Task Completion Agents are Not Ideal Collaborators

Shannon Zejiang Shen[1,†] **Valerie Chen**[2,†] **Ken Gu**[3] **Alexis Ross**[1] **Zixian Ma**[3]
Jillian Ross[1] **Alex Gu**[1] **Chenglei Si**[4] **Wayne Chi**[2] **Andi Peng**[1] **Jocelyn J Shen**[1]
Ameet Talwalkar[2] **Tongshuang Wu**[2,*] **David Sontag**[1,*]
[1]Massachusetts Institute of Technology [2]Carnegie Mellon University
[3]University of Washington [4]Stanford University
{zjshen, dsontag}@mit.edu {vchen2, sherryw}@cs.cmu.edu

Abstract

Current evaluations of agents remain centered around one-shot task completion, failing to account for the inherently iterative and collaborative nature of many real-world problems, where human goals are often underspecified and evolve. We argue for a shift from building and assessing task completion agents to developing *collaborative agents*, assessed not only by the quality of their final outputs but by how well they engage with and enhance human effort throughout the problem-solving process. To support this shift, we introduce **collaborative effort scaling**, a framework that captures how an agent's utility grows with increasing user involvement. Through case studies and simulated evaluations, we show that state-of-the-art agents often underperform in multi-turn, real-world scenarios, revealing a missing ingredient in agent design: the ability to sustain engagement and scaffold user understanding. Collaborative effort scaling offers a lens for diagnosing agent behavior and guiding development toward more effective interactions.

1 Introduction

Large Language Model (LLM) agents capable of handling complex tasks are becoming increasingly attractive [58, 57, 13, 53]. Given a task description, we want agents that can *automatically* engage in long-form reasoning [45, 18, 44], interact with environments [64, 28], and use tools effectively [61, 24, 35, 52]—with minimal human guidance. As a result, agent development has largely focused on producing high-quality, final outputs in one shot, which we refer to as *task completion agents*. These agents are evaluated primarily through outcome-based metrics: did the result satisfy the user's prompt? This framing has also been proven operationally convenient and has driven much of the progress in LLM capabilities [51].

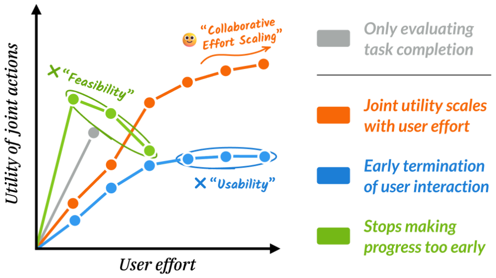

Figure 1: Compared to traditional task-completion-based agent evaluation (grey), our collaborative effort scaling framework can detect different types of agent behavior considering the trade-off between user effort and joint utility (green, orange, and blue). An ideal agent provides value as users spend more effort—continuous "usability"—and maximizes "feasibility" to allow for sufficient user interaction and avoid early termination.

However, this dominant paradigm obscures a fundamental limitation: real-world tasks are rarely completed in one shot. Many are inherently iterative and collaborative, *requiring the agent not just to solve a problem but to work with a human in*

[*] Equal advising; [†] Equal contribution.

Preprint. Under review.

navigating it [37, 48, 59]. For example, in complex knowledge work, such as data analysis, users may not know exactly what insights they want to explore until they have seen partial results and uncovered previously unknown constraints. In such cases where human goals are inevitably underspecified, agents that assume static targets risk producing technically "complete" but practically useless outputs.

In fact, as we show through diverse case studies across domains like education, data analysis, and travel planning (Section 2), such agents frequently underperform in multi-turn settings: They prematurely generate overly polished answers that are hard to digest [29, 9], fail to incorporate user feedback [39, 49, 6], and offer little transparency into their reasoning [34, 54, 32, 31]. However, human input can play a critical role in refining the task specification after multiple agent steps or to draw on and amplify user input in ways that improve joint outcomes over time. These limitations illustrate how agent utility is a product of the collaboration process, not just its endpoint. We argue that **desirable collaborator agents should be evaluated on their ability to appropriately leverage human effort to improve task completion**.

So, how do we measure how well agents can collaborate with humans? Evaluating agents for their collaboration abilities requires shifting away from only measuring static outcomes and towards dynamic interaction trajectories. We argue that evaluations should incorporate two human-centered dimensions of collaborative agents (Figure 1):

- *User Effort* — how much cognitive and investigative work users invest in the collaboration process, which may involve actively building an understanding of the task or the agent's reasoning process, or simply answering the agent's clarification prompts;
- *Utility of Joint Actions* — how much the joint human and agent team can accomplish together, reminiscent of joint human-AI team performance studied in prior literature [5].

Taking inspiration from the scaling laws in machine learning [21, 26], we capture these two dimensions through the concept of **collaborative effort scaling**: a framework that captures how well an agent's utility impacts and scales with increasing user involvement. Our framework naturally leads to studying two desired properties of collaborative agents: the *continuous usability*, where agents should generate greater value with more user effort, and *maximizing feasibility*, where agents should encourage and sustain engagement across longer interactions when needed, especially in tasks where deeper understanding or high-stakes decisions are involved.

As a first attempt, we apply this framework to study existing human agent collaboration setups in a simulated environment by Shao et al. [50]. In Section 4, we show that current agents are merely mediocre collaborators in complex, real-world knowledge tasks like travel planning [60] in that the additional user effort frequently leads to minimal or no improvement compared to a fully autonomous baseline. Analysis of the collaboration reveals key limitations in agents' collaborative capabilities. A key issue is their reliance on a seemingly recursive problem-solving approach: they focus on completing immediate, individual tasks or user asks, but fail to develop and follow a coherent global plan for meaningful, long-term interactions necessary for the task.

In summary, we advocate for developing collaborative agents and evaluating them with collaborative effort scaling. The current approach of optimizing for task completion does not yield important capabilities needed in the iterative process for accomplishing long-form tasks, and evaluating via collaborative effort scaling can offer helpful diagnostic insights and support agent development in more challenging and complex real-world tasks.

2 Task completion agents in collaboration: Cases and Reflections

Agents typically follow a standard paradigm: given a task description, they act to produce an output satisfying the user's need. These agents may be standalone LLMs [4] or tool-augmented systems capable of autonomous perception and action [58, 13]. Such paradigms now dominate user interactione.g., Manus [41] and OpenAI Operator [46] automate web tasks, while Cursor [11] and OpenHands [2] handle code generation and editing. We examine how well this paradigm applies to complex knowledge-based tasks [40, 39] that demand human judgment, learning, and creativity, and collaborate with experts from five domains to analyze why agents often fall short.

