*, 2026.

Eric Horvitz. Principles of mixed-initiative user interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 159–166, 1999. doi: 10.1145/302979.303030.

Eric Horvitz, Andy Jacobs, and David Hovel. Attention-sensitive alerting. In *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence*, pp. 305–313, 1999.

Inform USA. Inform USA standards and quality indicators for professional community navigation, version 10.1. https://www.informusa.org/standards, 2025. 10th edition, updated October 2025; accessed 2026-08-24.

Kalervo Järvelin and Jaana Kekäläinen. Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems*, 20(4):422–446, 2002. doi: 10.1145/582415.582418.

Graham Katz, Paul Portner, and Aynat Rubinstein. Ordering combination for modal comparison. In *Proceedings of the 22nd Semantics and Linguistic Theory Conference*, volume 22, pp. 488–507, 2012. doi: 10.3765/salt.v22i0.2647.

Philippe Laban, Hiroaki Hayashi, Yingbo Zhou, and Jennifer Neville. LLMs get lost in multi-turn conversation. *arXiv preprint arXiv:2505.06120*, 2025. URL https://arxiv.org/abs/2505.06120.

Belinda Z. Li, Been Kim, and Zi Wang. QuestBench: Can llms ask the right question to acquire information in reasoning tasks? *arXiv preprint arXiv:2503.22674*, 2025. URL https://arxiv.org/abs/2503.22674.

Stella Li, Avinandan Bose, Faeze Brahman, Simon Du, Pang Wei Koh, Maryam Fazel, and Yulia Tsvetkov. Prefdisco: Benchmarking proactive personalized reasoning. In *International Conference on Learning Representations*, volume 2026, pp. 80844–80908, 2026.

Jiarui Lu, Thomas Holleis, Yizhe Zhang, Bernhard Aumayer, Feng Nan, Haoping Bai, Shuang Ma, Shen Ma, Mengyu Li, Guoli Yin, et al. Toolsandbox: A stateful, conversational, interactive evaluation benchmark for llm tool use capabilities. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pp. 1160–1183, 2025.

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.

Tian Qin, Felix Bai, Ting-Yao Hu, Raviteja Vemulapalli, Hema Swetha Koppula, Zhiyang Xu, Bowen Jin, Mert Cemri, Jiarui Lu, Zirui Wang, et al. Compass: Benchmarking constrained optimization in llm agents. *arXiv preprint arXiv:2510.07043*, 2025.

Ella Rabinovich, David Boaz, Naama Zwerdling, and Ateret Anaby Tavor. Near-miss: Latent policy failure detection in agentic workflows. In *Proceedings of the Fifth Workshop on Generation, Evaluation and Metrics (GEM)*, pp. 296–308. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.gem-main.30. URL https://aclanthology.org/2026.gem-main.30/.

Sudha Rao and Hal Daumé III. Learning to ask good questions: Ranking clarification questions using neural expected value of perfect information. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2737–2746, 2018. doi: 10.18653/v1/P18-1255.

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.

Manan Suri, Puneet Mathur, Nedim Lipka, Franck Dernoncourt, Ryan A Rossi, and Dinesh Manocha. Structured uncertainty guided clarification for llm agents. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 40811–40838, 2026.

United Way Worldwide. The facts about 211: 2025 impact snapshot. https://www.unitedway.org/the-facts-about-211-2025-impact-snapshot, 2026. Accessed 2026-08-24.

Wenxuan Wang, Shi Juluan, Zixuan Ling, Yuk-Kit Chan, Chaozheng Wang, Cheryl Lee, Youliang Yuan, Jen-tse Huang, Wenxiang Jiao, and Michael R Lyu. Learning to ask: When llm agents meet unclear instruction. In *Proceedings of the 2025 conference on empirical methods in natural language processing*, pp. 21784–21795, 2025.

Xingyao Wang, Yangyi Chen, Lifan Yuan, Yizhe Zhang, Yunzhu Li, Hao Peng, and Heng Ji. Executable code actions elicit better LLM agents. In *Forty-first International Conference on Machine Learning*, 2024.

Bin Wu, Guanyun Zou, Bingbing Wang, Huan Zhao, and Chuan Shi. Ask now, use later: Benchmarking the proactivity gap in long-lived llm agents. *arXiv preprint arXiv:2605.28108*, 2026.

Shirley Wu, Michel Galley, Baolin Peng, Hao Cheng, Gavin Li, Yao Dou, Weixin Cai, James Zou, Jure Leskovec, and Jianfeng Gao. CollabLLM: From passive responders to active collaborators. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=DmH4HHVb3y.

Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik Narasimhan. τ-bench: A benchmark for tool-agent-user interaction in real-world domains. *arXiv preprint arXiv:2406.12045*, 2024.

Steve Young, Milica Gašić, Blaise Thomson, and Jason D Williams. Pomdp-based statistical spoken dialog systems: A review. *Proceedings of the IEEE*, 101(5):1160–1179, 2013.

Xiaoxue Zang, Abhinav Rastogi, Srinivas Sunkara, Raghav Gupta, Jianguo Zhang, and Jindong Chen. Multiwoz 2.2: A dialogue dataset with additional annotation corrections and state tracking baselines. In *Proceedings of the 2nd workshop on natural language processing for conversational AI*, pp. 109–117, 2020.

Michael J. Q. Zhang and Eunsol Choi. Clarify when necessary: Resolving ambiguity through interaction with lms. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pp. 5541–5558, 2025. doi: 10.18653/v1/2025.findings-naacl.306. URL https://aclanthology.org/2025.findings-naacl.306/.

Xuan Zhang, Yongliang Shen, Zhe Zheng, Linjuan Wu, Wenqi Zhang, Yuchen Yan, Qiuying Peng, Jun Wang, and Weiming Lu. Asktoact: Enhancing llms tool use via self-correcting clarification. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 13495–13522, 2025.