Cocoa: Co-Planning and Co-Execution with AI Agents

K. J. Kevin Feng*
University of Washington
Seattle, WA, USA
kjfeng@uw.edu

Kevin Pu*
University of Toronto
Toronto, ON, Canada
jpu@dgp.toronto.edu

Matt Latzke
Allen Institute for AI
Seattle, WA, USA
mattl@allenai.org

Tal August
UIUC
Urbana, IL, USA
taugust@illinois.edu

Pao Siangliulue
Allen Institute for AI
Seattle, WA, USA
paos@allenai.org

Jonathan Bragg
Allen Institute for AI
Seattle, WA, USA
jbragg@allenai.org

Daniel S. Weld
Allen Institute for AI
Seattle, WA, USA
danw@allenai.org

Amy X. Zhang
University of Washington
Seattle, WA, USA
axz@cs.uw.edu

Joseph Chee Chang
Allen Institute for AI
Seattle, WA, USA
josephc@allenai.org

Figure 1: Cocoa is an interactive system that facilitates co-planning and co-execution with AI agents in a document environment for scientific researchers. Cocoa integrates AI agents into documents using a novel interaction design pattern—*interactive plans*—through which a human user and an AI agent can jointly plan and execute plan steps using a shared representation of tasks, roles, and progress directly in the document.

ABSTRACT

Human collaboration benefits from continuous coordination— planning, delegating tasks, sharing progress, and adjusting objectives—to align on shared goals. However, agentic AI systems often limit users to previewing or reviewing an agent's plans for fully autonomous execution. While this may be useful for *confirmation* and *correction*, it does not support deeper *collaboration* between humans and AI agents. We present Cocoa, a system that introduces a novel design pattern—*interactive plans*—for collaborating with an AI agent on complex, multi-step tasks. Informed by a formative study (n = 9), Cocoa builds on interaction designs from computational notebooks and document editors to support flexible delegation of agency through *Co-planning* and *Co-execution*, where users collaboratively compose and execute plans with an *Agent*. Using scientific research as a sample domain, our lab (n = 16) and field deployment (n = 7) studies found that Cocoa improved agent steerability without sacrificing ease-of-use compared to a strong chat baseline. Additionally, researchers valued Cocoa for real-world projects and saw the interleaving of co-planning and co-execution as an effective novel paradigm for human-AI collaboration.

1 INTRODUCTION

Since the advent of personal computing, researchers and practitioners in artificial intelligence (AI) and human-computer interaction (HCI) have set sights on developing intelligent AI agents that can help perform everyday computer-based tasks on our behalf [35, 63, 69, 70]. Recent advancements in large language models (LLMs) and agent frameworks involving reasoning, planning, memory, and tool use [53, 93, 101, 104, 114] have accelerated the development of fully autonomous AI agents that can conduct complex tasks with varying degrees of success. These tasks include shopping online [112], writing software [42, 108, 111], performing "deep research" to assemble a report based on web sources [27, 79, 86], and general computer-based tasks [1, 80]. By contrast, turn-based and direct manipulation LLM systems that shift the burden of task planning and progress monitoring to users—such as chat or node-based interfaces [65, 87, 100, 109]—have become popular in HCI in recent years for offering users greater agency and control.

*Project completed during an internship at Semantic Scholar Research, Ai2.

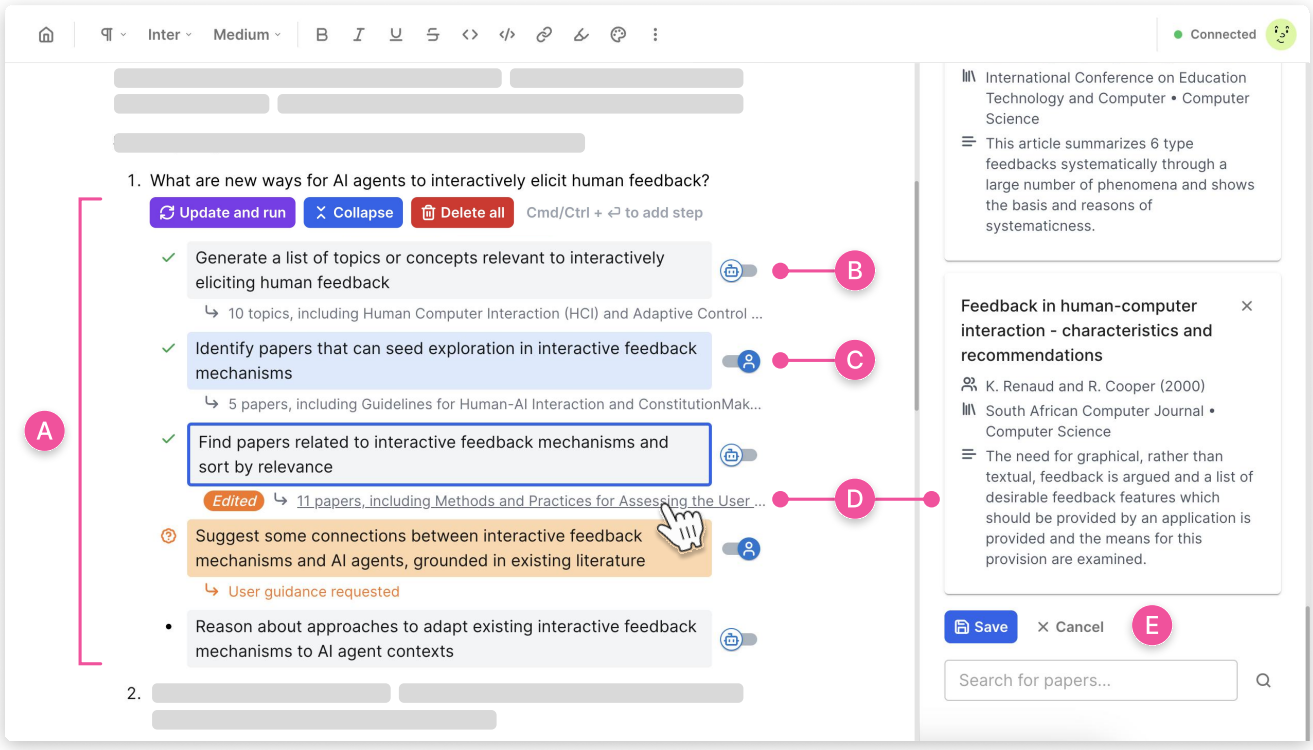

Figure 2: An overview of the Cocoa user interface. An interactive plan (A) affords human-AI co-planning and co-execution: a researcher and the AI agent can collaboratively edit the plan in the document and execute the plan steps, similar to executing code cells in a computational notebook. Steps can be assigned to the AI agent (B) or the researcher (C). The researcher can freely edit the AI agent's outputs in an interactive sidebar (D), such as adding relevant papers that the agent did not find (E) to help steer the agent with their feedback and expertise. In this example, the first three steps of a plan to summarize methods for human feedback elicitation have already been executed, and the agent is requesting guidance from the user in the next step.

While some systems have attempted to balance AI and human agency by allowing users to review an agent's reasoning chains *after execution* for verification [32, 77], or to edit and confirm its plans *before execution* [27, 80], few works have explored how to deeply involve and collaborate with users *during* iterative planning and execution. We argue that effective human interaction with agents remains essential for agents to operate effectively and safely in the real world [49, 96], such that human-AI teams can accomplish tasks that cannot easily be completed by humans or AI alone [8]. Realizing this vision demand deeper explorations into strategies for fostering synergistic collaboration between human users and AI agents [20, 39, 49].

More concretely, this lack of human-AI collaboration in task planning and execution can lead to several issues. First, we cannot steer the agent with our expertise and worldly understanding. In domains where agents can not yet produce a quality plan and the ability to steer agents towards one, agents can easily veer off track, wasting resources without achieving meaningful results [36, 103]. When executing the plan, human guidance can significantly boost agent performance—Shi et al. [96] found that even simple feedback from programmers to an agent for solving Olympiad-style programming problems increased agent accuracy from 0% to over 85%. Second, and perhaps more importantly, removing human agency in favor of AI agency can pose heightened safety risks, disempower us to think critically and be creative, and harm our well-being more generally [10, 13, 20, 56]. More fundamentally, for many nuanced, subjective, and high-stakes tasks, human input must be considered for AI agents to be successful and aligned with users' personal needs and goals [18, 23, 102].

In this work, we introduce Cocoa, a novel system for Co-planning and Co-execution with AI Agents. We chose scientific research tasks as a sample domain due to the complex, multi-step, and information-dense workflows researchers often engage in throughout their work [25, 26, 46, 47, 61]. Cocoa embeds AI agents into a document editor—a common environment many researchers use to keep track of tasks and progress for their research projects. Taking inspiration from computational notebook interfaces where data scientists compose program instructions and manage their execution and outputs, we introduce a new interaction design pattern for AI agents called *interactive plans*. Interactive plans orchestrate actions between a human user and an AI agent and enable flexible delegation of human and

AI agency. As a result, Cocoa supports human-AI **co-planning**: the agent, when invoked in the document, will first propose a plan of action that is interactive and seamlessly integrates into the document. The user can then edit the plan steps through familiar interactions that mirror typical document editing and assign steps to the agent or themselves. Cocoa also supports human-AI **co-execution**: drawing inspiration from the design of computational notebooks, the interactive plan allows the user and the agent to collaboratively complete one step at a time or re-execute steps as desired. The user can interactively refine the agent's intermediate outputs and also manually take over steps themselves to steer the workflow in a more effective direction. Most importantly, Cocoa **interleaves co-planning and co-execution**, such that users can smoothly transition between the two and modify their plans based on outputs from execution, and vice versa.

Concretely, our work makes the following contributions:

(1) A formative study with 9 researchers with a focus around their real-world project documents that uncovered needs and opportunities to better support planning and execution.
(2) Cocoa, an interactive system that implements a new design pattern—interactive plans—in a document editor for researchers to engage in co-planning and co-execution with an AI agent.
(3) A user evaluation of Cocoa with 16 researchers, where we found that interactive plans with novel interaction affordances enabled users to better steer the AI agent without sacrificing ease of use when compared to a conversational chat baseline.
(4) A 7-day deployment study of Cocoa with 7 researchers that offers insights into how interactive plans can provide value to researchers in their day-to-day work.

2 RELATED WORK

2.1 Planning and Interactivity in LLM Agents

Prior work has identified core components in LLM agent architectures to consist of memory, reasoning, planning, and tool use [53, 66, 101, 104, 114, 118]. Central to LLM agents' longitudinal operations is long-term planning, which demands capable reasoning abilities [33] and thus is often implemented by chain-of-thought (CoT)[1] [78, 88, 107, 114]. CoT provides a series of intermediate reasoning steps as exemplars in prompting to boost performance on complex reasoning tasks [78, 107]. CoT is crucial for facilitating task decomposition and therefore LLM agents' planning capabilities [17, 88]. However, even with CoT, critical investigations of LLMs' abilities to autonomously generate executable plans reveal limited success across diverse domains [103]. Indeed, although CoT can help improve model planning for tasks with well-defined, objective solutions that provide clear signals for reinforcement learning—e.g., numerical, tabular, and knowledge-based reasoning [53]—it may not be so effective in domains that require expert tacit knowledge to navigate ambiguous problem spaces with no single right answer [33]. Unlike model properties that show empirical improvement through scaling laws, limitations of planning in these domains may not resolve with scale alone as 1) tacit knowledge is not well-documented in training data and is thus difficult for a model to robustly learn [18, 22, 115], and 2) there may be no "correct" workflow for CoT to follow and verify the correctness post-hoc [43]. Scientific research is one such domain [115].

In light of this, recent work has recognized the need to interactively incorporate user feedback for improving LLM agents' planning capabilities and beyond [50, 58, 80, 95, 103, 114], particularly within scientific discovery [28, 71]. For example, Lawley and MacLellan [58] architected an approach where user interaction is used to guide the model in planning for unseen tasks on-the-fly using a hierarchical network of smaller actions. OpenAI's Operator [80] allows users to "take control" of the agent to perform tasks manually. Yet, interactive techniques in this area are still nascent, despite the new challenges identified for human-agent communication [7]. Interfaces for LLM assistants (e.g., AgentGPT [90], Devin [108]) have been primarily limited to chat interfaces targeted at *monitoring* agent activity rather than *empowering the user to proactively collaborate* with the agent. An emerging body of work has started to experiment with more interactive techniques for agents. Kazemitabaar et al. [50] developed interfaces for data analysts to edit an execution plan of an LLM to provide more control points for steering behavior, while Google's Deep Research [27] can re-generate a plan if the user provides feedback via prompting. However, these interactions are *corrective* rather than *collaborative*—that is, users would typically engage in these interactions to correct agent behavior post-hoc rather than *proactively iterate back-and-forth* with the agent at multiple points in the workflow. We target the latter in our work to elevate human feedback to beyond simply a corrective mechanism. That is, our goal is to enable *interactive* agents that effectively elicit and incorporate key guidance from the user through co-created human-AI representations.

2.2 Computational Notebooks

The computational notebook is an interactive paradigm that organizes program imperatives, input data, and rich outputs (e.g., tables, visualizations, interactive widgets) into linearly arranged cells [16, 57], reflecting Donald Knuth's early visions of literate programming [55]. Computational notebooks are a well-studied paradigm in HCI (e.g., [4, 16, 19, 64, 75, 117]), with most prior work focusing on how data scientists leverage them in their workflows.

In our work, we take inspiration from design decisions made for computational notebooks because of the many parallels between workflows in notebooks and mixed-initiative task completion systems [29, 35, 110]. For instance, data scientists often break down a high-level data analysis task into executable cells in computational notebooks [51, 91]. As they execute each cell and inspect outputs, they can verify the quality and sensibility of outputs before moving forward in their analysis, or go back to modify existing code cells before execution to interleave between programming, execution, and output examination [52].

By drawing these parallels, we can reveal new opportunities to design interactive workflows and interfaces for LLM agents. Just like how a data scientist may easily add new cells in a notebook or reconfigure existing cells to adapt their analysis plan, a user may interactively edit an agent's plan of execution to better steer

[1]We use CoT as an umbrella term encompassing chain-of-thought, trees-of-thought [113, 119], and related methods.

the agent in productive directions. Furthermore, an agentic system may simultaneously be more usable and resource-efficient if a user could direct an agent to iterate on a subtask to improve its output before continuing onto subtasks dependent on that output. This also demands new ways of viewing and editing agent outputs on subtasks, which increases opportunities for human input over existing approaches such as simply logging agent actions [3, 81, 90]. Our work exploits these parallels to contribute new design patterns for steering agentic systems.

2.3 AI for Scientific Research

Significant efforts in recent years have advanced how AI can be used to support scientific research. Of particular interest to us are works that leverage these recent advances to help researchers with planning and execution of *literature-focused tasks*. Planning a research project is a cognitively demanding, complex process consisting of iterative cycles of divergent and convergent thinking grounded in literature review [15, 21, 84], making it an ideal scenario to explore human-AI collaboration.

One major thread of prior research has focused on leveraging AI models as tools or components in user-driven systems. These include tools for paper reading [15, 89] and skimming [26], literature review [61, 84], paper recommendation [45, 46], information retrieval and sensemaking [14, 25, 48, 106], ideation [6, 31, 40] and "deep research" systems that combine support for multiple activities [27, 79, 86]. Progress in this area has given rise to excitement for a "computational inflection for scientific discovery" [34]. On the other end of the spectrum are fully automated end-to-end systems where an AI agent has much more agency in planning and executing tasks to attempt to carry out entire research projects on their own [28, 41, 67].