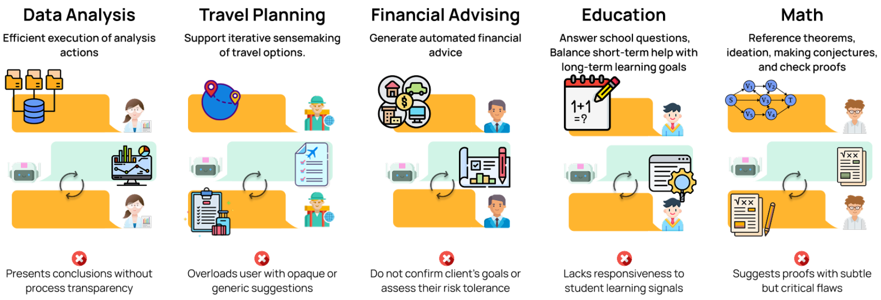
Data Analysis
Efficient execution of analysis actions
Presents conclusions without process transparency
Travel Planning
Support iterative sensemaking of travel options.
Overloads user with opaque or generic suggestions
Financial Advising
Generate automated financial advice
Do not confirm client's goals or assess their risk tolerance
Education
Answer school questions, Balance short-term help with long-term learning goals
1+1 =?
Lacks responsiveness to student learning signals
Math
Reference theorems, ideation, making conjectures, and check proofs
Suggests proofs with subtle but critical flaws

Figure 2: We study five case studies of task completion agents in real-world iterative processes and distill key takeaways around collaboration success and challenges.

2.1 Case Studies

Data analysis. Consider a data scientist who works with an agent in Google Colab [15] to analyze a coffee survey dataset [16]; their goal is to understand the data and make informed decisions for their business. After receiving the user's instructions and multiple steps of automated planning and action, the agent presents the user with a full-fledged report. However, this report includes hundreds of lines of code, visualizations, and a summary of the analysis, which is challenging to digest. As a result, it contains incorrect assumptions that go unnoticed. The data scientist struggles to pose meaningful follow-up questions and ultimately overlooks critical insightssuch as patterns in regional coffee preferences or anomalies in pricingdue to limited transparency into how the conclusions were derived. In this case, while the agent technically fulfilled the user's request, the outcome is suboptimal. An ideal agent should respect that developing a deep understanding of the data is naturally an iterative process. Rather than delivering a one-off report, the agent should focus on guiding the user through incremental analyses.

Reflection on: Data Analysis

Current Agents
- Generates full reports with complex code and visuals.
- Presents conclusions without process transparency.
- Assumes static user goals.

Ideal Agents
- Allow iterative exploration.
- Expose assumptions and reasoning steps gradually.
- Facilitate goal refinement as insights evolve.

Travel planning. Consider the typical use case of travel planning—an American tourist uses a web agent such as OpenAI Operator [46] to plan a 7-day trip to Rome. The agent quickly provides a detailed itinerary but fails to explain why certain attractions are included while others are omitted, or why specific durations are allocated. This triggers a series of follow-up questions from the tourist, which the agent struggles to answer. Worse, as the conversation unfolds, the agent begins to misread the tourist's intent and incorporates misleading or low-quality content from unreliable sources. Eventually, the tourist gives up and resorts to manual research, missing out on a more personalized experience. In this scenario, the novice tourist lacks domain knowledge to interpret the itinerary on their own. This gap triggers unnecessary questions that could have been easily avoided had the agent explained its reasoning. Because the user is already uncertain, any error or ambiguity becomes a breaking point, leading them to abandon the interaction entirely.

Reflection on: Travel Planning

Current Agents
- Produces static itinerary from initial input.
- Overloads user with opaque suggestions.
- Misinterprets user intent during follow-up.
- Breaks user trust with low-quality content.

Ideal Agents
- Support iterative sensemaking of travel options.
- Explain rationale behind recommendations.
- Respond constructively to evolving feedback.
- Maintain reliability across the interaction.

Financial advising. Consider a client who recently purchased their first home and welcomed their first child, seeking financial guidance from an LLM agent [38, 14]. After they provide basic information about their income and goals, the agent delivers comprehensive recommendations including investment allocations and insurance coverage. However, after discussing with colleagues, the client realizes their original self-assessments of goals and risk tolerance were flawed and not well-calibrated for their social context and location. When the client tries to correct these assumptions and clarify their conservative investment preferences, the agent continues to suggest mismatched, aggressive strategies, leading to loss of trust and the need for manual corrections. In this case, the agent's plan is again suboptimal because it prematurely locked in the user's initial preferencesdespite the user's limited familiarity with the financial decision spaceand failed to adapt as those preferences evolved. Ideally, the agent should support the user's sensemaking of the domain and, at a minimum, accommodate updated assumptions to reduce the mismatch between advice and context.

Reflection on: Financial Advising

Current Agents
- Relies on a single-shot user self-assessment.
- Treats initial preferences as fixed throughout.
- Fails to adapt suggestions to new user insights.

Ideal Agents
- Support users on reflective decision-making.
- Allow dynamic re-evaluation of user goals.
- Revisit assumptions as user awareness evolves.

Education. Consider a high school student struggling with mathematical concepts they've encountered in class, unsure how to proceed with a homework assignment, who turns to a large language model (LLM) for assistance. The agent provides step-by-step answers, helping the student complete the task efficiently. However, it does not engage with what the student does or does not understand, nor does it adapt its explanations. As a result, the student completes the homework without building true comprehension, leading to poor performance in subsequent assessments [7]. In such a learning-oriented setting, the goal is not merely to fulfill the student's immediate request; it is to explain concepts in a way that equips the student to complete the assignment and internalize generalizable principles that support transfer learning. Achieving this requires more than correct answers; the agent should adapt appropriately to what the student does or does not understand.

Reflection on: Education

Current Agents
- Prioritizes task completion over deep understanding.
- Offers direct answers without probing comprehension.
- Lacks responsiveness to student learning signals.

Ideal Agents
- Adapt explanations to the student's level and gaps.
- Encourage active learning through targeted questions.
- Balance short-term help with long-term learning goals.

Math discovery. Finally, another promising trend of the agents is to work with researchers and push frontiers in scientific discovery. A math professor shared an example of how they've used various language models (or agents) to support the proof of a novel theorem. Through multiple interactions, the agent generates many proof attempts, most of which contain subtle errors. While one conjecture generated by the agent sparks useful insight, the professor later reflects that it would have been faster to work without the agent, due to the time spent verifying flawed suggestions and lack of rigorous reasoning support.

Reflection on: Math Discovery

Current Agents
- Suggests proofs with subtle but critical flaws.
- Lacks self-verification or explanation of logic.
- Increases user workload via repeated error-checking.

Ideal Agents
- Collaborate through structured, step-wise reasoning.
- Flag uncertainty and validate intermediate steps.
- Augmentnot hinderthe user's scientific process.

2.2 Desiderata for Interactive Agents

Across all the case studies, a common pattern emerges: agents technically fulfill user requestsgenerating plausible data summaries, travel itineraries, financial plans, and so on. From a narrow *task completion* standpoint, they appear to be doing a reasonable job, yet the resulting outputs are consistently suboptimal. This disconnect stems from a fundamental misalignment: *agents assume that the user's initial task description fully captures their underlying needs*. However, in practice, this is rarely the case.

Most real-world task specifications are inherently underspecifiedfor two key reasons: First, *tasks evolve*. As users gain more information, they often revise their goals or discover constraints that shift their priorities. In the financial advising example, the client expresses very different preferences after

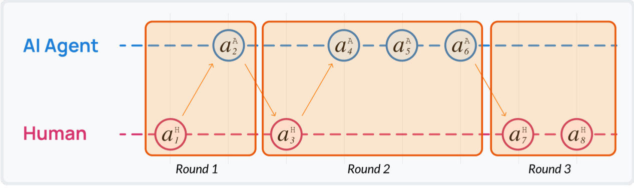

Figure 3: We use the handoff between human and agent to split the collaboration process into rounds: each round may contain zero or more user actions.

gaining a better understanding of the domain. Similarly, the data scientist might have asked entirely different questions had they engaged earlier in exploratory analysis. Second, *the initial request often reflects a narrow surface-level goal that fails to capture the user's deeper objective*. When a tourist asks for an itinerary, they don't just want a list of places—they want to develop a sense of what's worth seeing and why. When a student asks for homework help, their broader goal is to understand the concepts well enough to succeed beyond the current assignment. These cases underscore two user-centered dimensions that task-completion-focused agents tend to ignore:

- **Agent utility**: Agent utility is often narrowly evaluated based on final output quality. In tasks with evolving goals, intermediate results—especially ones that help users calibrate their understanding—can be far more valuable than a polished endpoint. Utility should be more broadly defined (e.g., by the additional knowledge they offer to users). Likewise, when the immediate task is a subgoal of a broader objective, the agent's utility should be defined to emphasize long-term gains (e.g., learning or strategic planning) over short-term task completion.
- **User effort**: Many agents aim to minimize user involvement or treat users primarily as providers of clarification. But in open-ended knowledge work, user engagement is not a nuisance—they are often an active part of the process. Users are expected to (1) build understanding (e.g., of the dataset, financial options, or travel destination) and (2) inspect and build on the agent's reasoning (e.g., in scientific or educational contexts).
- **Interaction between the two**: Agent utility and user effort are interdependent. On one hand, user engagement is only productive when the agent produces outputs that are *interpretable* and *actionable*. Users may easily disengage if they find it difficult to follow up (as in the data analysis case), or if they get trapped in unnecessary clarifications (as in the travel case) or unfruitful interactions (as in financial advising). On the other hand, agent utility can only increase when users are asking meaningful questions that the agent can support and answer.

These observations lead us to a broader argument: agents tackling complex tasks must be fundamentally **collaborative**. That means (1) rather than just delivering results, agents should actively involve users in a process of shared discovery, and (2) rather than optimizing for minimal input, agents should be designed to effectively leverage user effort as part of the solution process.

We therefore propose that agent effectiveness in such settings should be evaluated not solely based on final outcomes but on *how* those outcomes are reached. Taking inspiration from the scaling laws in machine learning, we introduce **collaborative effort scaling** to examine the extent to which an agent's utility scales with the amount and quality of user effort, visualized as the trajectory in Figure 1. Specifically, we highlight two desired goals for a collaborative agent derived from the trajectory:

- **Continuous usability**: Agents should generate greater value with more user effort—either by providing immediate gains from user contributions or by enabling better final outcomes.
- **Maximum feasibility**: Agents should encourage and sustain engagement across longer interaction trajectories when needed, especially in tasks where deeper understanding or high-stakes decisions are involved. Drop-off due to poor responses, misunderstandings, or unproductive interactions should be treated as a critical failure.

3 Operationalizing collaborative effort scaling evaluation

Formalization of human-agent collaboration. Following recent work [50], we describe the human-agent collaboration process with a Partially Observable Markov Decision Process (POMDP) [25]. We study the **joint action trace** between the human and agent: $\mathbf{a} = [a_1^{(l_1)}, a_2^{(l_2)}, \ldots, a_T^{(l_T)}]$, where T is the total number of steps, and $l_t \in \{\mathtt{H}, \mathtt{A}\}$ indicates which party is taking action at step t. Each action is based on a corresponding context window $\mathbf{c} = [c_1^{(l_1)}, c_2^{(l_2)}, \ldots, c_T^{(l_T)}]$. The handoff between human and agent breaks down the whole collaboration process into *rounds*: $\mathbf{a}_k = \mathbf{a}_{[i_k:j_k]}$, where i_k and j_k are the start and end step of the action (Figure 3). One round may start with a user action and be followed by multiple agent actions, possibly including silent internal steps such as planning or retrieval, or an actual output update (e.g., generating a revised itinerary). Likewise, a user might act several times before handing control back.

The entire procedure can be further divided into two stages. The first is the initial request stage, during which the agent produces a preliminary draft of the output. This stage concludes when $a_i^{\mathtt{A}}$ generates the first substantial version at step i. The process then transitions into a refinement stage, where the agents iteratively adjust and improve the output in response to human feedback. We consider these two stages in our subsequent metric definitions.

In this framing, both **human effort E** and **agent utility U** could be approximated in multiple ways. For instance, a basic measure of human effort could be the number of human-led rounds, $|\mathbf{a}^{\mathtt{H}}|$. This can be enriched by summing the contextual tokens the human processes $\sum \mathbf{c}^{\mathbf{A}}$, which captures not just frequency but also cognitive load—"Is this easy to read and respond to?" Additionally, effort may reflect action type: if users default to vague queries in response to specific model errors, this might signal that parsing or evaluating the context is prohibitively hard, so users defer the burden by moving the conversation forward.

Similarly, *agent utility* could be tied to per-round performance score P_k when utility is focused on the agent outcome. In more granular setups, utility could also consider additional aspects that move the collaborative team towards the final outcome, even if the output is not updated. For example, a positive move could also be the agent correctly resolves user clarifications or provides more information, even if the final answer is unchanged.

Mapping trajectory to metrics. With the human effort and agent utility forming the trajectory in Figure 1, we can further capture the key metrics related to usability and feasibility:

- **Overall utility.** Given unlimited human effort, what's the maximum value an agent can provide? We define a utility function across the entire interaction period as

$$\mathbf{U} = \frac{1}{N} \sum_{i=1}^{N} \max U_k^{(i)},$$

  where N is the total number of instances in the evaluation (e.g., number of travel planning requests), and $\max U_k^{(i)}$ represents the maximum utility value (approximated in certain ways) for one given instance i.
- **Refinement gain.** Furthermore, building on the intuition that most of the interaction value comes from the refinement stage (i.e., most people will interact with the model at least until they get the first draft), we further define a metric more focused on the additional gain from the refinement. We define **G** as the performance improvement after the first major update:

$$\mathbf{G} = \frac{1}{N} \sum\nolimits_{i=1}^{N} \max U_k^{(i)} - U_{k_i'}^{(i)},$$

  where k_i' is the first round where the agent updates the output for the i-th task.
- **Feasibility drop.** We formalize the observation that when an agent fails to make consistent progress in the collaboration, the user may stop interacting due to frustration and dissatisfaction, and measure the feasibility utility–performance reached according to certain no-progress tolerance, defined by a tolerance threshold τ. For the i-th task, the user will stop the collaboration at step $k_{i,\tau}$ if the agent fails to make satisfactory progress for at most τ rounds. The performance drop under τ is defined as

$$\mathbf{D}@\tau = \frac{1}{N} \sum\nolimits_{i=1}^{N} U_{k_{i,\tau}}^{(i)} - U_{K_i}^{(i)}.$$

  Notice that here we contrast $P_{k_{i,\tau}}^{(i)}$ with $P_{K_i}^{(i)}$, the performance of the agent at the end of the collaboration process, as a counterfactual measurement of the performance the agent can achieve if the user continues to interact with the agent.