As AI systems become more capable, we ask this question: Is the generation of full research artifacts (e.g., research questions, proposals) the most desired and promising paradigm to assist researchers? Indeed, when working with other (human) collaborators, researchers often find richness in co-evolving partially completed artifacts [43, 59, 98, 99] and answering or asking questions that stimulate critical thinking and reflection [82, 83]. Further, some fully automatically generated artifacts are still found to be of lower quality than those generated with human involvement [41]. A limited number of works have thus explored *human-AI co-creation* of research artifacts [5, 46, 65, 74]. These include in-document commands that trigger an AI assistant to complete a partial citation based on the user's preferred bibliographies and paper collections [5], interactive node-based editors to iteratively expand upon and refine AI-generated research questions [65, 87], and emulating colleague and mentor personas using LLMs to work with the user in developing research proposals [74].

However, despite these systems, there is little clarity on *how researchers want to co-create with AI* during research planning and execution. Prior work (e.g., [65, 74]) assumed that generating particular artifacts is helpful to researchers. Rather than focusing on generating predetermined research artifacts, we contribute an interaction design pattern that allows for flexible specification of the final artifact and collaboration between a researcher and an AI agent.

3 FORMATIVE STUDY

Building on prior work, we aim to enhance collaboration between humans and AI agents using a shared, interactive operational representation. We focus on scientific research—a challenging domain for planning given its complex, multi-step workflows and the nuanced decisions researchers make upon processing information-dense scientific papers. Specifically, we investigate the potential for project documents—continuous records of ideas, updates, and tasks—to act as the agent environment. We thus conducted a formative study to answer the following research questions:

- **F1:** What are the properties and opportunities of researchers' project documents for human-agent collaboration?
- **F2:** How do researchers engage in planning within project documents?
- **F3:** How would researchers prefer to collaborate with an AI agent in their research workflows?

3.1 Participants, Procedure, and Data Analysis

We recruited 9 Ph.D. students (detailed demographics in Table 1 of Appendix B) through an interest form sent to a research organization's internal Slack channel and the authors' professional connections. We targeted Ph.D. students as they often lead the detailed planning and execution of research projects and are the primary editors of project documents.

All studies were conducted virtually over Google Meet and lasted 60 minutes. Before the study, we collected from participants an active or past research project document and a brief description of their research interests. The study was divided into 3 parts:

(1) An activity involving the participant's project document to better understand their current planning behavior (25 minutes).
(2) An activity with a Wizard-of-Oz (WoZ) design probe to explore how researchers plan and explore research ideas in a document editor with LLM support via a chatbot (25 minutes).
(3) An exit interview where researchers reflected on their experience with the probe and using LLMs in research (10 minutes).

Details for each part can be found in Appendix C. Each participant was given a $35 USD honorarium after the study. The study was reviewed and exempted by our organization's internal review board. We qualitatively analyzed project documents produced from the study probe activity alongside study transcripts. Our data analysis methods can be found in Appendix D.

3.2 Findings

In this section, we redact any project-specific details for privacy and intellectual confidentiality.

3.2.1 ***[F1 & F3] Project documents are promising environments for human-agent interaction.*** Participants' project documents acted as a "hub" for their projects. These documents commonly included meeting notes, *planned to-do items*, *progress updates* tied to to-do items, *research questions* to discuss with collaborators, and *links to other documents* detailing a particular project component in greater depth. These documents often served as "scratch

paper" for researchers to informally leave a trace of their reasoning and high-level goals, conduct short-term and long-term planning, and progress tracking of these plans. Most documents (P1, P2–4, P7–P9) contained one or more problem statements that motivated the project and stated its core contributions. From there, participants listed subgoals and ideas for planned exploration (all participants).

Participants expressed a desire for tighter integration between an AI agent and their project documents in their research workflows for two main reasons. **First**, they wanted to receive in-situ AI support as they plan and execute research tasks in their documents. P1 and P2 both envisioned an AI system *"actively engaging with the content I'm writing [...] after I write each statement, the system could retrieve relevant papers that could provide background or related work for me to read more."* P9 appreciated *"the ability to lay out the steps when I start something. [It] was very helpful [for] visualizing the outcome."* **Second**, participants wanted the AI agent to have access to the broader context that already existed in their project documents. When using the probe, P2 was concerned that they would need to *"prompt [the chatbot] with a lot of background,"* but realized they *"might have [the background] written down"* in their document. P3 desired closer document integration but warned that the agent may clutter their document. They suggested: *"having something on the side that does not flow into where I am writing, like a side [panel]—if I want something from it, I want to bring it back into my doc."*

3.2.2 ***[F2] Literature search and understanding play a central role in research planning.*** In all project documents, we saw places where participants intended to initiate a multi-step plan (often via a *how?* and *can we?* question) to address different research tasks. The most common category of tasks are ones that were literature-augmented (e.g., literature search and understanding), which are often combined with a wide variety of other research actions, such as experiment/artifact design (P3), experiment/code execution (P4), data inspection and synthesis (P7), communicating and discussing results (P5), and ideation (P2). Literature was not only a significant component of *existing* plans, but was important for informing *future* plans. Some participants' plans (e.g., P3, P4, P7) were formed from hypotheses they hoped to verify, which prior work may have already addressed; P7 pointed out that the hypotheses they had in mind can *"actually be proven from previous papers"* so they saw literature review as a planning aid. Consequently, all participants expressed a desire to use AI to help with *literature-augmented tasks*—exploring and understanding relevant literature to inform decision-making. This finding echoes recent surveys on how researchers leverage LLMs to conduct research, where the most frequent usage category was information-seeking [62].

3.2.3 ***[F3] Participants preferred to perform higher-level reasoning and synthesis themselves.*** In addition to highlighting tasks where they wanted AI assistance, participants also spoke about tasks they did not want AI to automate away. In particular, participants saw higher-level reasoning and information synthesis as critical tasks they wanted to do themselves. They were also often unsatisfied with AI's outputs on these tasks. P3 explains: *"this tinkering process around reading and playing around with things is what gives you the ideas. I don't know if I want those things automated because the process is as helpful as the final result."* P5 agreed that they *"wouldn't want it to be making the final decisions for sure. Just give me inspiration for where I can go."* Specifically, they shared that *"the main thing I worry about is feeling enough ownership"* if AI makes more consequential decisions. An overeager AI assistant that attempts to perform tasks researchers prefer to do themselves is frustrating because it does not complement the user's work and instead creates undesired noise for them to filter through.

3.3 Design Goals

We synthesize our formative study findings into three design goals for an interactive system that facilitates meaningful collaboration between researchers and AI agents by using plans as a shared representation.

DG1: Integrate seamlessly into a document environment. Project documents contain rich externalizations of researchers' workflow, thought processes, and plans (e.g., "to-dos") [F1]. This is key information an AI agent can use to better assist the researcher. The document is also an appealing environment for researcher-agent collaboration, but requires careful information management strategies to prevent unwanted distractions [F3]. Thus, we aim to 1) allow the researcher to use familiar document editing affordances to interact with the agent, and 2) strategically manage outputs to avoid excessively cluttering the document.

DG2: Allow flexible delegation of agency between researcher and agent. Researchers may not want to delegate all parts of a plan to AI [F2, F3]. This preference may also be context-dependent and constantly shifting. Our system uses a concrete implementation of a flexible delegation approach proposed by Satyanarayan [92].

DG3: Provide opportunities for researcher-AI collaboration in both planning and execution. Agent developers have called for decoupling planning and execution to avoid fruitless execution [38]. Indeed, participants discussed *planning* (i.e., breaking down a problem into smaller, more actionable tasks like they did in their project documents) and execution (i.e., completing steps in the plan and refining outputs, with or without AI assistance) as separate but complementary actions. We thus see planning and execution as two distinct stages for fertile researcher-AI collaboration. Because these two stages are closely intertwined and synergistic, we aim to interleave the two in our system.

4 COCOA: SYSTEM WALKTHROUGH AND IMPLEMENTATION

We present Cocoa, a system that embeds an AI agent into a document editor using a novel interaction design pattern we call *interactive plans*. Interactive plans allow users to collaboratively plan (**co-plan**) with the agent—the agent proposes an initial plan of execution to tackle a user request that the user can edit to their liking. Then, users can collaboratively execute (**co-execute**) the plan with the agent—the user and the agent can build off each other's work to synergize human and AI capabilities, and provide flexible negotiation of human and machine agency.

To illustrate the features and functionality of Cocoa, we follow the journey of Nóirín, a researcher in human-AI interaction, as she

uses her project document to further explore open questions in interactive interfaces for AI agents.

4.1 Co-Planning

4.1.1 Invoking the agent and selecting plans. Nóirín's project document is an informal, reverse chronological log of research progress and includes information such as meeting notes, rough ideas, links to literature, and questions for herself and her collaborators. She has a specific question in her document that she wants to explore further, but is not sure how to proceed on her own: *"What are new ways AI agents can interactively elicit human feedback?"*

To initiate co-planning, she highlights the question to invoke the agent on it using a button that appears in a floating menu. The agent analyzes the question within the context of her entire document and proposes a few different plans for approaching it. These plans are displayed in the document via a **plan selector** UI within the editor, which allows Nóirín to browse and select a plan. Upon Nóirín's selection, a plan becomes fully interactive within the document (Figure 3).

4.1.2 Agent steps and user steps. The initial plan included steps for brainstorming relevant topics, searching for papers, and making connections between papers (e.g., Figure 3). Nóirín wants to assign tasks that are relatively low risk but high effort to the agent, such as searching for papers and expanding on some of her preliminary ideas, and keep tasks requiring higher-level thinking and deep research expertise to herself, such as identifying papers that can seed deeper exploration and surfacing connections between papers. To do so, she uses the **step assignment toggle** (Figure 4) to assign the step to herself (a "user step") or the agent (an "agent step"). User steps are highlighted in blue. When executing the plan, the agent will automatically attempt agent steps but will request Nóirín's input on user steps.

4.1.3 Editing the plan step description. Upon further review of the plan, Nóirín is now curious about authors who have published on "interactive feedback mechanisms in AI" rather than seeing papers on the topic directly. She edits the **step description** to reflect this. The step description can be edited just like editing any other bulleted list in her document, making this intuitive for Nóirín. If she chooses, Nóirín can also employ LLM assistance to suggest a step to replace the current one, informed by the previous steps (Figure 5). Nóirín can easily undo the suggestion if she prefers the original step. Once Nóirín is satisfied with her editing, she saves her changes with the keyboard shortcut `Cmd/Ctrl+S` to register them with Cocoa.

4.1.4 Replanning. At this point, Nóirín's edits have changed the plan's trajectory—the next step, which has the agent suggest common themes in papers, is now irrelevant because none of the previous steps retrieve papers. The **system automatically detects this and replans subsequent steps** accordingly by removing those steps and "autocompleting" the plan with new ones (Figure 6). However, before making these changes, Cocoa notifies Nóirín and asks whether she would like to proceed with replanning; Nóirín accepts. Just like with full plans, Cocoa will present a selection of new steps for Nóirín to choose from. She selects an option to complete the plan.

4.1.5 Adding and deleting steps. In Cocoa, Nóirín can easily **add and delete steps from the plan**. She notices that the agent has proposed a summary step that may be irrelevant, so she highlights the entire step and hits `Backspace` to delete it from the plan. This interaction mirrors how one would delete an item in a bulleted list elsewhere in the document. She also thinks it may be useful for the agent to take her ideas from the last step and offer some constructive critiques based on existing literature. She uses the keyboard shortcut `Cmd/Ctrl+Enter` while selecting the last step to add a new step below it, and edits the step description of the newly added step accordingly. After saving her changes, Nóirín has made all her desired edits to the plan.

4.2 Co-Execution

Now that Nóirín has made her desired edits to the plan, she is ready to co-execute it with the agent. There are two modes of co-execution: continuous and stepwise. Each step has a **step completion indicator** that indicates whether the step has not yet been run (a bullet point), is in progress (a spinner), requires user input (a question mark badge), or is complete (a checkmark). Drawing inspiration from computational notebooks, these indicators offer Nóirín a glanceable way to stay informed about the agent's progress and where she needs to take action in the plan.

4.2.1 Continuous and stepwise execution. Nóirín wishes to run through the entire plan once to see the output. She triggers *continuous co-execution* (Figure 7A) with the **[Run all]** button at the top of the plan. In **continuous co-execution**, the agent will automatically continue onto the next step as soon as it (or the user) has completed the previous one, until it reaches a user step or the end of the plan.

When she iterates on the plan in subsequent runs to refine specific steps and outputs, Nóirín may opt for **stepwise co-execution** (Figure 7B). This allows Nóirín to run one plan step at a time, like running individual cells in computational notebooks. When Nóirín hovers her mouse over the bullet point of the first plan step, the bullet point turns into a play button that, when clicked, will run only that step. Unlike some computational notebooks, however, Cocoa does not permit out-of-order stepwise execution[2] because some steps may rely on the outputs of previous ones.

Nóirín can flexibly switch from stepwise to continuous co-execution at any point by clicking the **[Run remaining]** button at the top of the plan. Conversely, Nóirín can switch from continuous to stepwise co-execution by clicking on the **[Pause after this step]** button during execution and manually running subsequent steps.

4.2.2 Completing user steps. The agent completes the first step of the plan and arrives at a user step that Nóirín assigned herself. Here, the agent requires Nóirín's input to continue, and Nóirín can see this because the step has been highlighted in orange. When she clicks into that step, an interactive sidebar opens to the right (Figure 8). The sidebar offers her a built-in paper search functionality connected to the Semantic Scholar academic database, and she uses this to add some relevant papers she has in mind. She clicks Save, adding the papers and their metadata to a plan-specific context

[2]We also note that out-of-order execution in computational notebooks is a major user pain point identified in prior academic work [16, 57] and industry reports [30, 44].

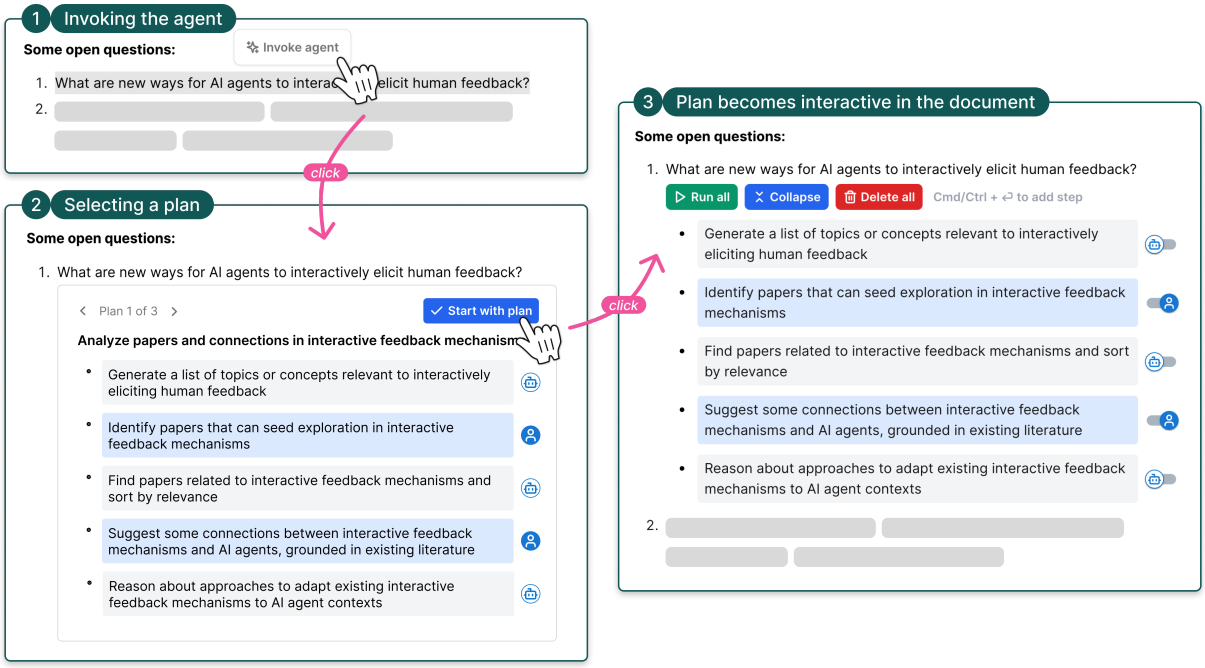

Figure 3: A user invokes the agent on a piece of text in the document by clicking on the "Invoke agent" button that appears on hover whenever text is highlighted. The agent will use the highlighted text and context from elsewhere in the document to propose a series of plans, displayed in a plan selector that appears under the highlighted text. Once the user selects a plan, it becomes fully interactive in the document.