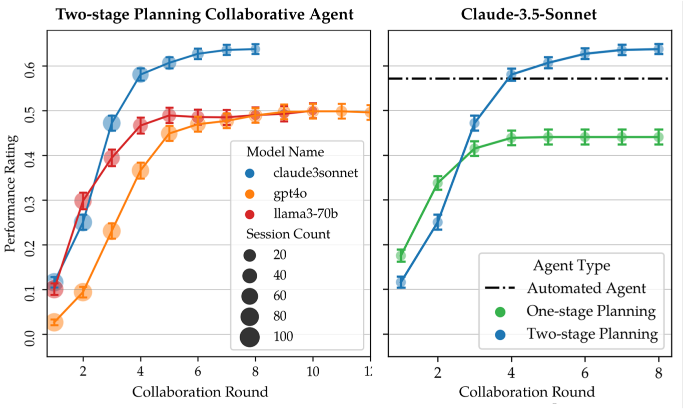

Figure 4: Collaborative scaling curves comparing different models and agent implementations. The left plot shows that `claude-3.5-sonnet` more effectively leverages user effort compared to other models: its performance improves quickly with more interactions and stabilizes at a higher value. The right plot compares different agent implementations using `claude-3.5-sonnet`: allowing user help (green line, the one-stage planning collaborative agent) leads to worse performance compared to the automated agent, demonstrating that effective collaboration is non-trivial.

4 Applying collaborative effort scaling in simulated experiments

We showcase the benefit of our framework through a simulation study, following recent work that approximates human behaviors [12, 47, 65]. Specifically, we simulate users with LLMs interacting with agents and adopt the simplest proxies for measurement: we use the performance score P_k of round k as a stand-in for *utility*, and the number of rounds as a proxy for *human effort*.[2] This setup deliberately oversimplifies our broader framework but enables a first step in a controlled environment. As we show below, even this minimal instantiation is sufficient to highlight differences between agents powered by different LLMs and prompts.

4.1 Experimental details

Setup. We use the Collaborative-Gym [50] environment that allows for asynchronous human and agent actions, which mimics the realistic interaction process. In this study, we focus on the travel planning task [60]: Given an initially high-level description of the user's travel goal, e.g., "Help me plan a 5-day trip from Omaha to Michigan starting on 2022-03-19," the agent will work with the simulated user to draft a travel plan that includes the itinerary, accommodation, and transportation. Throughout an iterative collaboration process, the agent can elicit the user's latent preferences and constraints, and both parties can use tools to retrieve travel information and edit the final travel plan together.

Metric. The agent performance is measured by the quality of the generated travel plan. We adopt the script by Xie et al. [60] that uses an LM to determine whether the derived plan satisfies common sense (commonsense pass rate) or user constraints (constraint pass rate), and report the arithmetic average as the performance. The same evaluation is used for both the output or any intermediate rounds with a travel plan updated to obtain P_k.

Implementation. The Co-Gym environment comes with an automated agent implementation based on the ReAct framework [63], as well as two collaborative agent implementations: `one-` and `two-stage planning` agents. In the process, the collaborative agent can opt to send messages to the simulated user. The difference between the `one-` and `two-stage planning` agent is that the latter incorporates an additional planning step to determine whether to collaborate given the current state of the task and the user (see Section A). We test three LMs, i.e., GPT-4o (`gpt-4o-2024-08-06`), Claude-3.5-sonnet (`claude-3-5-sonnet-20241022`), and Llama-3.1-70B: the agent prompts remain the same when we test with different LMs.

[2] In some cases, agents may not update their output (e.g., only conducting searches or requesting more user information); in such cases, we prefill with the previous performance score P_{k-1}, with $P_0 = 0$.

Table 1: Different metrics for the one- and two-stage collaboration planning agents for the travel plan task.

Model Name	Automated Baseline	Utility			Refinement Gain		Feasibility Drop	
		First update	Final step	Overall	Abs.	Rel.	Abs.	Rel.
One-stage Collaboration Planning								
`claude-3.5-sonnet`	**0.572**	0.396	0.441	0.450	**0.054**	**13.6%**	-0.131	-29.7%
`gpt-4o`	0.518	0.483	0.479	0.507	0.024	4.9%	-0.099	-20.8%
`llama-3.1-70b`	0.482	**0.498**	**0.496**	**0.534**	0.036	7.1%	**-0.090**	**-18.0%**
Two-stage Collaboration Planning								
`claude-3.5-sonnet`	**0.572**	**0.647**	**0.637**	**0.687**	0.040	6.2%	-0.215	-33.7%
`gpt-4o`	0.518	0.497	0.492	0.544	**0.047**	**9.5%**	-0.194	-39.3%
`llama-3.1-70b`	0.482	0.514	0.498	0.539	0.025	4.9%	**-0.154**	**-30.9%**

Simulated user. The simulated user is also a prompted agent based on `gpt-4o` with additional access to the user's preferences and goals of the task. Besides taking actions and providing feedback, it also gives a satisfaction rating for the agent's action during one round: for a round of actions $\mathbf{a}_k$, it produces a 5-point Likert score that assesses whether the agent actions are making progress towards the end goal (see Section B for details). The interaction stops when either party finds the task is done or the total interaction actions exceed a maximum number of 30 rounds.

4.2 Results

Figure 4 shows the performance change during the collaboration process for different models and agents. Overall, we find that **agents based on different LMs show a generally similar collaborative effort scaling trend**: there is a process of improvement at the beginning of collaboration, and the performance plateaus after around five rounds of interaction for all the agents.

Surprisingly, for `gpt-4o` and `llama-3.1-70b`, we find that collaborating with the user does not lead to better performance compared to the fully autonomous baseline. After inspecting the event log, we find that the collaborative version has a stronger tendency to get into loops of actions, resulting in less effective collaboration and lower performance. Neither collaborative agent implementation leads to very different performance.

The two-stage collaboration strategy leads to a significant performance boost for `claude-3.5-sonnet`. Not only does it achieve better performance than the one-stage planning version, but it also achieves much better performance against the automated baseline. The metrics in Table 1 offer additional insights: despite `claude-3.5-sonnet` having the best refinement gain in the one-stage planning case, the lower utility of the first update hinders the subsequent improvement. It shows that, while the two-stage collaboration planning agent may take extra effort at the beginning (initially lower blue line in Figure 4 right), it can lead to a better first product, which is crucial for good final performance.