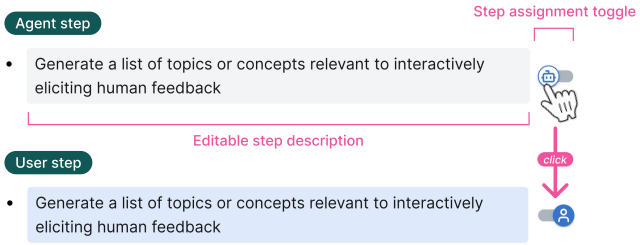

Figure 4: For each plan step, the user can assign the step to either themselves (a user step) or the agent (an agent step) by using the step assignment toggle, as well as edit the step's description.

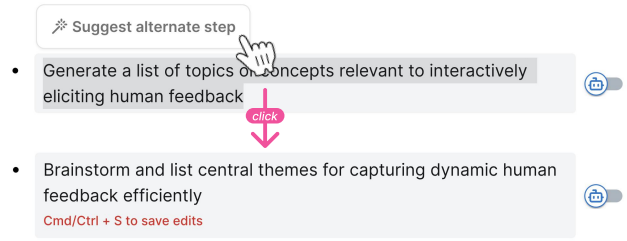

Figure 5: Highlighting text within the step description will bring up an option for the agent to suggest an alternate step. The user can undo the suggestion or save to accept it.

pool that the agent references when completing future steps. The agent then proceeds to complete the next step with this updated context to perform a more extensive paper search given the seed papers Nóirín provided.

4.2.3 Editing outputs of agent steps. The agent has now completed the third step, where it conducted a paper search guided by Nóirín's seed papers from the previous step. Nóirín clicks into that step, opening a sidebar similar to the one she used in the previous step to add papers, this time populated with interactive paper cards. This presents opportunities for her to further guide the agent with her expertise by removing papers she considers irrelevant and adding papers the agent missed. She sees a couple of low-quality results that have incoherent titles and removes them. She then adds one paper she recalls from a past discussion with a collaborator and saves this curated list. The papers in this list, along with their metadata, are stored in the plan's context pool, accumulated across all completed steps.

The interactive UI in the sidebar dynamically adapts according to the step's *output type*, which fall into one of categories in Cocoa: **papers**, **authors**, **topics**, **entities**, and **text**. Papers, authors, and topics are drawn from Semantic Scholar and are displayed as interactive cards that Nóirín can add, delete, or search for using the sidebar's built-in Semantic Scholar search. Entities are lists of items in natural language (search queries, research questions, etc.) and are displayed as editable pills. Text is natural language text displayed in an editable text box.

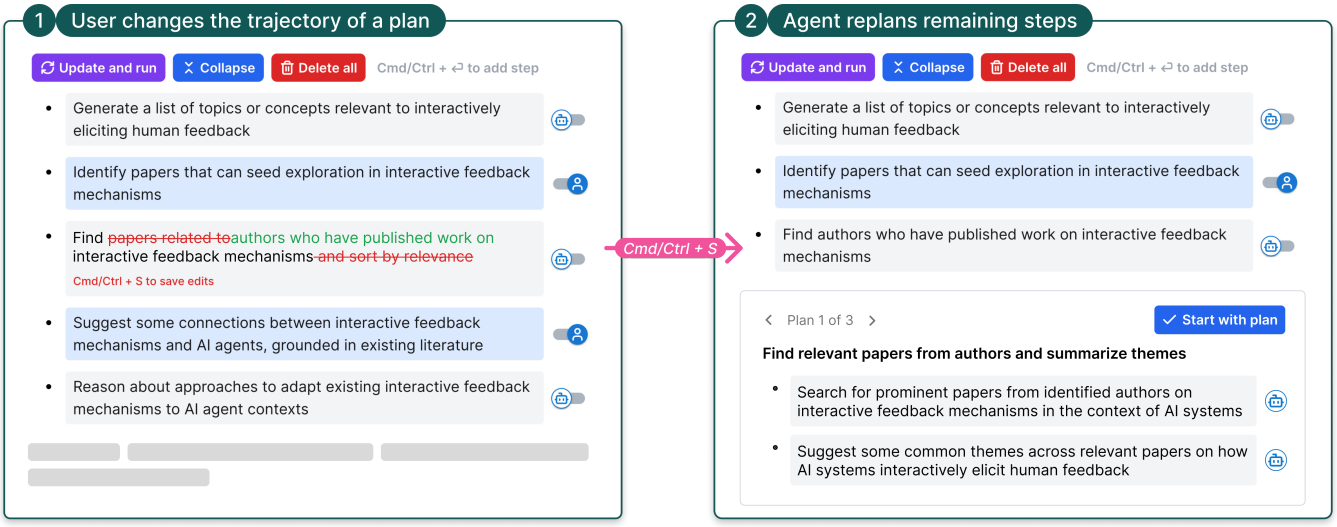

Figure 6: If major changes are made to a step that changes the rest of the plan's trajectory, Cocoa detects this and will trigger replanning. Replanning replaces subsequent steps with ones the agent suggested for "autocompleting" the plan.

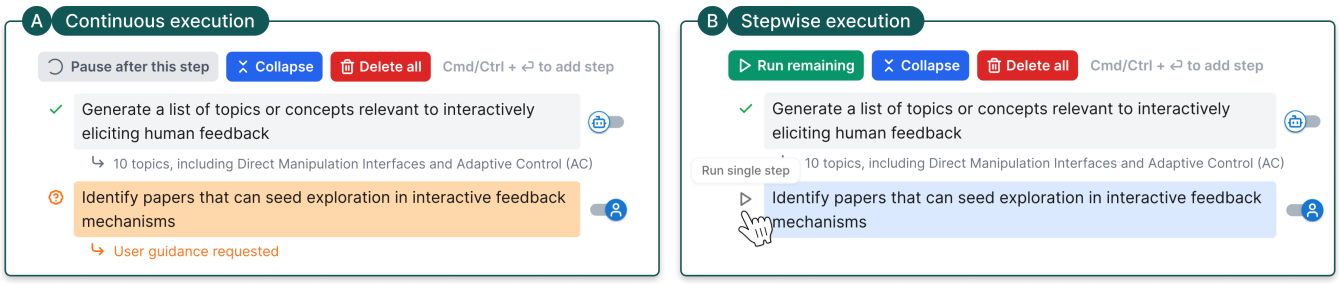

Figure 7: Users can run a plan using continuous execution (left) or stepwise execution (right). A single button click at the top of the plan triggers continuous execution, in which the agent will automatically move onto the next step as soon as it has completed the previous one. To trigger stepwise execution, users hover over the play button beside each step. The agent will not move on to the next step until the user explicitly clicks the play button for that step.

For any agent step, if the agent fails to return any output, the system will alert Nóirín and ask her to complete the step instead. These features loop in human guidance to bolster the quality of outputs in the face of weak agent performance or agent failures.

4.2.4 Plan output panel. After the final plan step has been completed, the agent adds a **plan output panel** (Figure 9) containing a modified version of the last step's output into the document just below the plan. The modification involves rewriting the output, given the context of the plan and its outputs, to more directly connect to the original user request. This brings the results of running the plan into the document so Nóirín can easily reference it in the context of other content. The panel is collapsed by default to save space and reduce clutter, but she can expand it to copy and paste parts of the output she finds useful and add them to her document. If she does not want the panel in her document at all, she can simply delete it; she can still access the outputs of all steps in the sidebar.

4.3 Interleaving Co-Planning and Co-Execution

In addition to supporting co-planning and co-execution independently, Cocoa is designed to support **synergistic interleaving of the two**. As Nóirín co-executes the plan with the agent, she encounters a relevant author that the agent identified and wishes to further explore the author's papers. She edits the following step's plan description to satisfy this. She reruns that step and receives a new batch of papers, and the agent automatically reruns the rest of the plan with the updated output. She is also not satisfied with the agent's interpretation of common themes and believes that she can surface deeper insights by reading the papers herself, so she toggles that step to be a user step. After reading the papers in detail, she returns to her document and jots down her insights. She now finds the next step repetitive, so she deletes it. Overall, her use of Cocoa is highly iterative, and she smoothly transitions between co-planning and co-execution.

So far, Nóirín has just been interacting with one plan. While waiting for the agent to complete a series of steps, she can collapse

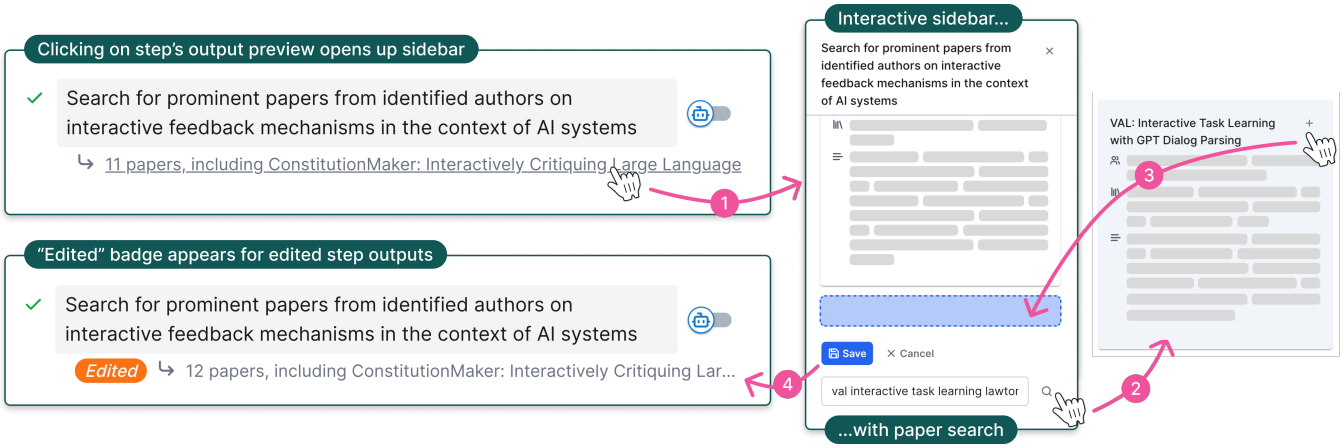

Figure 8: Users can click the output preview text beneath the plan step to access the interactive sidebar (1). By editing agent outputs in the interactive sidebar (2), users draw from their expertise to edit outputs, such as adding papers the agent might have missed (3), to guide the agent. Once edited, an indicator will appear in the output's text preview below the step (4).

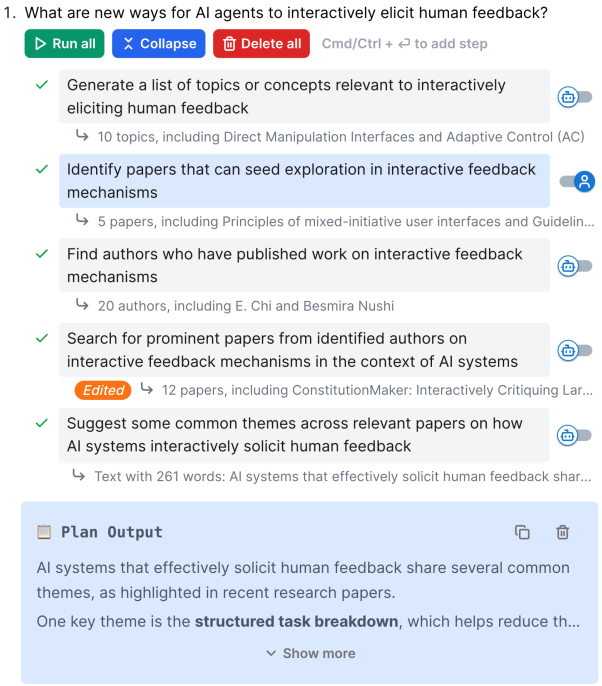

Figure 9: Once execution of the entire plan is complete, the agent inserts a plan output panel in the document with the last step's output. This allows the user to view the plan's results within the context of their document. The panel can be expanded or collapsed, and remains in the document even if the plan itself is collapsed for easy access.

the plan to hide the steps and only reveal essential information about the plan's status. She identifies a couple more areas of her document that merit further exploration and invokes the agent on them. Soon, she is co-planning and co-executing with multiple

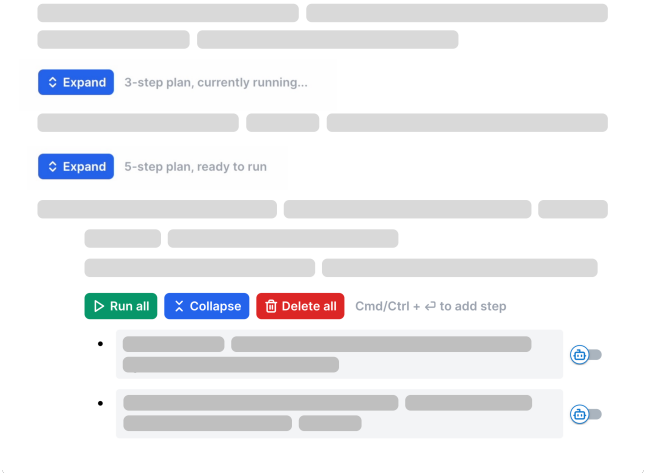

Figure 10: Users can create plans at multiple locations in their document and have the agent tackle them in parallel. Plans can be collapsed to only reveal essential information about their operating status.

agents on different parts of her document in parallel. This is not cognitively burdensome for her as she only has to attend to one plan at a time, since the others are all collapsed and running in the background (Figure 10). Finally, the document in Cocoa supports real-time multi-user editing, so Nóirín's collaborators can also leverage the agent and interact with plans.

4.4 Implementation Details

4.4.1 Technical stack. Cocoa is implemented as a web application with a Next.js and TypeScript frontend communicating with a Flask backend. The frontend uses the Tiptap[3] framework for the main

[3]https://tiptap.dev/.

document editor. Each document supports synchronous collaboration via Hocuspocus[4] and all changes are auto-saved to Tiptap Cloud, which also serves as storage for all participant documents. Interactive components within the document editor are implemented as custom Tiptap extensions in TypeScript.

4.4.2 Underlying LLM Agent. The Flask backend orchestrates LLM activity with calls to GPT-4o (scaffolded by LangChain[5]). To optimize for speed, simple actions such as plan generation or text summarization involve a direct call to GPT-4o. For plan steps that required access to the scientific literature, we created a custom tool-calling LLM agent powered by GPT-4o that had two tools that allowed it to access the public Semantic Scholar API[6], which covers paper/author/topic search, and the *Ask this Paper*[7] feature for paper summarization and question-answering. While GPT-4o provides APIs for managing multi-turn conversation context, the system manages its own context, which includes previous plan steps and their outputs, and calls GPT-4o's single-turn completion API. All prompts used are provided in Appendix I.

4.4.3 Guidance for plan generation. We use findings from our formative study to guide the agent's generation of initial plans. In our formative study, we had asked participants to write down brief plans when interacting with our probe (see Appendix C). Based on participants' preferences of which steps they prefer assigning to an AI vs. keeping for themselves (**F3** from our formative study), we organized these steps into agent and user steps. We then wrote examples of how these steps are composed into plans for tackling particular questions, once again drawing from plans participants wrote in the probe activity. These examples were provided for in-context learning in Cocoa's system prompt for plan generation (see prompts in Appendix I).

We also enable Cocoa to learn from interactive plans the user has previously created and edited. When the agent is invoked, Cocoa collects the existing plans in the document and adds them to the in-context learning examples on-the-fly. This allows the user's co-planning efforts to be reused for similar requests elsewhere in the document.