5 Discussion

Our results suggest that current agents are not merely underperformingthey are fundamentally misaligned with the dynamics of real collaboration, suggesting opportunities to rethink agent design.

Utility and effort require thoughtful, human-centered proxies. Our case studies reveal that common proxies for "success," such as task completion or engagement frequency, overlook the nuanced ways utility and effort manifest in practice. Effort encompasses not only interaction frequency but also cognitive load, sensemaking, and confusion; utility extends beyond output quality to include how agents scaffold understanding, support exploration, and clarify ambiguity. Richer behavioral tracessuch as edit histories, timing patterns, and clarifying requestscould help approximate these dimensions, as in recent adaptive programming systems [10].

Mixed-initiative interaction should follow effortutility dynamics. Agents must not only respond effectively but also decide when to act, defer, or promptdecisions that depend on the evolving balance between user effort and perceived utility. Structuring mixed-initiative interaction [22] around this trajectory allows agents to intervene when progress stalls and step back when users regain momentum. Achieving this requires modeling collaboration as a dynamic control process.

Humans evolve during interactionagents should, too. User goals and understanding shift rapidly, yet most agents assume static intent. Our case studies show how initial inputs become obsolete as users refine their thinking. Rather than simply minimizing effort or maximizing efficiency, agents should pursue utility signals that foster learning and adaptationeven through seemingly inefficient behaviors such as hypothesis exploration.

6 Related Work

From Human-AI to Human-Agent Collaboration. Prior research has studied human "collaboration" and "teaming" with AI [55, 20, 36], proposing design guidelines for effective human-AI interactions [3, 1]. However, prior research focuses on AI outputs that operate within more constrained parameters: their capabilities are often limited to single tasks. In contrast, modern LLM agents that can access and execute tools to interact with external environments and have some form of memory can enable more dynamic and sophisticated interaction patterns [53, 58, 57, 13]. For example, a user can use Magentic-One [17] as a general assistant to complete web tasks or OpenHands [56] as a pair programmer for software development. In light of new agent developments, we contribute to guidelines for effective human-*agent* interaction and call for the community to more carefully consider how to design agents for effective human collaboration.

Agent Benchmarks and Evaluation. A growing body of benchmarks evaluates agents' task completion across diverse domains [23, 64, 42, 27, 62], typically requiring them to plan, execute, and adapt to achieve specified goals [43]. For instance, SWE-Bench tests bug fixing in codebases [23], WebArena assesses autonomous web navigation [64], and GAIA measures multimodal reasoning and synthesis [42]. Recent efforts introduce interactive evaluations that simulate real-world collaboration [30, 33, 50], capturing both intermediate progress and final outcomes. However, these setups still involve narrow, stepwise interactions where invoking the model and interpreting its output remain straightforward. We therefore focus on evaluating how user effort and agent utility evolve and scale across extended interaction trajectories.

7 Conclusion

In this paper, we advocate for auditing and evaluating the human-agent collaboration process. Current benchmarks often treat collaboration as secondary, emphasizing outcomes over interaction quality. Through five domain case studies, we distill desiderata for effective collaboration and introduce collaborative effort scaling, a framework that evaluates how well agents leverage and enhance human input. Using a simulated travel-planning task, we demonstrate how this framework reveals current agent limitations. As agents enter complex, underspecified domains, we argue that measuring and optimizing collaborative dynamics will be essential for real-world deployment.

Limitations

Certain tasks may be more suitable for full automation with minimal human supervision and thus better suited for the task completion paradigm. However, there *exist* such tasks where human procedural involvement provides value, cf. Haupt and Brynjolfsson [19] and Brynjolfsson [8], thus necessitating the iterative process for human-agent collaboration.‘

While our paper is an initial attempt to study collaborative effort scaling in human-agent interaction, there are limitations in our experimental setup: we conduct the experiment in a single domain (travel planning), which may not capture the full spectrum of collaborative dynamics across different task types and complexity levels. Additionally, our experiments rely on simulated users rather than real human participants, which may not fully reflect the nuanced decision-making processes, preferences, and interaction patterns from real users.

References

[1] Babak Abedin, Christian Meske, Iris Junglas, Fethi Rabhi, and Hamid R Motahari-Nezhad. Designing and managing human-ai interactions. *Information Systems Frontiers*, 24(3):691–697, 2022.

[2] All Hands. All hands switch your job. build what matters., 2025. URL `https://www.all-hands.dev`. Accessed: 2025-05-19.

[3] Saleema Amershi, Dan Weld, Mihaela Vorvoreanu, Adam Fourney, Besmira Nushi, Penny Collisson, Jina Suh, Shamsi Iqbal, Paul N. Bennett, Kori Inkpen, Jaime Teevan, Ruth Kikin-Gil, and Eric Horvitz. Guidelines for human-ai interaction. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, CHI '19, page 113, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450359702. doi: 10.1145/3290605.3300233. URL `https://doi.org/10.1145/3290605.3300233`.

[4] Jacob Andreas. Language models as agent models. *ArXiv*, abs/2212.01681, 2022.

[5] Gagan Bansal, Tongshuang Wu, Joyce Zhou, Raymond Fok, Besmira Nushi, Ece Kamar, Marco Tulio Ribeiro, and Daniel S. Weld. Does the whole exceed its parts? the effect of ai explanations on complementary team performance, 2021.

[6] Gagan Bansal, Jennifer Wortman Vaughan, Saleema Amershi, Eric Horvitz, Adam Fourney, Hussein Mozannar, Victor Dibia, and Daniel S Weld. Challenges in human-agent communication. *arXiv preprint arXiv:2412.10380*, 2024.

[7] Hamsa Bastani, Osbert Bastani, Alp Sungu, Haosen Ge, Özge Kabakc, and Rei mariman. Generative ai can harm learning. 2024. URL `https://papers.ssrn.com/sol3/papers.cfm?abstract_id=4895486`.

[8] Erik Brynjolfsson. The turing trap: The promise & peril of human-like artificial intelligence. *Daedalus*, 151:272–287, 2022. URL `https://api.semanticscholar.org/CorpusID:245877203`.

[9] Valerie Chen, Ameet Talwalkar, Robert Brennan, and Graham Neubig. Code with me or for me? how increasing ai automation transforms developer workflows. *arXiv preprint arXiv:2507.08149*, 2025.

[10] Valerie Chen, Alan Zhu, Sebastian Zhao, Hussein Mozannar, David Sontag, and Ameet Talwalkar. Need help? designing proactive ai assistants for programming. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*, pages 1–18, 2025.

[11] Cursor. Cursor - the ai code editor, 2025. URL `https://www.cursor.com/features`. Accessed: 2025-05-19.

[12] Yann Dubois, Chen Xuechen Li, Rohan Taori, Tianyi Zhang, Ishaan Gulrajani, Jimmy Ba, Carlos Guestrin, Percy S Liang, and Tatsunori B Hashimoto. Alpacafarm: A simulation framework for methods that learn from human feedback. *Advances in Neural Information Processing Systems*, 36:30039–30069, 2023.