5 LAB STUDY

We conducted a within-subjects task-based lab study with 16 researchers to evaluate Cocoa and its interactive plan design against a strong chat baseline. Specifically, our study aimed to address the following research questions:

- **L1:** How does Cocoa compare to our chat baseline for ease of use, steerability, and general utility in research project documents?
- **L2:** When did researchers prefer interacting with an AI agent through interactive planning versus our chat baseline?
- **L3:** What kinds of steps did researchers wish to assign to an agent and themselves in practice?

[4]https://tiptap.dev/docs/hocuspocus/.
[5]https://www.langchain.com/
[6]https://www.semanticscholar.org/product/api
[7]https://www.semanticscholar.org/product

5.1 Participants

We recruited 16 Ph.D. and postdoctoral researchers (14 Ph.D.s, 2 postdocs; 10 female, 6 male) in computer science (CS) or CS-adjacent areas via university mailing lists, word of mouth, social media recruitment messages (on Twitter/X, Mastodon, Bluesky), and personal connections. We recruited on a first-come, first-serve basis as we conducted our studies and closed recruitment when we approached data saturation. Further details about our recruitment process, pilot studies, and participants can be found in Appendix F and Table 2 of Appendix E.

5.2 Baseline System

The baseline system (Figure 11) was a chat interface powered by the same LLM agent used in Cocoa: it used GPT-4o with access to the same tools for completing literature-related tasks using a combination of Semantic Scholar and standard LLM capabilities (summarization, accessing knowledge stored in weights, etc.). While we could integrate interactive plans within chat and vice versa (more on this in Section 8.1), we opted not to for a cleaner comparison of the two design patterns. The design of the chat interface closely resembled that of popular LLM chatbots (e.g., ChatGPT, Claude). During the study, participants positioned the chat interface beside the baseline document editor, which is the same editor that Cocoa uses, except we removed the option to invoke the agent to eliminate all co-planning and co-execution interactions.

5.3 Research Task and Study Procedure

Because participants identified literature search and understanding as one of the areas they can benefit most from AI assistance (Section 3.2.2), we had participants tackle a literature-augmented task in both Cocoa and the baseline. By *literature-augmented*, we mean tasks that are to be completed by referencing or drawing from academic literature; in the study, participants worked on not only literature review tasks, but also tasks where researchers needed to make literature-informed decisions such as study design and ideation. This stands in contrast with tasks focused on writing mechanics (e.g., rewording a sentence) or tasks that do not involve literature (e.g., booking travel) that past work has covered (e.g., [54, 60, 68, 111, 112, 116]).

To make the study more realistic and grounded in real-world research projects, tasks for this study were open questions or unfinished items from participants' existing project documents. Details for how we controlled for the length of the study and ensured the two tasks were valid and comparable can be found in Appendix G. Tasks were randomly assigned to Cocoa and the baseline.

The first author (study facilitator) conducted 1:1 studies with the 16 participants over Google Meet between October and December 2024. Each study was 90 minutes in length. The study started off with introductions and a brief overview, followed by the two tasks, which were counterbalanced across participants. Detailed procedures for each condition can be found in Appendix [x]. After each condition, participants filled out a short evaluation form with 5-point Likert scale questions for that condition (see Appendix H). The task-based portion of the study took up 75 minutes. The study concluded with a semi-structured interview that lasted around 15 minutes. Participants were asked to reflect on their experiences

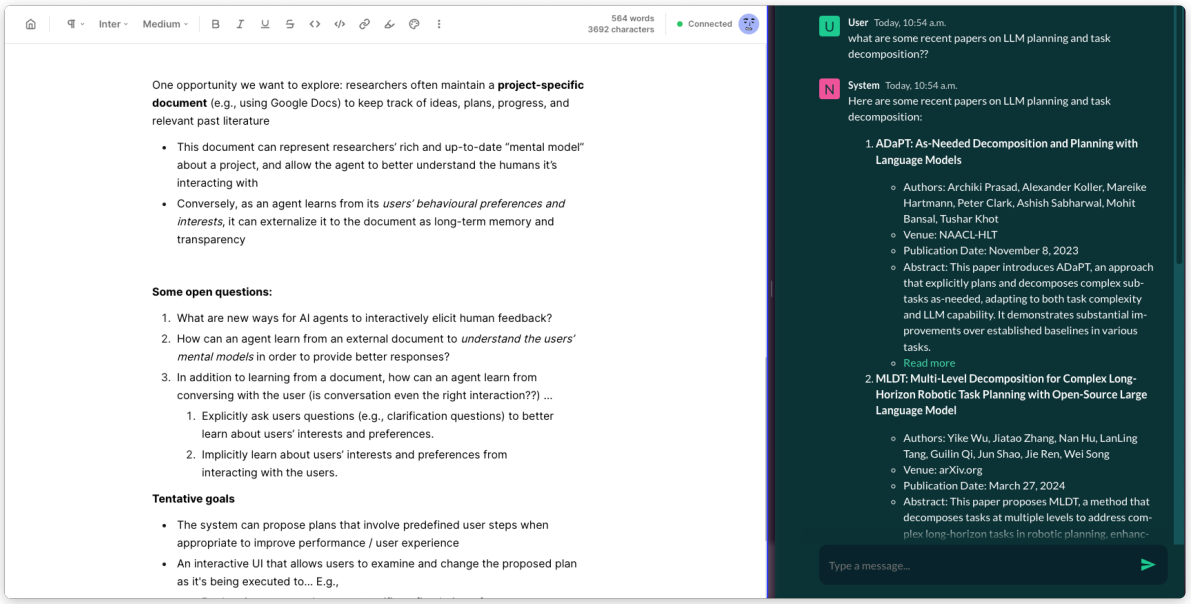

Figure 11: Participants' typical setup for our baseline condition. On the left, the same document editor from Cocoa but without an option to invoke the agent for co-planning and co-execution. On the right, the agent is situated within a chat interface.

across the two systems and discuss the pros and cons of both. They were also asked about particular decisions the study facilitator observed when using each system.

Participants received a $75 USD honorarium upon completion of the study. All studies were recorded and transcribed by Google Meet. This study was reviewed and approved by our organization's internal IRB.

5.4 Data Analysis

The first author analyzed the recording transcripts using Braun and Clarke's reflexive thematic analysis [11]. This approach uses a hybrid inductive-deductive approach to iteratively surface codes and themes across the data. We paid close attention to codes related to participants' comparisons of their experiences across the two systems and gradually grouped them into themes. NotebookLM[8] was used to help iterate on themes and discover new ones. We performed statistical tests of participants' ratings on the evaluation forms with the Wilcoxon signed-rank test (with the Bonferroni correction for our multi-rating analysis on significant results) given the non-parametric nature of our data. We saved all documents used in the study as well as all conversation histories in the baseline system, and referred back to them when needed.

The first author also coded all video recordings from the study to note the timestamps at which participants engaged and disengaged in a particular interaction. In Cocoa, these interactions were **co-planning** (see 4.1), **co-execution** (see 4.2), and **output inspection** (when the participant passively inspected the system's outputs). The closest equivalent interactions were used as codes in the baseline: **prompting for planning** (instructing the agent to perform an action without any attempts to modify the agent's earlier outputs), **prompting for editing**[9] (instructing the agent to modify or only consider a subset of its output in future actions), and **output inspection**. During video coding, the first author also captured the number of **input items** (items generated as part of the output of a plan step or conversation turn) and **output items** (items fed as context into the next plan step or conversation turn) for all plan steps or turns that output discrete items (i.e., papers, authors, topics, text entities).

6 LAB STUDY RESULTS

A high-level overview of participants' interactions with Cocoa and the baseline is depicted in Figure 12. In both conditions, output inspection occupied the most time out of the coded interactions, but was lengthier in the baseline than Cocoa (54.1% vs. 33.6% of the approximately 25-minute session). A paired t-test found this difference to be significant ($t(15) = 3.95, p < 0.001$). Participants also engaged in co-execution in Cocoa significantly more often than the closest equivalent interaction—output editing via prompting—in the baseline (15.2% vs. 3.1%, $t(15) = 4.29, p < 0.001$). They spent slightly more time on co-planning (21.5% vs. 17.4%), although this was not significant ($t(15) = 1.01, p = 0.33$). Our analysis indicates that, when provided with more affordances to interactively modify the agent's outputs, users will take advantage of those opportunities for active engagement and shift away from passive output consumption.

Could more output editing activity mean that participants are less satisfied with Cocoa's outputs? To answer this, we analyzed

[8]https://notebooklm.google.com

[9]Since there were no user steps in the baseline, the closest equivalent interaction is attempting to edit the agent's output via prompting.

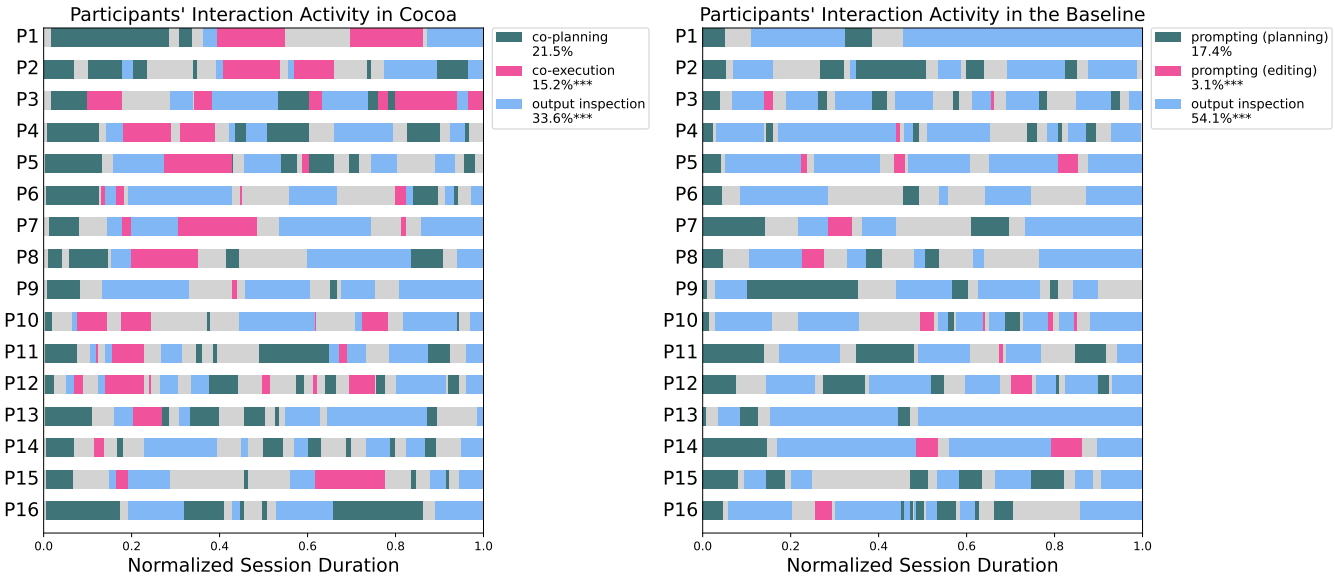

Figure 12: Participants' interaction activity in Cocoa and the baseline within the study sessions. * indicates a statistically significant difference ($p < 0.001$ via a paired t-test) with the closest equivalent interaction in the opposing condition. Gray areas represent spans of time where the participant was not engaging in any of our coded interactions because they were answering a question posed by the study facilitator and/or waiting on the system. Definitions for all coded interactions can be found in Section 5.4.**

the *output-to-input item ratio* for each round of interaction—where a round is a plan step in Cocoa or conversation turn in the baseline—across all participants. The output-to-input ratio tells us what percentage of items generated by a round are carried over as context to be used in the next round. As can be seen in Figure 13, the ratio is much higher in Cocoa ($M = 97.5, \sigma = 23.0$) compared to the baseline ($M = 0, \sigma = 33.0$). This difference is significant per a Mann-Whitney U Test ($U = 1722.0, p < 0.001$). These results suggest that the flow of data within the two systems differ fundamentally: the flow of items from one plan step to the next is rather fluid in Cocoa where much results were carried over to the next steps in the plan, but disjoint in the baseline where most or all of the results were unused. This suggests that participants, when using the baseline system, more often focused on few items in the output, or pivoted to a different task entirely. Thus, there is no indication that increased output editing activity is due to dissatisfaction with system outputs.

In the rest of this section, we present quantitative and qualitative analyses of participants' data from our user study. We group these results by our research questions from the start of Section 5.

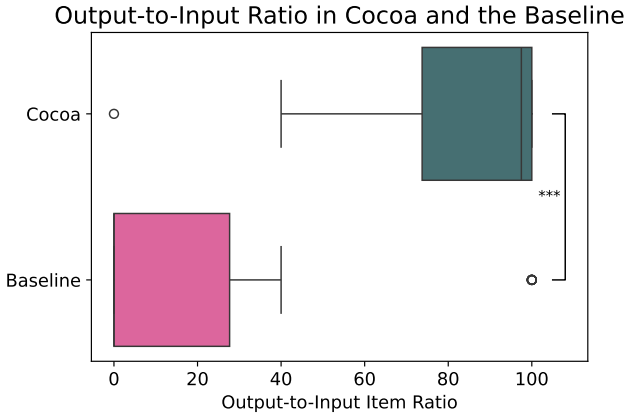

Figure 13: The output-to-input ratio in Cocoa and the baseline, across all rounds of interaction from all participants ($n = 16$). * indicates $p < 0.001$ via a Mann-Whitney U Test.**

6.1 Steerability, Ease of Use, and Utility (L1)

We denote the median rating of our baseline and Cocoa as M_b and M_c, respectively, and the Wilcoxon test statistic as W.

Introducing novel interactive systems with additional affordances can often lead to higher effort when using the systems as a trade-off for better utility [37, 97]. However, based on participants' post-task ratings on a 5-point Likert scale, there was no significant difference in perceived *ease of use* between the baseline and Cocoa with 16 participants ($M_b = 4.5, M_c = 4, p = 1.000, W = 7.50$). At the same time, we observed a significant difference that Cocoa provided better *steerability*. In response to the question *"I could easily steer the system towards doing something helpful,"* participants rated Cocoa higher than the baseline to a significant degree ($M_b = 3, M_c = 4, p = 0.005, W = 0$),[10] with $W = 0$ indicating that all participants rated Cocoa's steerability as greater than or equal

[10]Here, we set the significance threshold to $\alpha = 0.05/3 = 0.015$ using Bonferroni correction. Our result is significant post-correction: $p = 0.005 < 0.015$.

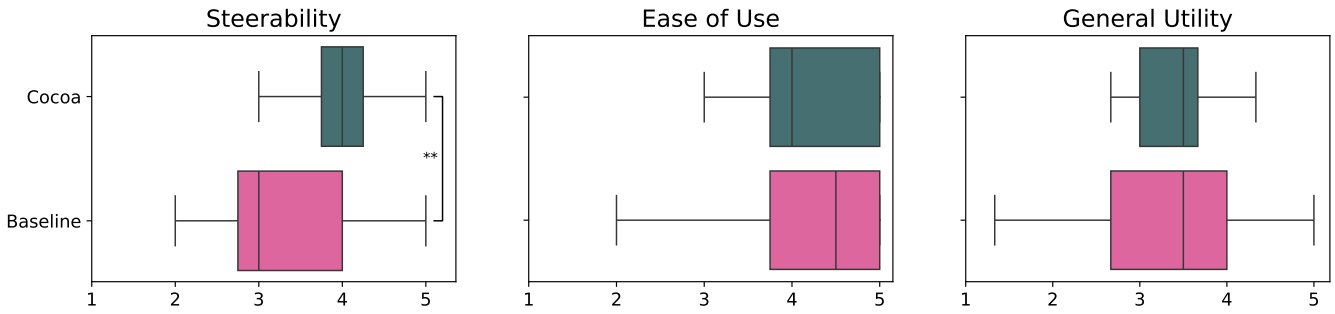

Figure 14: Participants' Likert scale ratings of steerability, ease of use, and general utility across our baseline and Cocoa. ** indicates $p < 0.01$ via a Wilcoxon signed-rank test.

to that of our baseline. In sum, these results suggest that by taking inspiration from familiar interaction patterns of computation notebooks and text editors, Cocoa provided richer affordances, and does so without sacrificing ease of use to better support human-AI collaboration compared to the simple chat interface used in the baseline condition (Figure 14).

We also tested for differences in *general utility*, which was a composite metric consisting of three measures broadly related to the usefulness and insightfulness of system outputs, but we found no significant differences ($M_b = M_c = 3.5, p = 0.7, W = 40$). This was unsurprising— the systems' utility to the researcher may depend on a range of system-agnostic factors beyond our control (e.g., contextual interpretations of outputs by researchers, the extent to which researchers thought about the task before the study, etc.) However, improved steerability may have allowed users to steer the agent away from providing downright unhelpful outputs in Cocoa, as indicated by fewer lowly-rated outliers (see Figure 14).

Additionally, the nature of the task may have influenced perceived utility, as Cocoa might be better suited for certain tasks, while chat may work better for others. Qualitative insights from post-task interviews and participants' think-aloud sessions provide insights into the task suitability of interactive plans (see Section 6.2.1).

6.1.1 ***Co-planning afforded steerability.*** Qualitative results based on think alouds and post-interviews showed that participants found various co-planning features provided by Cocoa helped improve the steerability of the agent. Many thought that the ability to compose, edit, and rerun plan steps allowed them to better "fine-tune" the agent's outputs and their own research process. P1 was inspired to *"use [plan steps] as building blocks"* to create a custom workflow. P7 and P15 both appreciated the ability to edit and rerun steps, which enabled them to quickly *"organize thoughts and iterate on ideas"* (P7); P15, specifically, iterated on "paper search" step, reran it to cover papers from venues that they preferred, and was able to obtain more relevant final outputs with running the rest of the steps in the plan. Another strategy was "backtracking" to a specific step and editing it which P6 found to be much more helpful than digging through a chat conversation: *"With chat, when it does something wrong there's not really an easy way to fix it because there's no concrete steps that it's following, whereas [with Cocoa], I can go back and be like, let me have it do something else here."*

Besides iterating and steps by editing and rerunning them, we also observed participants adding and removing agent-suggested steps to improve the plans. For example, P12 added a step at the end of their plan to have the agent suggest some potential next steps to pursue after seeing its paper summaries, while P5 removed a less relevant intermediate step on searching for papers related to autonomous vehicles to avoid distractions from this tangential topic.

6.1.2 ***Co-execution afforded steerability.*** Participants appreciated the flexibility to choose between stepwise and continuous plan execution. Many chose stepwise execution because it allowed them to better control the overall process and prevent error propagation. P4 and P11 both wanted to *verify* and *manually curate* the list of papers returned by the agent from a paper search step before proceeding to subsequent steps. P11 specifically cites resource efficiency as a practical reason: *"If I update [outputs of] the first or second step it would need to rerun the following steps. I just want to reduce some API usage."* P10 similarly mentioned that they would "waste energy" running subsequent steps if they detected errors in initial steps. Others preferred continuous execution. P8 and P9 thought continuous execution was more efficient because it parallelized human and AI efforts: *"Running all is a little bit of parallelization in my head where I can get the next thing rolling, and if I end up not having to edit it, that means we're already moving ahead"* (P8). P15 stated that while they opted for continuous execution, their choice depended on the familiarity with the task: "*if it were a topic that I felt like I didn't already know what type of literature I would be looking for or what the output should look like, I would go through them individually and maybe use it as more of a literature exploration tool. But in this case, I already know in general what types of papers I want to see it come up with.*" This underscores the importance of providing users with a choice of execution method to cater to researchers' contextual needs and preferences.

The ability to directly edit the agent's outputs via direct manipulation by deleting irrelevant items and adding items of their own also improved steerability. Many participants, including P6, liked the precision with which they could edit the output of a particular step, which is much more difficult in a chat interface: *"With a conversation agent, even if I'm really good at prompting it, I have to redo the prompt and keep changing that original prompt." (P6).* This feature also allowed P2 to quickly pivot and repurpose the

output when the agent produced unexpected (albeit helpful) paper search queries: *"I was looking for more application intervention based stuff, but this is great because now I can actually use these keywords to broaden my horizon of how I was thinking about these solutions, which is I think interesting. I'm just trying to remove a couple [for future steps]."* P15 thought direct manipulation enabled more granular control than chat, allowing them to easily guide the agent with their expertise: *"[Cocoa] was easier to control because [the outputs] are so specific. My control is scoped down to these very small tasks. If I want to add more papers, that's easy to do."*

6.2 Task-Specific Preferences for Interactive Planning vs. Chat (L2)

It is reasonable for participants not to strictly prefer Cocoa over our chat baseline for every research task. After all, chat is a strong baseline because it is flexible, easy to use, and ubiquitous. Here, we aim to surface participants' insights on when they prefer interactive plans over chat, and vice versa. Ultimately, many participants recognized that both interfaces have their strengths and weaknesses, and, in P13's words, it was *"not fair to say one is better than the other."* Instead, participants imagined using the two systems at different stages of their research. In Section 8.1, we further discuss the costs and benefits between the two paradigms and how to combine them in future systems to get the best of both worlds.

6.2.1 Preferences for interactive plans. Participants generally agreed that when tackling questions or tasks that required a structured or organized approach, they preferred Cocoa over chat. One example of such tasks is literature review and synthesis. P12 considered the plan more usable as an end product than a conversation: *"I think it's a better way to have a finished draft [plan] that I can directly use as opposed to trying to dilute a plan in a conversation."* Others appreciated that the plan helped them stay organized and consolidated lot of information: *"[the plan] is a really good way to organize my thoughts"* (P14) and "*everything can be part of one consolidated [artifact] which I can refer to again*" (P7).

The structure of an interactive plan was also helpful when participants had an idea of their desired final output, but were unsure of how to get there. For their task, P6 appreciated Cocoa because they needed *"some concrete steps even if I need to iterate over those steps,"* and because the agent's first step *"helped me get to search terms [for papers] which was what I was struggling with."* Similarly, P14 said that plans can help them overcome mental blockers when faced with a tough problem: *"If I'm really stuck and frustrated, I can click a single button, and it could output a whole plan for me. That would make me feel really safe because it's always here outputting something for me which potentially get me out of this frustration."* For P16, they did not consider anything but a structured approach to be acceptable for research. They envisioned the research process as *"a lot of subtasks that you just come up with"* and collected information to be *"concrete items"* that they would operate on with *"high-level actions [like] directly extracting insights."* As a result, Cocoa aligned well with their mental model and found the chat *"very frustrating [...] and almost like the opposite"* of their preferred approach.

6.2.2 Preferences for chat-based interactions. While interactive plans afford more steerability, chat-based interactions may be more appropriate when the task at hand is *open-ended and difficult to create plans in advance*, or too narrowly scoped to benefit from a multi-step plan. Many preferred chat for freeform exploration, where the nature of the desired final output was ambiguous. P2, while conducting an early brainstorming task, thought it was *"a little overwhelming to go through all these steps, and I also felt this kind of pressure to stick with the plan although I was editing it."* P1 suggested that, when brainstorming, Cocoa could show them *"a single step at a time"* and allow them to *"choose from options for the next step"* after they have seen the outputs for the previous step. Question-answering was another category of tasks where chat was preferred, where *the next step depends on the output of the previous step.* P13 found it much easier to *"keep on asking [follow-up] questions"* to drill down on a specific line of inquiry in chat.

6.3 Task Assignment to Agent vs. User (L3)

In our formative study (Section 3.2.3), participants shared that they preferred to assign tasks related to information retrieval and brainstorming tasks to the agent while keeping higher-level reasoning and synthesis tasks for themselves. In our study, however, most participants toggle all plan steps—even higher-level ones assigned to them by default—to the agent. Some participants chose not to include user steps in their plans because they were curious about the agent's capabilities and limitations—a natural tendency when learning to use a new system. For example, P1 wanted to see "*how the bot does it*" while acknowledging that they prefer to "*make the decisions and here, these are more exploratory steps rather than decision-making ones.*" P13 also opted for this approach and considered the agent's errors to be low-stakes because they can directly modify the outputs at any time: *"when I don't like something, I can just easily remove it."* P9 also felt assured that they could edit the outputs and did not consider the output quality to be particularly problematic: "*usually as long as it's somewhat helpful and it's just running in the background, I would take a look at [the output].*" Adopting a cost-benefit framework, they mentioned that they would leave a step as an agent step as long as *"the cost of looking at the agent's output is less than the benefit of whatever insights I obtain."*

While we did not seen substantial use of user steps in the lab, participants also did not view the ability to create user steps negatively. Many speculated in the interviews that they would leverage this mechanism more if they had more time to interact with the system and outside of the lab environment. P8 explicitly said that *"I assign everything to [the agent] because I don't think we have that much time for me to refine my [own steps]."* This further motivated us to conduct a longitudinal deployment study of Cocoa, and Section 7.2.1 details how participants' task assignment strategy differs in the field.

7 DEPLOYMENT STUDY AND RESULTS

As surfaced in Section 6.3, participants' usage behaviors in Cocoa may shift given more time to interact with the system. To obtain a more holistic understanding of how Cocoa can support researchers in their day-to-day work, we conducted a 7-day field deployment study with 7 participants.

7.1 Participants and Procedure

We invited all participants from our lab study to participate in this field deployment. 7 participants (P3, P5, P10, P11, P12, P15, P16) agreed to participate.

Participants used Cocoa to work on a real-world, in-progress research project of their choice over a 7-day period. Before the study, the study facilitator sent each participant a user manual documenting Cocoa's functionalities. The manual also contained some general guidelines for the week (see Appendix J), including aiming to spend at least 90 minutes with the system throughout the duration of the study. Each participant could log onto their personal workspace in Cocoa with a set of credentials provided by the study facilitator. We used system logs to verify that there was at least semi-regular activity for each participant throughout the study.

At the end of the 7-day period, participants completed a 30-minute semi-structured exit interview where they were asked about their general experience with the system as well as specific co-planning and co-execution interactions the study facilitator identified from within their workspace. All interviews were recorded, transcribed, and analyzed with the same qualitative analysis procedure described in Section 5.4. Participants received a $100 USD honorarium upon completing the exit interview.

7.2 Field Deployment Study Results

7.2.1 ***Step assignment strategy changed: more user steps for complex longitudinal tasks.*** Compared to the lab study, where most co-execution was focused on removing or adding to an agent step's output, participants in the field study assigned more plan steps to themselves. These user steps often involve high-level strategic decision-making, writing and developing arguments, running experiments and evaluations, deep reading of literature, and articulating core research ideas and novelty. The ability to assign steps to the agent and themselves helped participants create a back-and-forth workflow with the agent. For example, P15 first figured out *"broad buckets of literature or themes or topics that I need to talk about"* before passing those to the agent to *"help me brainstorm ways to connect those bodies of literature."* P3, who had already conducted some interviews for their project, first had the agent look up key literature before assigning themselves a step to *"pass in more of the dynamics from my interview notes"* and then had the agent propose interventions based on the literature and notes. For many participants, user steps came later in the plan because of their high-level nature. P11 often assigned the last plan step to themselves because *"it was about the main innovation in the project idea [...] no one has [done it] before so I don't trust the agent."* P16 assigned themselves the last step as a to-do item and a reminder to reason about how the agent's output *"could actually be tied into some of the document."* Participants also indicated that their step assignment also depended on the stage of research they were in. P3 admitted that *"if I was just doing initial discovery and study planning, I would have gone more agent [steps]."*

7.2.2 ***Interleaving co-planning and co-execution was valuable.*** Participants identified the interleaving of co-planning and co-execution afforded by interactive plans as a unique and valuable aspect of Cocoa. This interleaving allowed for a flexible and iterative workflow. P12 shared that *"I don't need to run the [earlier] retrieval step again. I know that those papers are relevant"* but instead could focus on *"only improving the second step [...] to identify something different from the literature."* Similarly, P16 felt like they could easily *"run the plan fully and then go in and adjust the rerun things,"* which enabled them to "debug" suboptimal parts of the plan: *"by forcing these checkpoints of brainstorming search queries and assembling lists of papers, it helps me get a sense of what parts of the process are good and which parts need work."* Outputs from co-execution also inspired new approaches for participants to *"iteratively build upon a plan or reshape a plan"* (P5). Participants even mentioned that they wanted similar interleaving strategies integrated into other LLM-powered systems; for P11, it was *"one of the improvements I want to see in LLM [reasoning] models"* so that users can force an early stop to the reasoning or steer it in more productive directions.

7.2.3 ***Cocoa was most useful for literature synthesis and early-stage project planning.*** All participants agreed that Cocoa was especially helpful for literature discovery and synthesis. P15 found the plan steps involving literature search "*super useful*" and the search query generation steps that commonly preceded them to be *"really helpful for refining my thought process because it made me think more carefully about what I'm actually trying to look for."* P10 also found Cocoa to be *"really helpful with looking for papers"* and that it was *"less burdensome"* to use compared to Google Scholar. The combination of plan steps involving search query generation and paper search using those queries left P10 with *"more faith that I was getting a good coverage of the field."* P3 was impressed by Cocoa's performance on thematic summarization steps: *"[Cocoa] did a really good job at grouping papers that went together."* P11 also appreciated Cocoa's speed at literature search and synthesis and found it conducive to early-stage explorations: *"if I were to come up with a summary of 20 papers, it takes probably a day, but [in Cocoa] it's around a minute which is really useful to formulate early-stage research ideas."* Beyond literature search, the structure of the plan was also useful to *"break down tasks and provide a sense of progress"* (P16) and *"identifying important actionable steps I could take with a research problem"* (P12). Overall, participants found Cocoa valuable in their work, with P3 in particular sharing that *"I have an entire tab in my Google doc where we have stuff from Cocoa."*

7.2.4 ***Additional desiderata.*** Participants identified several desiderata for Cocoa after spending more time with it. Some participants wanted to clearly see what context Cocoa was used to generate plans and execute plan steps. For example, P10 was unsure whether the system was *"using the context from my other plans to generate the [other] plan."* The system indeed was, but did not visualize relevant context in the document. P3, P15, and P16 all wanted more transparency into the shared context pool (Section 4.2) used across all steps and the specific context drawn upon by each step. Several participants desired tool interoperability, such that they could swap out Cocoa's default tools for their own ones (e.g., an LLM finetuned on paper summaries instead of Cocoa's paper summarizer). This way, interactive plans would serve as a "hub" for their custom tools. Finally, P5 took inspiration from Notion's ability to nest pages to suggest nested plans, where plans can be organized and expanded

in a hierarchical manner. These suggestions all provide valuable directions for future work.

8 DISCUSSION

8.1 Chat, Interactive Plans, or Both?

In practice, rather than choosing between *either* interactive plans *or* chat when building future systems, we see research opportunities for AI agent interfaces to strategically combine the two, especially when assisting the user with complex, long-running requests.

Consider "deep research" systems that assemble a detailed report on a particular topic upon searching for and synthesizing information from around the web [27, 79, 86]. Some of these systems may clarify user requests in a chat conversation [79] or allow the user to prompt the agent to devise a different plan [27] before starting the research process, but neglect user involvement thereafter. Integrating interactive plans can be particularly valuable here because they afford iteration and control over *specific parts of planning and execution*. For example, the user may not be satisfied with the sources the agent is using to inform its answers and wants to steer the agent to use information from a narrow set of websites for a particular plan step. This kind of interaction is expensive without interactive plans because it requires re-running the entire research process. Interactive plans allow the user to focus on improving outcomes of a particular step by editing that step and rerunning it or directly editing the agent's outputs at that step. Rather than re-executing the entire plan, the agent updates only the subsequent plan steps with the new user input. After the interactive plan has finished executing, users may return to interacting with the agent in a chat interface to ask questions about specific outputs or steps. These are precisely the benefits that computational notebooks provide for data analysts [4, 16, 19].