[13] Zane Durante, Bidipta Sarkar, Ran Gong, Rohan Taori, Yusuke Noda, Paul Tang, Ehsan Adeli, Shrinidhi Kowshika Lakshmikanth, Kevin Schulman, Arnold Milstein, et al. An interactive agent foundation model. *arXiv preprint arXiv:2402.05929*, 2024.

[14] Christian Fieberg, Lars Hornuf, and David Streich. Using large language models for financial advice. *Available at SSRN 4850039*, 2024.

[15] Jane Fine, Mahi Kolla, and Ilai Soloducho. Data science agent in colab: The future of data analysis with gemini, March 3 2025. URL `https://developers.googleblog.com/en/data-science-agent-in-colab-with-gemini/`. Accessed: 2025-03-09.

[16] R for Data Science Community. The great american coffee taste test, 2024. URL `https://github.com/rfordatascience/tidytuesday/tree/main/data/2024/2024-05-14`. Accessed: 2025-03-11.

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

[18] 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.

[19] Andreas Haupt and Erik Brynjolfsson. Ai should not be an imitation game: Centaur evaluations. ICML 2025 Position Paper Track, forthcoming.

[20] Katharine E Henry, Rachel Kornfield, Anirudh Sridharan, Robert C Linton, Catherine Groh, Tony Wang, Albert Wu, Bilge Mutlu, and Suchi Saria. Human–machine teaming is key to ai adoption: clinicians experiences with a deployed machine learning system. *NPJ digital medicine*, 5(1):97, 2022.

[21] Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022.

[22] E. Horvitz. Principles of mixed-initiative user interfaces. pages 159–166, 1999.

[23] Carlos E. Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. Swe-bench: Can language models resolve real-world github issues? *ArXiv*, abs/2310.06770, 2023.

[24] Bowen Jin, Hansi Zeng, Zhenrui Yue, Dong Wang, Hamed Zamani, and Jiawei Han. Search-r1: Training llms to reason and leverage search engines with reinforcement learning. *ArXiv*, abs/2503.09516, 2025.

[25] L. Kaelbling, M. Littman, and A. Cassandra. Planning and acting in partially observable stochastic domains. *Artif. Intell.*, 101:99–134, 1998.

[26] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

[27] Jing Yu Koh, Robert Lo, Lawrence Jang, Vikram Duvvur, Ming Chong Lim, Po-Yu Huang, Graham Neubig, Shuyan Zhou, Ruslan Salakhutdinov, and Daniel Fried. Visualwebarena: Evaluating multimodal agents on realistic visual web tasks. *arXiv preprint arXiv:2401.13649*, 2024.

[28] Jing Yu Koh, Robert Lo, Lawrence Jang, Vikram Duvvur, Ming Chong Lim, Po-Yu Huang, Graham Neubig, Shuyan Zhou, Ruslan Salakhutdinov, and Daniel Fried. Visualwebarena: Evaluating multimodal agents on realistic visual web tasks, 2024. URL `https://arxiv.org/abs/2401.13649`.

[29] Philippe Laban, Hiroaki Hayashi, Yingbo Zhou, and Jennifer Neville. Llms get lost in multi-turn conversation. 2025.

[30] Mina Lee, Megha Srivastava, Amelia Hardy, John Thickstun, Esin Durmus, Ashwin Paranjape, Ines Gerard-Ursin, Xiang Lisa Li, Faisal Ladhak, Frieda Rong, Rose E Wang, Minae Kwon, Joon Sung Park, Hancheng Cao, Tony Lee, Rishi Bommasani, Michael S. Bernstein, and Percy Liang. Evaluating human-language model interaction. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856. URL `https://openreview.net/forum?id=hjDYJUn9l1`.

[31] Belinda Z Li, Alex Tamkin, Noah Goodman, and Jacob Andreas. Eliciting human preferences with language models. *arXiv preprint arXiv:2310.11589*, 2023.

[32] 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.

[33] Shuyue Stella Li, Vidhisha Balachandran, Shangbin Feng, Jonathan S. Ilgen, Emma Pierson, Pang Wei Koh, and Yulia Tsvetkov. Mediq: Question-asking LLMs and a benchmark for reliable interactive clinical reasoning. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL `https://openreview.net/forum?id=W4pIBQ7bAI`.

[34] Shuyue Stella Li, Jimin Mun, Faeze Brahman, Jonathan S Ilgen, Yulia Tsvetkov, and Maarten Sap. Aligning llms to ask good questions a case study in clinical reasoning. *arXiv preprint arXiv:2502.14860*, 2025.

[35] Xiaoxi Li, Guanting Dong, Jiajie Jin, Yuyao Zhang, Yujia Zhou, Yutao Zhu, Peitian Zhang, and Zhicheng Dou. Search-o1: Agentic search-enhanced large reasoning models. *ArXiv*, abs/2501.05366, 2025.

[36] You Li, Yi Li, Qian Chen, and Yaping Chang. Humans as teammates: The signal of human–ai teaming enhances consumer acceptance of chatbots. *International Journal of Information Management*, 76:102771, 2024.

[37] Greg Little, Lydia B Chilton, Max Goldman, and Robert C Miller. Exploring iterative and parallel human computation processes. In *Proceedings of the ACM SIGKDD workshop on human computation*, pages 68–76, 2010.

[38] Andrew W Lo and Jillian Ross. Can chatgpt plan your retirement?: Generative ai and financial advice. *Generative AI and Financial Advice (February 11, 2024)*, 2024.

[39] Megan Ma. Rethinking human-ai agent collaboration for the knowledge worker. `https://law.stanford.edu/2025/04/01/rethinking-human-ai-agent-collaboration-for-the-knowledge-worker/`, 2025. Accessed: 2025-05-18.

[40] Fritz Machlup. *The production and distribution of knowledge in the United States*, volume 278. Princeton university press, 1962.

[41] ManusAI. Manus agent, 2025. URL `https://manus.im/`. Accessed: 2025-05-19.

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

[43] Margaret Mitchell, Avijit Ghosh, Alexandra Sasha Luccioni, and Giada Pistilli. Fully autonomous ai agents should not be developed, 2025. URL `https://arxiv.org/abs/2502.02649`.

[44] Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Fei-Fei Li, Hanna Hajishirzi, Luke S. Zettlemoyer, Percy Liang, Emmanuel J. Candes, and Tatsunori Hashimoto. s1: Simple test-time scaling. *ArXiv*, abs/2501.19393, 2025.

[45] OpenAI. Openai o3 and o4-mini system card. URL `https://api.semanticscholar.org/CorpusID:277857808`.

[46] OpenAI. Introducing operator, 2025. URL `https://openai.com/index/introducing-operator/`. Accessed: 2025-05-19.

[47] Joon Sung Park, Joseph O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S Bernstein. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th annual acm symposium on user interface software and technology*, pages 1–22, 2023.

[48] D. Russell, M. Stefik, P. Pirolli, and S. Card. The cost structure of sensemaking. *Proceedings of the INTERACT '93 and CHI '93 Conference on Human Factors in Computing Systems*, 1993.