In addition to using chat and interactive plans sequentially, we can also consider how they can be *integrated within one another*. Embedding chat within interactive plans can provide more flexibility and organization. For example, during co-planning, chat-based features can be used to specify higher-level desiderata for the proposed plans when none are satisfactory for the user. During co-execution, the user may engage in conversation with the agent at every step to iterate on outputs. This way, the interactive plan also serves as a way to organize conversation threads with the agent as the agent operates over many plan steps. Conversely, interactive plans can be incorporated into chat-first interfaces to provide more transparency and steerability. This can be useful for open-ended exploratory problems, with occasional periods of structured execution.

8.2 The Practical Significance of User Steps and Advanced Planning

In Cocoa, users are presented with a multi-step interactive plan up front, which the agent adheres to and can modify while operating. This plan also allows users to delegate a step to themselves or the agent. Both design decisions are a departure from many existing agentic AI systems, where the agent dynamically constructs a plan step-by-step as it generates outputs [114] and seeks user input only when it is unable to complete a task [108]. We discuss two practical reasons—cost and oversight—for why such a departure may be desirable for practical agents in the real world.

AI agents can be prohibitively expensive to run [49, 111]. Because an agent's actions often involve multiple LLM calls, costs can quickly accumulate, especially for more complex and long-running tasks. In response, academic projects have implemented cost-capping measures to alleviate this [111], and researchers have called for agent evaluations to be cost-controlled [49]. In user-facing interactive systems, cost also includes time spent waiting for agents to come back with results. These cost considerations change user behavior. When participants made the decision to assign most or all steps to the agent in our lab studies, they viewed the agent's actions as rather inconsequential and low-cost—they could run the agent and see what happens, and edit the output post hoc (Section 6.3). However, while this was the case in the lab, it may not be true in some real-world contexts where users and/or developers need to pay monetary and time costs for each agent step. The user steps become an important cost-saving measure. User can choose to complete a step that is not particularly burdensome or is too difficult for current AI agents to complete effectively. Efficiently leveraging human effort and expertise where appropriate via user steps can be a key step towards developing cost-aware agents that are usable and affordable in practice.

The increased autonomy of AI agents also comes with increased risks, as AI harms become more difficult to anticipate, and accountability for AI actions becomes harder to trace [12, 13]. Interactive plans not only provide more transparency into agent actions, but also provide some assurance about and control over the agent's execution trajectory. By generating a plan ahead of execution that the agent adheres to, harm anticipation and mitigation become more tractable. While step-by-step plan generation used by existing agent frameworks, such as ReAct [114], may also accomplish a similar goal by asking for user approval whenever a new step is generated, this can severely limit agent autonomy when desirable and may also disengage the user if their only form of supervision is repetitive rubber-stamping [49]. Additionally, advanced planning enables assessments of task risk pre-execution. Plan steps determined to be of higher risk (e.g., requiring working with passwords) can be assigned to the user—even if the agent is capable of completing them—as a risk mitigation measure.

9 LIMITATIONS AND FUTURE WORK

Based on our formative study, we built Cocoa as a document editor because documents are natural sites of planning for researchers and present an ideal environment for agent interaction (Section 3.2.1). However, the linear nature of a document also has its limitations and does not support "forking" plans and iterating on versions of a plan in parallel. These types of interactions are better supported by node-based canvases, which have been gaining popularity as a means of sensemaking and creative exploration with LLMs (e.g., [2, 87, 100]). Future work can explore new interaction techniques for interactive plans. The default assignment of user and agent steps in Cocoa is also a result of researchers' preferences from our formative study. Past work explored automated methods for a model to decide when to defer a task to a human expert versus acting on its own [72]. Future work may investigate applications of these methods in interactive plans and/or leverage inference-time scaling to improve plan quality [73, 78].

The underlying agent used in Cocoa also had some technical limitations. Since it was not multimodal, it was not capable of taking visual content (e.g., figures or slides) as input. The agent was also not capable of executing code—a capability which a couple of participants inquired about during the study. While it is sufficient for this current work to explore interactive plans as a novel interaction paradigm, future work can expand their utility with multimodal agents and/or agents that can execute code.

Another limitation is the demographics of our participants. Our participants were researchers in CS and CS-adjacent areas. Research culture and incentives specific to CS may bias our results and limit our imagination of how interactive plans can be used. P10, who used to work in wet labs in their undergraduate research, shared that a useful application of interactive plans would be helping wet lab researchers walk through the steps necessary to prepare for lab experiments. Future work may investigate novel applications of interactive plans in domains beyond CS.

While our work only explored the use of interactive plans by a single user, one interesting future direction is to explore interactive plans in *multi-user collaborative scenarios*. Interactive plans can potentially coordinate the efforts between *multiple human users* and an AI agent. Project documents are often highly collaborative artifacts. Interactive plans can play a similar role to existing task management tools like Trello[11] that serve as coordinative artifacts [9, 94] and broaden common ground [76] in teams. They can help plan to-dos for the team, consolidate outputs in a central location, and display progress as items in the plan get completed.

Novel challenges arise for the agent in these collaborative scenarios. When proposing plans, the agent not only has to reason about *whether* a step should be assigned to a human user, but also *who* that will be. To do this, the agent may leverage context in the form of populated profiles of team members (e.g., researchers' webpages and publication records, LinkedIn profiles outside of research). The agent may also infer expertise from the edit history of individuals within the document, similar to techniques for socially grounding AI chatbots in group chats using conversation history [105]. More generally, challenges in human-agent communication get exacerbated in collaborative settings. For example, the question of *"which preferences should the agent respect?"* [7] is already challenging in a single-user setting because it requires the agent to elicit, preserve, and reason over task context and user preferences simultaneously. Shifting to collaborative settings necessitates keeping track of multiple sets of contexts and preferences and how they interact with each other. Collaborative use of interactive plans thus leave many open questions for future work.

10 CONCLUSION

In this paper, we presented Cocoa, an interactive document editing environment for scientific researchers to fluidly collaborate with an AI agent to tackle open questions and tasks within their research projects. Cocoa introduced a new interaction design pattern—*interactive plans*—that enabled Co-planning and Co-execution between a researcher and an AI agent.

[11] https://trello.com/

Our lab ($n = 16$) and field deployment ($n = 7$) studies showed that participants not only engaged in collaborative planning and execution in Cocoa, but also interleaved co-planning and co-execution to progress on their tasks. Compared to a strong baseline with a more familiar chat interface, Cocoa enhanced agent steerability without sacrificing ease of use. In the lab study, participants' main mode of co-execution was refining agent outputs, but when Cocoa was integrated into their research workflows in the field over a 7-day period, participants took advantage of user steps to guide the agent with their expertise and exert more of their agency. Overall, our work demonstrates the potential of interactive plans as a novel design pattern to flexibly mediate human and AI agency. Importantly, our work offers many practical takeaways for the design of agentic AI systems, such as leveraging user steps and advanced planning for agent cost management and oversight.

Agentic AI systems have exciting potential to transform our digital experiences and advance long-standing visions held by HCI and AI communities. However, these transformations and advancements also demand renewed attention to strategies for ensuring effective collaboration between human users and AI agents in the real world. Our work advances emerging efforts in this new frontier of human-AI collaboration.

ACKNOWLEDGMENTS

We extend a warm thanks to all participants from our formative, pilot, and user studies. We also thank Raymond Fok and Shannon Shen for helpful discussions.

REFERENCES

[1] Anthropic. 2024. Developing a computer use model. https://www.anthropic.com/news/developing-computer-use.
[2] Ian Arawjo, Chelse Swoopes, Priyan Vaithilingam, Martin Wattenberg, and Elena L. Glassman. 2023. ChainForge: A Visual Toolkit for Prompt Engineering and LLM Hypothesis Testing. *ArXiv* abs/2309.09128 (2023). https://api.semanticscholar.org/CorpusID:262044762
[3] AutoGPT. 2024. Empower your digital tasks with AutoGPT. https://agpt.co/.
[4] Amid Ayobi, Jacob Hughes, Christopher Duckworth, Jakub J Dylag, Sam James, Paul Marshall, Matthew Guy, Anitha Kumaran, Adriane Chapman, Michael J. Boniface, and Aisling Ann O'Kane. 2023. Computational Notebooks as Co-Design Tools: Engaging Young Adults Living with Diabetes, Family Carers, and Clinicians with Machine Learning Models. *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems* (2023). https://api.semanticscholar.org/CorpusID:258218057
[5] Tamara Babaian, Barbara J. Grosz, and Stuart M. Shieber. 2002. A writer's collaborative assistant. In *International Conference on Intelligent User Interfaces*. https://api.semanticscholar.org/CorpusID:215754709
[6] Jinheon Baek, Sujay Kumar Jauhar, Silviu Cucerzan, and Sung Ju Hwang. 2024. ResearchAgent: Iterative Research Idea Generation over Scientific Literature with Large Language Models. *ArXiv* abs/2404.07738 (2024). https://api.semanticscholar.org/CorpusID:269042844
[7] Gagan Bansal, Jennifer Wortman Vaughan, Saleema Amershi, Eric Horvitz, Adam Fourney, Hussein Mozannar, Victor Dibia, and Daniel S Weld. 2024. Challenges in Human-Agent Communication. (2024). https://api.semanticscholar.org/CorpusID:270870360
[8] Gagan Bansal, Tongshuang Wu, Joyce Zhou, Raymond Fok, Besmira Nushi, Ece Kamar, Marco Tulio Ribeiro, and Daniel Weld. 2021. Does the whole exceed its parts? the effect of ai explanations on complementary team performance. In *Proceedings of the 2021 CHI conference on human factors in computing systems*. 1–16.
[9] Jakob E Bardram and Claus Bossen. 2005. A web of coordinative artifacts: collaborative work at a hospital ward. In *Proceedings of the 2005 ACM International Conference on Supporting Group Work*. 168–176.
[10] Dan Bennett, Oussama Metatla, Anne Roudaut, and Elisa D. Mekler. 2023. How does HCI Understand Human Agency and Autonomy? *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems* (2023). https://api.semanticscholar.org/CorpusID:256389761

[11] Virginia Braun and Victoria Clarke. 2019. Reflecting on reflexive thematic analysis. *Qualitative Research in Sport, Exercise and Health* 11 (2019), 589 – 597. https://api.semanticscholar.org/CorpusID:197748828

[12] Alan Chan, Carson Ezell, Max Kaufmann, Kevin Wei, Lewis Hammond, Herbie Bradley, Emma Bluemke, Nitarshan Rajkumar, David Krueger, Noam Kolt, et al. 2024. Visibility into AI Agents. In *The 2024 ACM Conference on Fairness, Accountability, and Transparency*. 958–973.

[13] Alan Chan, Rebecca Salganik, Alva Markelius, Chris Pang, Nitarshan Rajkumar, Dmitrii Krasheninnikov, Lauro Langosco, Zhonghao He, Yawen Duan, Micah Carroll, et al. 2023. Harms from increasingly agentic algorithmic systems. In *Proceedings of the 2023 ACM Conference on Fairness, Accountability, and Transparency*. 651–666.

[14] Joel Chan, Joseph Chee Chang, Tom Hope, Dafna Shahaf, and Aniket Kittur. 2018. Solvent: A mixed initiative system for finding analogies between research papers. *Proceedings of the ACM on Human-Computer Interaction* 2, CSCW (2018), 1–21.

[15] Joseph Chee Chang, Amy X. Zhang, Jonathan Bragg, Andrew Head, Kyle Lo, Doug Downey, and Daniel S. Weld. 2023. CiteSee: Augmenting Citations in Scientific Papers with Persistent and Personalized Historical Context. *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems* (2023). https://api.semanticscholar.org/CorpusID:256868353

[16] Souti Chattopadhyay, I. V. R. K. V. Prasad, Austin Z. Henley, Anita Sarma, and Titus Barik. 2020. What's Wrong with Computational Notebooks? Pain Points, Needs, and Design Opportunities. *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems* (2020). https://api.semanticscholar.org/CorpusID:210927488

[17] Jiangjie Chen, Siyu Yuan, Rong Ye, Bodhisattwa Prasad Majumder, and Kyle Richardson. 2023. Put your money where your mouth is: Evaluating strategic planning and execution of llm agents in an auction arena. *arXiv preprint arXiv:2310.05746* (2023).

[18] Inyoung Cheong, King Xia, K. J. Kevin Feng, Quan Ze Chen, and Amy X. Zhang. 2024. (A)I Am Not a Lawyer, But...: Engaging Legal Experts towards Responsible LLM Policies for Legal Advice. In *Conference on Fairness, Accountability and Transparency*. https://api.semanticscholar.org/CorpusID:267413187

[19] Frederick Choi, Sajjadur Rahman, Han Jun Kim, and Daz Zhang. 2023. Towards Transparent, Reusable, and Customizable Data Science in Computational Notebooks. *Extended Abstracts of the 2023 CHI Conference on Human Factors in Computing Systems* (2023). https://api.semanticscholar.org/CorpusID:257687372

[20] Katherine M Collins, Ilia Sucholutsky, Umang Bhatt, Kartik Chandra, Lionel Wong, Mina Lee, Cedegao E Zhang, Tan Zhi-Xuan, Mark Ho, Vikash Mansinghka, et al. 2024. Building machines that learn and think with people. *arXiv preprint arXiv:2408.03943* (2024).

[21] Douglas L. Dean, Jillian M. Hender, Thomas Lee Rodgers, and Eric L. Santanen. 2006. Identifying Quality, Novel, and Creative Ideas: Constructs and Scales for Idea Evaluation. *J. Assoc. Inf. Syst.* 7 (2006), 30. https://api.semanticscholar.org/CorpusID:15910404

[22] K. J. Kevin Feng, Quan Ze Chen, Inyoung Cheong, King Xia, and Amy X. Zhang. 2023. Case Repositories: Towards Case-Based Reasoning for AI Alignment. *ArXiv* abs/2311.10934 (2023). https://api.semanticscholar.org/CorpusID:265295304

[23] K. J. Kevin Feng, Inyoung Cheong, Quan Ze Chen, and Amy X. Zhang. 2024. Policy Prototyping for LLMs: Pluralistic Alignment via Interactive and Collaborative Policymaking. *ArXiv* abs/2409.08622 (2024). https://api.semanticscholar.org/CorpusID:272654085

[24] Jennifer Fereday and Eimear Muir-Cochrane. 2006. Demonstrating rigor using thematic analysis: A hybrid approach of inductive and deductive coding and theme development. *International journal of qualitative methods* 5, 1 (2006), 80–92.

[25] Raymond Fok, Joseph Chee Chang, Tal August, Amy X. Zhang, and Daniel S. Weld. 2023. Qlarify: Recursively Expandable Abstracts for Directed Information Retrieval over Scientific Papers. https://api.semanticscholar.org/CorpusID:263835343

[26] Raymond Fok, Hita Kambhamettu, Luca Soldaini, Jonathan Bragg, Kyle Lo, Andrew Head, Marti A. Hearst, and Daniel S. Weld. 2022. Scim: Intelligent Skimming Support for Scientific Papers. *Proceedings of the 28th International Conference on Intelligent User Interfaces* (2022). https://api.semanticscholar.org/CorpusID:254591867

[27] Google. 2025. Gemini Deep Research. https://gemini.google/overview/deep-research/.

[28] Juraj Gottweis, Wei-Hung Weng, Alexander Daryin, Tao Tu, Anil Palepu, Petar Sirkovic, Artiom Myaskovsky, Felix Weissenberger, Keran Rong, Ryutaro Tanno, et al. 2025. Towards an AI co-scientist. *arXiv preprint arXiv:2502.18864* (2025).

[29] Madeleine Grunde-McLaughlin, Michelle S. Lam, Ranjay Krishna, Daniel S. Weld, and Jeffrey Heer. 2023. Designing LLM Chains by Adapting Techniques from Crowdsourcing Workflows. *ArXiv* abs/2312.11681 (2023). https://api.semanticscholar.org/CorpusID:266362444

[30] Joel Grus. 2018. I don't like notebooks. https://docs.google.com/presentation/d/1n2RlMdmv1p25Xy5thJUhkKGvjtV-dkAIsUXP-AL4ffI/edit#slide=id.g362da58057_0_1.

[31] Xuemei Gu and Mario Krenn. 2024. Generation and human-expert evaluation of interesting research ideas using knowledge graphs and large language models. https://api.semanticscholar.org/CorpusID:270062620

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

[33] Shibo Hao, Yi Gu, Haodi Ma, Joshua Jiahua Hong, Zhen Wang, Daisy Zhe Wang, and Zhiting Hu. 2023. Reasoning with Language Model is Planning with World Model. *ArXiv* abs/2305.14992 (2023). https://api.semanticscholar.org/CorpusID:258865812

[34] Tom Hope, Doug Downey, Oren Etzioni, Daniel S. Weld, and Eric Horvitz. 2022. A Computational Inflection for Scientific Discovery. *Commun. ACM* 66 (2022), 62 – 73. https://api.semanticscholar.org/CorpusID:248512482

[35] Eric Horvitz. 1999. Principles of mixed-initiative user interfaces. In *International Conference on Human Factors in Computing Systems*. https://api.semanticscholar.org/CorpusID:8943607

[36] Hamel Husain, Isaac Flath, and Johno Whitaker. 2025. Thoughts On A Month With Devin. https://www.answer.ai/posts/2025-01-08-devin.html.

[37] Edwin L. Hutchins, James Hollan, and Donald A. Norman. 1985. Direct Manipulation Interfaces. *Hum. Comput. Interact.* 1 (1985), 311–338. https://api.semanticscholar.org/CorpusID:16355120

[38] Chip Huyen. 2025. Agents. https://huyenchip.com/2025/01/07/agents.html.

[39] Lujain Ibrahim, Saffron Huang, Lama Ahmad, and Markus Anderljung. 2024. Beyond static AI evaluations: advancing human interaction evaluations for LLM harms and risks. *ArXiv* abs/2405.10632 (2024). https://api.semanticscholar.org/CorpusID:269899912

[40] Peter Jansen, Marc-Alexandre Côté, Tushar Khot, Erin Bransom, Bhavana Dalvi Mishra, Bodhisattwa Prasad Majumder, Oyvind Tafjord, and Peter Clark. 2024. DISCOVERYWORLD: A Virtual Environment for Developing and Evaluating Automated Scientific Discovery Agents. *arXiv preprint arXiv:2406.06769* (2024).

[41] Peter Jansen, Oyvind Tafjord, Marissa Radensky, Pao Siangliulue, Tom Hope, Bhavana Dalvi, Bodhisattwa Prasad Majumder, Daniel S. Weld, and Peter Clark. 2025. CodeScientist: End-to-End Semi-Automated Scientific Discovery with Code-based Experimentation. https://api.semanticscholar.org/CorpusID:277451644

[42] Carlos E. Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. 2023. SWE-bench: Can Language Models Resolve Real-World GitHub Issues? *ArXiv* abs/2310.06770 (2023). https://api.semanticscholar.org/CorpusID:263829697

[43] Marina Jirotka, Charlotte P Lee, and Gary M Olson. 2013. Supporting scientific collaboration: Methods, tools and concepts. *Computer Supported Cooperative Work (CSCW)* 22 (2013), 667–715.

[44] Project Jupyter. 2015. Jupyter Notebook UX Survey. https://github.com/jupyter/surveys/tree/master/surveys/2015-12-notebook-ux.

[45] Hyeonsu B Kang, Joseph Chee Chang, Yongsung Kim, and Aniket Kittur. 2022. Threddy: An Interactive System for Personalized Thread-based Exploration and Organization of Scientific Literature. *Proceedings of the 35th Annual ACM Symposium on User Interface Software and Technology* (2022). https://api.semanticscholar.org/CorpusID:251402552

[46] Hyeonsu B Kang, Rafal Kocielnik, Andrew Head, Jiangjiang Yang, Matt Latzke, Aniket Kittur, Daniel S. Weld, Doug Downey, and Jonathan Bragg. 2022. From Who You Know to What You Read: Augmenting Scientific Recommendations with Implicit Social Networks. *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems* (2022). https://api.semanticscholar.org/CorpusID:248299830

[47] Hyeonsu B Kang, Xin Qian, Tom Hope, Dafna Shahaf, Joel Chan, and Aniket Kittur. 2022. Augmenting Scientific Creativity with an Analogical Search Engine. *ACM Transactions on Computer-Human Interaction* 29 (2022), 1 – 36. https://api.semanticscholar.org/CorpusID:249209576

[48] Hyeonsu B Kang, Sherry Wu, Joseph Chee Chang, and Aniket Kittur. 2023. Synergi: A Mixed-Initiative System for Scholarly Synthesis and Sensemaking. *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology* (2023). https://api.semanticscholar.org/CorpusID:260899915

[49] Sayash Kapoor, Benedikt Stroebl, Zachary S. Siegel, Nitya Nadgir, and Arvind Narayanan. 2024. AI Agents That Matter. *ArXiv* abs/2407.01502 (2024). https://api.semanticscholar.org/CorpusID:270870360

[50] Majeed Kazemitabaar, Jack Williams, Ian Drosos, Tovi Grossman, Austin Z. Henley, Carina Negreanu, and Advait Sarkar. 2024. Improving Steering and Verification in AI-Assisted Data Analysis with Interactive Task Decomposition. https://api.semanticscholar.org/CorpusID:270923956

[51] Mary Beth Kery, Marissa Radensky, Mahima Arya, Bonnie E. John, and Brad A. Myers. 2018. The Story in the Notebook: Exploratory Data Science using a Literate Programming Tool. *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* (2018). https://api.semanticscholar.org/CorpusID:5060661

[52] Mary Beth Kery, Donghao Ren, Fred Hohman, Dominik Moritz, Kanit Wongsuphasawat, and Kayur Patel. 2020. mage: Fluid Moves Between Code and

Graphical Work in Computational Notebooks. *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology* (2020). https://api.semanticscholar.org/CorpusID:221836345

[53] Joongwon Kim, Bhargavi Paranjape, Tushar Khot, and Hanna Hajishirzi. 2024. Husky: A Unified, Open-Source Language Agent for Multi-Step Reasoning. https://api.semanticscholar.org/CorpusID:270370824

[54] Jeongyeon Kim, Sangho Suh, Lydia B Chilton, and Haijun Xia. 2023. Metaphorian: Leveraging Large Language Models to Support Extended Metaphor Creation for Science Writing. In *Proceedings of the 2023 ACM Designing Interactive Systems Conference*. 115–135.

[55] Donald Ervin Knuth. 1984. Literate Programming. In *Computer/law journal*. https://api.semanticscholar.org/CorpusID:1200693

[56] Jan Kulveit, Raymond Douglas, Nora Ammann, Deger Turan, David Krueger, and David Duvenaud. 2025. Gradual Disempowerment: Systemic Existential Risks from Incremental AI Development. *arXiv preprint arXiv:2501.16946* (2025).

[57] Sam Lau, Ian Drosos, Julia M. Markel, and Philip J. Guo. 2020. The Design Space of Computational Notebooks: An Analysis of 60 Systems in Academia and Industry. In *2020 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*. 1–11. https://doi.org/10.1109/VL/HCC50065.2020.9127201

[58] Lane Lawley and Christopher J. MacLellan. 2024. VAL: Interactive Task Learning with GPT Dialog Parsing. *Proceedings of the CHI Conference on Human Factors in Computing Systems* (2024). https://api.semanticscholar.org/CorpusID:263609076

[59] Charlotte P Lee. 2007. Boundary negotiating artifacts: Unbinding the routine of boundary objects and embracing chaos in collaborative work. *Computer Supported Cooperative Work (CSCW)* 16 (2007), 307–339.

[60] Mina Lee, Katy Ilonka Gero, John Joon Young Chung, Simon Buckingham Shum, Vipul Raheja, Hua Shen, Subhashini Venugopalan, Thiemo Wambsganss, David Zhou, Emad A. Alghamdi, Tal August, Avinash Bhat, Madiha Zahrah Choksi, Senjuti Dutta, Jin L.C. Guo, Md. Naimul Hoque, Yewon Kim, Seyed Parsa Neshaei, Agnia Sergeyuk, Antonette Shibani, Disha Shrivastava, Lila Shroff, Jessi Stark, S. Sterman, Sitong Wang, Antoine Bosselut, Daniel Buschek, Joseph Chee Chang, Sherol Chen, Max Kreminski, Joonsuk Park, Roy Pea, Eugenia H. Rho, Shannon Zejiang Shen, and Pao Siangliulue. 2024. A Design Space for Intelligent and Interactive Writing Assistants. *Proceedings of the CHI Conference on Human Factors in Computing Systems* (2024). https://api.semanticscholar.org/CorpusID:268553985

[61] Yoonjoo Lee, Hyeonsu B Kang, Matt Latzke, Juho Kim, Jonathan Bragg, Joseph Chee Chang, and Pao Siangliulue. 2024. PaperWeaver: Enriching Topical Paper Alerts by Contextualizing Recommended Papers with User-collected Papers. In *International Conference on Human Factors in Computing Systems*. https://api.semanticscholar.org/CorpusID:268248445

[62] Zhehui Liao, Maria Antoniak, Inyoung Cheong, Evie Yu-Yen Cheng, Ai-Heng Lee, Kyle Lo, Joseph Chee Chang, and Amy X Zhang. 2024. LLMs as Research Tools: A Large Scale Survey of Researchers' Usage and Perceptions. *arXiv preprint arXiv:2411.05025* (2024).

[63] Henry Lieberman. 1997. Autonomous interface agents. *Proceedings of the ACM SIGCHI Conference on Human factors in computing systems* (1997). https://api.semanticscholar.org/CorpusID:6576547

[64] Yanna Lin, Haotian Li, Leni Yang, Aoyu Wu, and Huamin Qu. 2023. InkSight: Leveraging Sketch Interaction for Documenting Chart Findings in Computational Notebooks. *IEEE Transactions on Visualization and Computer Graphics* 30 (2023), 944–954. https://api.semanticscholar.org/CorpusID:259936935

[65] Yiren Liu, Si Chen, Haocong Cheng, Mengxia Yu, Xiao Ran, Andrew Mo, Yiliu Tang, and Yun Huang. 2024. How AI Processing Delays Foster Creativity: Exploring Research Question Co-Creation with an LLM-based Agent. In *International Conference on Human Factors in Computing Systems*. https://api.semanticscholar.org/CorpusID:269748457

[66] Yue Liu, Sin Kit Lo, Qinghua Lu, Liming Zhu, Dehai Zhao, Xiwei Xu, Stefan Harrer, and Jon Whittle. 2024. Agent Design Pattern Catalogue: A Collection of Architectural Patterns for Foundation Model based Agents. *arXiv preprint arXiv:2405.10467* (2024).

[67] Chris Lu, Cong Lu, Robert Tjarko Lange, Jakob Foerster, Jeff Clune, and David Ha. 2024. The ai scientist: Towards fully automated open-ended scientific discovery. *arXiv preprint arXiv:2408.06292* (2024).

[68] Xiao Ma, Swaroop Mishra, Ariel Liu, Sophie Ying Su, Jilin Chen, Chinmay Kulkarni, Heng-Tze Cheng, Quoc V. Le, and Ed Huai hsin Chi. 2023. Beyond ChatBots: ExploreLLM for Structured Thoughts and Personalized Model Responses. *Extended Abstracts of the CHI Conference on Human Factors in Computing Systems* (2023). https://api.semanticscholar.org/CorpusID:265551437

[69] Pattie Maes. 1994. Agents that reduce work and information overload. *Commun. ACM* 37 (1994), 30–40. https://api.semanticscholar.org/CorpusID:59868493

[70] Pattie Maes and Alan Wexelblat. 1996. Interface agents. *Conference Companion on Human Factors in Computing Systems* (1996). https://api.semanticscholar.org/CorpusID:26827189

[71] Bodhisattwa Prasad Majumder, Harshit Surana, Dhruv Agarwal, Sanchaita Hazra, Ashish Sabharwal, and Peter Clark. 2024. Data-driven Discovery with Large Generative Models. *ArXiv* abs/2402.13610 (2024). https://api.semanticscholar.org/CorpusID:267770682

[72] Hussein Mozannar and David Sontag. 2020. Consistent estimators for learning to defer to an expert. In *International conference on machine learning*. PMLR, 7076–7087.

[73] Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. 2025. s1: Simple test-time scaling. *arXiv preprint arXiv:2501.19393* (2025).

[74] Harshit Nigam, Manasi S. Patwardhan, Lovekesh Vig, and Gautam M. Shroff. 2024. Acceleron: A Tool to Accelerate Research Ideation. *ArXiv* abs/2403.04382 (2024). https://api.semanticscholar.org/CorpusID:268264612

[75] Jasper Tran O'Leary, Gabrielle Benabdallah, and Nadya Peek. 2023. Imprimer: Computational Notebooks for CNC Milling. *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems* (2023). https://api.semanticscholar.org/CorpusID:258217042

[76] Gary M Olson and Judith S Olson. 2000. Distance matters. *Human–computer interaction* 15, 2-3 (2000), 139–178.

[77] OpenAI. 2024. Introducing canvas. https://openai.com/index/introducing-canvas/.

[78] OpenAI. 2024. Learning to Reason with LLMs. https://openai.com/index/learning-to-reason-with-llms/.

[79] OpenAI. 2025. Introducing deep research. https://openai.com/index/introducing-deep-research/.

[80] OpenAI. 2025. Introducing Operator. https://openai.com/index/introducing-operator/.

[81] OpenAI Platform. 2024. Assistants API. https://platform.openai.com/docs/assistants/overview.

[82] James C Overholser. 1993. Elements of the Socratic method: I. Systematic questioning. *Psychotherapy: Theory, Research, Practice, Training* 30, 1 (1993), 67.

[83] Christine A Padesky. 1993. Socratic questioning: Changing minds or guiding discovery. In *A keynote address delivered at the European Congress of Behavioural and Cognitive Therapies, London*, Vol. 24.

[84] Srishti Palani, Aakanksha Naik, Doug Downey, Amy X. Zhang, Jonathan Bragg, and Joseph Chee Chang. 2023. Relatedly: Scaffolding Literature Reviews with Existing Related Work Sections. *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems* (2023). https://api.semanticscholar.org/CorpusID:256846632

[85] Perplexity. 2024. Perplexity AI. https://www.perplexity.ai/.

[86] Perplexity. 2025. Introducing Perplexity Deep Research. https://www.perplexity.ai/hub/blog/introducing-perplexity-deep-research.

[87] Kevin Pu, K. J. Kevin Feng, Tovi Grossman, Tom Hope, Bhavana Dalvi, Matt Latzke, Jonathan Bragg, Joseph Chee Chang, and Pao Siangliulue. 2024. IdeaSynth: Iterative Research Idea Development Through Evolving and Composing Idea Facets with Literature-Grounded Feedback. *ArXiv* abs/2410.04025 (2024). https://api.semanticscholar.org/CorpusID:273186404

[88] Yujia Qin, Shi Liang, Yining Ye, Kunlun Zhu, Lan Yan, Ya-Ting Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, Sihan Zhao, Runchu Tian, Ruobing Xie, Jie Zhou, Marc H. Gerstein, Dahai Li, Zhiyuan Liu, and Maosong Sun. 2023. ToolLLM: Facilitating Large Language Models to Master 16000+ Real-world APIs. *ArXiv* abs/2307.16789 (2023). https://api.semanticscholar.org/CorpusID:260334759

[89] Napol Rachatasumrit, Jonathan Bragg, Amy X. Zhang, and Daniel S. Weld. 2022. CiteRead: Integrating Localized Citation Contexts into Scientific Paper Reading. *27th International Conference on Intelligent User Interfaces* (2022). https://api.semanticscholar.org/CorpusID:247585131

[90] Reworkd. 2024. AgentGPT. https://agentgpt.reworkd.ai/.