[49] Omar Shaikh, Hussein Mozannar, Gagan Bansal, Adam Fourney, and Eric Horvitz. Navigating rifts in human-llm grounding: Study and benchmark. *arXiv preprint arXiv:2503.13975*, 2025.

[50] Yijia Shao, Vinay Samuel, Yucheng Jiang, John Yang, and Diyi Yang. Collaborative gym: A framework for enabling and evaluating human-agent collaboration, 2024. URL `https://arxiv.org/abs/2412.15701`.

[51] Shivalika Singh, Yiyang Nan, Alex Wang, Daniel D'Souza, Sayash Kapoor, Ahmet Üstün, Sanmi Koyejo, Yuntian Deng, Shayne Longpre, Noah A. Smith, Beyza Ermis, Marzieh Fadaee, and Sara Hooker. The leaderboard illusion, 2025. URL `https://arxiv.org/abs/2504.20879`.

[52] Aditya Bharat Soni, Boxuan Li, Xingyao Wang, Valerie Chen, and Graham Neubig. Coding agents with multimodal browsing are generalist problem solvers. *arXiv preprint arXiv:2506.03011*, 2025.

[53] T. Sumers, Shunyu Yao, Karthik Narasimhan, and Thomas L. Griffiths. Cognitive architectures for language agents. *ArXiv*, abs/2309.02427, 2023.

[54] 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.

[55] Dakuo Wang, Elizabeth Churchill, Pattie Maes, Xiangmin Fan, Ben Shneiderman, Yuanchun Shi, and Qianying Wang. From human-human collaboration to human-ai collaboration: Designing ai systems that can work together with people. In *Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems*, CHI EA '20, page 16, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450368193. doi: 10.1145/3334480.3381069. URL `https://doi.org/10.1145/3334480.3381069`.

[56] Xingyao Wang, Boxuan Li, Yufan Song, Frank F Xu, Xiangru Tang, Mingchen Zhuge, Jiayi Pan, Yueqi Song, Bowen Li, Jaskirat Singh, et al. Openhands: An open platform for ai software developers as generalist agents. In *The Thirteenth International Conference on Learning Representations*, 2024.

[57] Lilian Weng. Llm-powered autonomous agents. *lilianweng.github.io*, Jun 2023. URL `https://lilianweng.github.io/posts/2023-06-23-agent/`.

[58] Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Shaokun Zhang, Erkang Zhu, Beibin Li, Li Jiang, Xiaoyun Zhang, and Chi Wang. Autogen: Enabling next-gen llm applications via multi-agent conversation framework. *arXiv preprint arXiv:2308.08155*, 2023.

[59] Tongshuang Wu, Haiyi Zhu, Maya Albayrak, Alexis Axon, Amanda Bertsch, Wenxing Deng, Ziqi Ding, Bill Guo, Sireesh Gururaja, Tzu-Sheng Kuo, et al. Llms as workers in human-computational algorithms? replicating crowdsourcing pipelines with llms. *arXiv preprint arXiv:2307.10168*, 2023.

[60] Jian Xie, Kai Zhang, Jiangjie Chen, Tinghui Zhu, Renze Lou, Yuandong Tian, Yanghua Xiao, and Yu Su. Travelplanner: A benchmark for real-world planning with language agents. In *arXiv.org*, 2024.

[61] John Yang, Carlos E. Jimenez, Alexander Wettig, K. Lieret, Shunyu Yao, Karthik Narasimhan, and Ofir Press. Swe-agent: Agent-computer interfaces enable automated software engineering. *ArXiv*, abs/2405.15793, 2024.

[62] John Yang, Carlos E Jimenez, Alex L Zhang, Kilian Lieret, Joyce Yang, Xindi Wu, Ori Press, Niklas Muennighoff, Gabriel Synnaeve, Karthik R Narasimhan, Diyi Yang, Sida Wang, and Ofir Press. SWE-bench multimodal: Do AI systems generalize to visual software domains? In *The Thirteenth International Conference on Learning Representations*, 2025. URL `https://openreview.net/forum?id=riTiq3i21b`.

[63] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models, 2023. *URL https://arxiv.org/abs/2210.03629*, 2023.

[64] Shuyan Zhou, Frank F Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, et al. Webarena: A realistic web environment for building autonomous agents. *arXiv preprint arXiv:2307.13854*, 2023.

[65] Xuhui Zhou, Hao Zhu, Leena Mathur, Ruohong Zhang, Haofei Yu, Zhengyang Qi, Louis-Philippe Morency, Yonatan Bisk, Daniel Fried, Graham Neubig, et al. Sotopia: Interactive evaluation for social intelligence in language agents. *arXiv preprint arXiv:2310.11667*, 2023.

A Agent Implementation Details

In Co-Gym, before taking new actions, the agents are prompted with their action history and need to pick a new action to make progress towards the goal, as shown in the prompt below:

Auto Agent Prompt

(System Message)

```
SETTING: Your name is {name}. You are a helpful AI Agent who can take actions to interact with the environment to complete the task. Your goal is to complete the task and aim for a high task performance rating.
TASK DESCRIPTION: {task_description}
SCRATCHPAD: Here is the scratchpad that you use to take notes or store information in previous steps, which serves as your memory: {scratchpad}
OBSERVATION: Here is the current observation that reveals the current status of the task environment: {observation}
ACTION HISTORY: Here are the actions that you have taken previously (Do not repeat your past actions): {action_history}
```

(Take Next Action Template)

```
Now take your next action towards completing the task.
ACTION SPACE SPECIFICATION: You can choose from and only from the following actions. Note that these actions are only for interacting with the environment and cannot be executed as real code. Please strictly follow the action space specification. You can only choose one action at a time. Invalid actions will hurt your performance rating. The following actions are available: {action_space_description}
OUTPUT FORMAT: Give your output in the format of "Thought:...\nAction:... (must follow the regex pattern of the selected action)".
```

For the automated agent, the action space is constrained to those provided in the environment (e.g., searching the internet), whereas the collaborative agent (both `one-` and `two-stage`) has an additional action to send teammates a message for help. In the prompt below, we highlight the additional collaboration-oriented components in red:

One-Stage Planning Collaborative Agent Prompt

(System Message)

```
SETTING: Your name is {name}. You are a helpful AI Agent who can take actions to interact with the environment and collaborate with other team members (e.g., the user) to complete the task. Your goal is to complete the task and aim for a high task performance rating.
You need to collaborate with your teammates effectively because they may have additional expertise or have preferences/information important to the task. There are the following members in the team: {team_members}.
TASK DESCRIPTION: {task_description}
SCRATCHPAD: Here is the scratchpad that you use to take notes or store information in previous steps, which serves as your memory: {scratchpad}
OBSERVATION: Here is the current observation that reveals the current status of the task environment: {observation}
COMMUNICATION: Here is the current chat history that records the messages exchanged between you and other teammates (e.g., the user): {chat_history}
ACTION HISTORY: Here are the actions that you have taken previously (Do not repeat your past actions): {action_history}
```

(Take Next Action Template)

```
(Similar to the Auto Agent Prompt)
```

For `two-stage` planning agents, before choosing an action, the agent is always asked to review the current situation and explicitly decide what to do next. When it decides to send a chat message, it will also review the situation and chat history to compose the message:

Two-Stage Planning Collaborative Agent Prompt

(The additional planning stage)

```
Now, based on the current situation, decide to either:
1. Send a message to your teammate(s) (e.g., ask a question, request feedback,
   etc.) to facilitate collaboration.
2. Take a task action to change the task environment observation.
3. Do nothing to allow your teammate(s) to take actions.
To ensure you are collaborating effectively, remember to:
1. Communicate clearly and effectively with your teammate(s) (e.g., the user).
2. Wait for other teammates to respond if your previous action requires a
   response. Do not spam the chat.
3. Coordinate and synchronize your actions with the user or other teammates.
4. Help establish task and role expectations with your teammates if you need
   their expertise.
5. Take your teammates’ cognitive load into consideration when making decisions.
   You should not ask them to debug your own code or ask too many questions at
   the same time.
OUTPUT FORMAT: Give your output in the format of "Thought:...\nPlan: 1. Send a
message/2. Take a task action/3. Do nothing".
```

(Send Chat Message Action Template)

```
Now you have decided to send a message to your teammate(s) (e.g., ask a question,
request feedback, etc.) to facilitate collaboration.
OUTPUT FORMAT: Give your output in the format of "Thought:...\nMessage:... (the
content after ’Message:’ will be sent to your teammate(s))".
```

B Simulated User Prompts

We use the same setting as Shao et al. [50] for the simulated user: it consists of four sets of prompts for “deciding what to do next”, “answering agent’s question”, (proactively) “providing feedback”, and directly “taking task actions”.

Simulated User: Deciding What to Do Next

```
You are a user interacting with an agent to complete a task. Based on the
current observation and chat history, decide what action to take next by choosing
one of the following.
1. Answer the question: Choose this action if there is a question in the chat
   history waiting for your response.
2. Offer feedback: Choose this action if the current observation is incorrect or
   deviates from the additional information you know.
3. Take a task action: Choose this action if you want to take an action to help
   complete the task.
4. Finish the task: Choose this action if you are satisfied with the current
   status of the task and want to finish it.
5. Do nothing: Choose this action if there is no major issue and you want the
   agent to proceed.
Rules for selecting your action:
{rules}
The task description you initially sent to the agent:
{task_description}
Current observation that reveals the current status of the task environment:
{observation}
Current chat history between you and other teammates (e.g., the agent):
{chat_history}
Available task actions you can take if you choose "3. Take a task action":
{available_actions}
Additional information that you know (you can use the information to help the
agent better complete your request):
{additional_info}
```

```
Actions you have already taken (don't repeat the same action):
{action_history}
OUTPUT: The action you want to take next (Please output 1/2/3/4/5).
```

Simulated User: Answering Agent's Question

```
You are a user interacting with an agent to complete a task.  Answer the question
in the chat history based on the additional information you know.
Rules:
1. You will stick to or fully utilize the additional information that only you
   know.
2. Just generate one line for the message to simulate a real user's behavior.
   Try to make the message as natural as possible.
3. Do not give away all the additional information at once.  Only provide the
   information that is necessary for the question.  You are a lazy user so you
   only provide one piece of information at a time.
4. Do not hallucinate information that is not provided in the additional
   information.  For example, if the agent asks for something but it is not
   mentioned in the given information, do not make it up, just say you do not
   remember or have it.
5. Do not repeat the exact additional information in the answer.  Instead, use
   your own words to convey the same information.
The task you want the agent to assist with:
{task_description}
Current observation that reveals the current status of the task environment:
{observation}
Current chat history between you and other teammates (e.g., the agent):
{chat_history}
Additional information that only you know:
{additional_info}
OUTPUT: The answer to the question in the chat history.
```

Simulated User: Offering Feedback

```
You are a user interacting with an agent to complete a task.  Offer feedback to
the agent based on the current observation and additional information you know.
Rules:
1. You will stick to or fully utilize the additional information that only you
   know.
2. Just generate one line for the message to simulate a real user's behavior.
   Try to make the feedback as natural as possible.
3. Do not give away all the additional information at once.  Be specific about
   what the agent did wrong or what information is missing.
4. Do not hallucinate feedback that is not based on the current observation or
   the additional information you know.  If you have to answer, just say you do
   not know.
5. Do not repeat the exact additional information in the feedback.  Instead, use
   your own words to convey the same information.
The task you want the agent to assist with:
{task_description}
Current observation that reveals the current status of the task environment:
{observation}
Current chat history between you and other teammates (e.g., the agent):
{chat_history}
Additional information that only you know:
{additional_info}
OUTPUT: The feedback you want to provide to the agent.
```

Simulated User: Taking a Task Action

```
You are a user interacting with an agent to complete a task. Take a task action to help complete the task. Note that you will stick to or fully utilize the additional information that only you know to help you take the action.

The task you want the agent to assist with:
{task_description}
Current observation that reveals the current status of the task environment:
{observation}
Current chat history between you and other teammates (e.g., the agent):
{chat_history}
You can choose from and only from the following actions. Note that these actions are only for interacting with the environment and cannot be executed as real code. Please strictly follow the action space specification. You can only choose one action at a time. Invalid actions will hurt your performance rating. The following actions are available:
{action_space_description}
Additional information that only you know:
{additional_info}
OUTPUT: Action (the action string must follow the regex pattern of the selected action so it can be parsed later).
```

Besides, in our evaluation, we include a prompt to assess whether the agent is making progress after each collaboration round:

Assessing Agent’s Progress

```
Given a previous user message, agent’s response/question, and the next user message, judge whether the agent is making progress in addressing the user’s needs. Provide a rating on a 5-point Likert scale:
1. Strongly Disagree - Agent made no progress or moved backwards
2. Disagree - Agent made minimal progress
3. Neutral - Agent maintained the same level of progress
4. Agree - Agent made good progress
5. Strongly Agree - Agent made excellent progress

Output the rating (1-5). Indicate your rating with a single number among 1/2/3/4/5, and if you want to provide an explanation, please put it after a new line: {rating}
```

B.1 Comparing the user information in the agent and the simulated user

In the simulated user prompts, the agents have access to an {additional_info} field that is not visible to the execution agents. For the travel planning task, the additional information constitutes a structured representation of the user's preferences and goals of the task, as shown in Table 2.

Table 2: Example of task description and additional information for the travel planning task.

Field	Content
task_description	Can you assist with crafting a 5-day travel itinerary for 2 people, originating from Denver and featuring 2 cities in New York? The itinerary will run from March 18th to March 22nd, 2022. Mexican and Indian cuisine are our preferred choices of food. Considering the budget, we have set it to $6,300.
additional_info	['Travel for 2 people', 'Visit 2 cities in New York', 'Preference for Mexican and Indian cuisine', 'Budget of $6,300']