[91] Adam Rule, Aurélien Tabard, and James Hollan. 2018. Exploration and Explanation in Computational Notebooks. *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* (2018). https://api.semanticscholar.org/CorpusID:5048947

[92] Arvind Satyanarayan. 2024. Intelligence as Agency. In *Adjunct Proceedings of the 37th Annual ACM Symposium on User Interface Software and Technology*. 1–3.

[93] Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. Toolformer: Language Models Can Teach Themselves to Use Tools. *ArXiv* abs/2302.04761 (2023). https://api.semanticscholar.org/CorpusID:256697342

[94] Kjeld Schmidt and Ina Wagner. 2002. Coordinative artifacts in architectural practice. In *COOP*, Vol. 2. Citeseer, 257–274.

[95] Yijia Shao, Vinay Samuel, Yucheng Jiang, John Yang, and Diyi Yang. 2024. Collaborative Gym: A Framework for Enabling and Evaluating Human-Agent Collaboration. https://api.semanticscholar.org/CorpusID:274964990

[96] Quan Shi, Michael Tang, Karthik Narasimhan, and Shunyu Yao. 2024. Can Language Models Solve Olympiad Programming? *ArXiv* abs/2404.10952 (2024). https://api.semanticscholar.org/CorpusID:269187896

[97] Ben Shneiderman and Pattie Maes. 1997. Direct manipulation vs. interface agents. *Interactions* 4 (1997), 42–61. https://api.semanticscholar.org/CorpusID:27708923

[98] Wesley Shrum, Joel Genuth, and Ivan Chompalov. 2007. *Structures of scientific collaboration*. MIT Press.

[99] Susan Leigh Star and James R Griesemer. 1989. Institutional ecology,translations' and boundary objects: Amateurs and professionals in Berkeley's Museum of Vertebrate Zoology, 1907-39. *Social studies of science* 19, 3 (1989), 387–420.

[100] Sangho Suh, Bryan Min, Srishti Palani, and Haijun Xia. 2023. Sensecape: Enabling Multilevel Exploration and Sensemaking with Large Language Models. *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology* (2023). https://api.semanticscholar.org/CorpusID:258822925

[101] Theodore R. Sumers, Shunyu Yao, Karthik Narasimhan, and Thomas L. Griffiths. 2023. Cognitive Architectures for Language Agents. *ArXiv* abs/2309.02427 (2023). https://api.semanticscholar.org/CorpusID:261556862

[102] Michael Terry, Chinmay Kulkarni, Martin Wattenberg, Lucas Dixon, and Meredith Ringel Morris. 2023. Interactive AI Alignment: Specification, Process, and Evaluation Alignment. https://api.semanticscholar.org/CorpusID:264935292

[103] Karthik Valmeekam, Matthew Marquez, Sarath Sreedharan, and Subbarao Kambhampati. 2023. On the Planning Abilities of Large Language Models - A Critical Investigation. *ArXiv* abs/2305.15771 (2023). https://api.semanticscholar.org/CorpusID:260440590

[104] Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi (Jim) Fan, and Anima Anandkumar. 2023. Voyager: An Open-Ended Embodied Agent with Large Language Models. *ArXiv* abs/2305.16291 (2023). https://api.semanticscholar.org/CorpusID:258887849

[105] Ruotong Wang, Xinyi Zhou, Lin Qiu, Joseph Chee Chang, Jonathan Bragg, and Amy X Zhang. 2024. Social-RAG: Retrieving from Group Interactions to Socially Ground Proactive AI Generation to Group Preferences. *arXiv preprint arXiv:2411.02353* (2024).

[106] Xingbo Wang, Samantha Lee Huey, Rui Sheng, Saurabh Mehta, and Fei Wang. 2024. SciDaSynth: Interactive Structured Knowledge Extraction and Synthesis from Scientific Literature with Large Language Model. *ArXiv* abs/2404.13765 (2024). https://api.semanticscholar.org/CorpusID:269293213

[107] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed Huai hsin Chi, F. Xia, Quoc Le, and Denny Zhou. 2022. Chain of Thought Prompting Elicits Reasoning in Large Language Models. *ArXiv* abs/2201.11903 (2022). https://api.semanticscholar.org/CorpusID:246411621

[108] Scott Wu. 2024. Introducing Devin, the first AI software engineer. https://www.cognition.ai/blog/introducing-devin.

[109] Tongshuang Sherry Wu, Michael Terry, and Carrie J. Cai. 2022. AI Chains: Transparent and Controllable Human-AI Interaction by Chaining Large Language Model Prompts. *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems* (2022). https://api.semanticscholar.org/CorpusID:238353829

[110] Tongshuang Sherry Wu, Haiyi Zhu, Maya Albayrak, Alexis Axon, Amanda Bertsch, Wenxing Deng, Ziqi Ding, Bill Boyuan Guo, Sireesh Gururaja, Tzu-Sheng Kuo, Jenny T Liang, Ryan Liu, Ihita Mandal, Jeremiah Milbauer, Xiaolin Ni, N. Padmanabhan, Subhashini Ramkumar, Alexis Sudjianto, Jordan Taylor, Ying-Jui Tseng, Patricia Vaidos, Zhijin Wu, Wei Wu, and Chenyang Yang. 2023. LLMs as Workers in Human-Computational Algorithms? Replicating Crowdsourcing Pipelines with LLMs. *ArXiv* abs/2307.10168 (2023). https://api.semanticscholar.org/CorpusID:259982473

[111] John Yang, Carlos E. Jimenez, Alexander Wettig, Kilian Lieret, Shunyu Yao, Karthik Narasimhan, and Ofir Press. 2024. SWE-agent: Agent-Computer Interfaces Enable Automated Software Engineering. *ArXiv* abs/2405.15793 (2024). https://api.semanticscholar.org/CorpusID:270063685

[112] Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. 2022. Webshop: Towards scalable real-world web interaction with grounded language agents. *Advances in Neural Information Processing Systems* 35 (2022), 20744–20757.

[113] Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. Tree of Thoughts: Deliberate Problem Solving with Large Language Models. *ArXiv* abs/2305.10601 (2023). https://api.semanticscholar.org/CorpusID:258762525

[114] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2022. ReAct: Synergizing Reasoning and Acting in Language Models. *ArXiv* abs/2210.03629 (2022). https://api.semanticscholar.org/CorpusID:252762395

[115] Xingjian Zhang, Yutong Xie, Jin Huang, Jinge Ma, Zhaoying Pan, Qijia Liu, Ziyang Xiong, Tolga Ergen, Dongsub Shim, Honglak Lee, and Qiaozhu Mei. 2024. MASSW: A New Dataset and Benchmark Tasks for AI-Assisted Scientific Workflows. https://api.semanticscholar.org/CorpusID:270370776

[116] Zheng Zhang, Jie Gao, Ranjodh Singh Dhaliwal, and Toby Jia-Jun Li. 2023. Visar: A human-ai argumentative writing assistant with visual programming and rapid draft prototyping. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*. 1–30.

[117] Chengbo Zheng, Dakuo Wang, April Yi Wang, and Xiaojuan Ma. 2022. Telling Stories from Computational Notebooks: AI-Assisted Presentation Slides Creation for Presenting Data Science Work. *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems* (2022). https://api.semanticscholar.org/CorpusID:247594488

[118] Andy Zhou, Kai Yan, Michal Shlapentokh-Rothman, Haohan Wang, and Yu-Xiong Wang. 2023. Language Agent Tree Search Unifies Reasoning Acting and Planning in Language Models. *ArXiv* abs/2310.04406 (2023). https://api.semanticscholar.org/CorpusID:263829963

[119] Yuchen Zhuang, Xiang Chen, Tong Yu, Saayan Mitra, Victor S. Bursztyn, Ryan A. Rossi, Somdeb Sarkhel, and Chao Zhang. 2023. ToolChain*: Efficient Action Space Navigation in Large Language Models with A* Search. *ArXiv* abs/2310.13227 (2023). https://api.semanticscholar.org/CorpusID:264405734

A FREQUENTLY ASKED QUESTIONS

A.1 What's the novelty of Cocoa compared to "deep research" systems?

It's true that Cocoa and deep research systems can be used for similar tasks (e.g., literature review) and might even obtain similar results. However, deep research systems (at least at the time of writing) use fully automated AI workflows and do not accommodate user steering during the research process. Deep research systems may engage in some back and forth with users *before* the research process to clarify requests, or allow the user to follow up *after* the process ends, but does not afford user interaction *during* the process itself.

Why might this be undesirable? Let's say you want to write a literature review on *whether buttons with icons and labels are more usable than buttons without labels, or labels without icons*. This is a real example that OpenAI has on their Deep Research release announcement webpage [79]. As an HCI researcher, you know of some reputable venues and authors you can rely on, but the agent used sources you might view as less credible (see sources in Deep Research's response in the UX Design tab of [79]). If Deep Research offered the affordances, you could guide the agent with your expertise during the research process to steer it towards using your trusted sources, which can also help improve the quality of its output. Without these affordances, you could follow up via prompting afterwards, but 1) there's no guarantee that the agent will reliably follow those new instructions, and 2) re-running the research process is time- and resource-intensive. That's why we draw on the computational notebook metaphor in Cocoa: interactive plans allow you to iteratively hone and refine the agent's trajectory and outputs to bolster steerability, control, efficiency, and quality.

A.2 Can a user technically co-plan and co-execute with an agent via chat?

Technically, chat interfaces allow the user to ask the agent for a plan and prompt the agent to regenerate the plan before executing it (which has some aspects of co-planning). The user can also direct the agent to save some steps for themselves and prompt the agent to modify its outputs for steps assigned to the agent (which has some aspects of co-execution). However, these interactions are quite cumbersome in chat and not aligned with users' mental models of chat interfaces—we did not witness any participants using our baseline system this way. Interactive plans are a fundamentally distinct design pattern designed to support interleaved co-planning and co-execution, and elicits different user behaviors than chat interfaces (see Figures 12 and 13 for evidence of this).

A.3 Do you see Cocoa's lack of improvement over the baseline for general utility as a negative result?

No, we do not. General utility is a combination of three measures we obtained by asking participants the following Likert scale questions: 1) I've developed a better understanding of potential solutions to this particular problem, 2) I obtained new and useful insights from using system, and 3) The system's outputs were useful for exploring the problem. It is unsurprising that there was no significant difference in answers to these questions, because these aspects are often bottlenecked by *the performance of the underlying agent*, and we used the same agent in both Cocoa and the baseline. However, we can improve aspects such as steerability with interface-level advances, which we have shown in this work. If general utility had gone up in Cocoa, it would have been a pleasant surprise, but it is understandable if it did not.

A.4 Why is it important to get the user involved? Can't we just let AI automate everything?

While existing efforts to make agents operate as autonomously as possible without seeking user input may be a compelling research agenda, this approach is not always practical. First, there some tasks humans can more effectively complete than AI, especially if the task requires context or domain expertise the LLM does not have access to. Second, there are some tasks humans may prefer to perform rather than having AI automate it (e.g., if doing the task allows the user to gain new insights or knowledge that they may not obtain otherwise). Third, having an agent operate for a long period of time without adequate steering mechanisms for users can both be resource-inefficient from a cost perspective and unsettling from a user experience perspective.

The list can go on, but the point here is that agents will be working with and operated by humans in the real world, so we need to think more deeply about how humans and agents can collaborate effectively. Our work is one product of such thinking.

A.5 Why didn't you use ChatGPT or a similar publicly available AI chatbot for your baseline?

Our baseline's interface closely matches that of ChatGPT and other AI chatbots, but the underlying system is different. We use an LLM agent powered by GPT-4o, but specifically trained it to use the Semantic Scholar API as tools to reliably perform research-related tasks such as searches for papers/authors/topics, sorting papers by various metrics including citation count, summarizing papers, comparing papers, retrieving prominent papers by a particular author, etc. See Section 4.4 for technical details. However, our main contribution is not building a new LLM agent nor advancing agent capabilities—rather, it is to explore cost and benefit tradeoffs between interaction techniques for humans to work with AI agents. We use the same underlying agent in Cocoa, so we kept this factor constant across our system and baseline.

A.6 I'm a developer who builds agentic AI applications. What are some practical takeaways your work has for me?

Our Discussion and Future Work section is where you'd find most of our practical takeaways, so we'd encourage you to read that. In addition, here are two high-level takeaways to consider:

- Users may not want agents to automate every step of workflows for many reasons, whether it's to have more ownership over their work, or feeling that the agent is incompetent at

performing a particular task. Strategic and configurable automation is key to building great user experiences for agents.
- Agents can be prohibitively expensive to run, and this may be one of the major bottlenecks for widespread adoption. Because agents will inevitably work with humans in real-world settings, it's important to identify opportunities for human input to lower cost AND improve system performance.

A.7 Can I use Cocoa and/or the baseline?

We are continuing to iterate on Cocoa and the baseline, so we are not yet planning a public release. However, we encourage you to check out Ai2's ScholarQA[12] and PaperFinder[13] tools, which share some similar features.

B FORMATIVE STUDY PARTICIPANTS

See Table 1

C COCOA FORMATIVE STUDY PROCEDURE DETAILS

We provide more details of each portion of our formative study and its associated activities below.

C.1 Project document activity

First, we asked participants to discuss some of the ideas in their project document they shared with us, the origins of those ideas, and the ideal ways they envision AI-powered tools helping them throughout the research process. We then invited participants to a Google Doc or Slides that we converted from their project document for easy collaboration; this conversion was made by copying and pasting content from their project document if it was not already in Docs or Slides. We asked participants to brainstorm 2–3 remaining questions they have about their project that they would be interested in tackling. For each question, participants created a bulleted plan they would undertake to pursue it, which we then used to answer **F2**.

C.2 Probe activity

In the second activity, we presented participants with a WoZ design probe in the form of two side-by-side Google Docs. One Doc, the "planning document," contained 3 research ideas generated by Perplexity AI [85] based on the description of research interests submitted by the participant and 1–3 of their most recent publications and/or preprints. Participants were asked to work in the doc to expand upon an idea (or combine multiple) through a self-defined plan, with the option of invoking the help of an AI assistant through requests prefaced with a "!" command. The study facilitator (the Wizard) entered this command into Perplexity AI and pasted the output to the other Doc, the "AI output document." While waiting for the response, we asked participants to write a brief plan consisting of 3–6 steps for how they would complete the request themselves. Participants then inspected the assistant's output and incorporated any useful text into their document. This process repeated until the participant felt like their idea was sufficiently concrete to write a short descriptive paragraph about it, typically after at least 2–3 requests to the AI. This activity helped us answer **F3**.

C.3 Concluding interview

The study ended with a concluding interview, where we asked participants about their experiences, perceptions, and desiderata for the AI assistant, along with preferred ways to interact with it. Each participant was given a $35 USD honorarium after the study. The study was reviewed and approved by our organization's internal IRB.

D FORMATIVE STUDY DATA ANALYSIS

To answer **F1**, the first author used a hybrid inductive-deductive coding process [24] to code participants' submitted project documents. The first 5 documents were inductively coded to surface common structural elements before the elements were deductively applied to the remaining 4. We iterated on existing elements and added new ones as needed.

For **F2**, we performed inductive thematic analysis on two documents—participants' submitted project documents and in-study planning documents. The first author sourced *intentions for planning* within submitted project documents by locating the snippet of text that initiates a plan. For example, if the plan consists of a bulleted list, the intention is often expressed in the lead-in text that immediately precedes the list. We note that an individual plan item can also signal planning intent if it contains a nested plan. The first author performed thematic analysis by inductively coding the extracted text. The first author then identified and extracted *plan steps* participants wrote in both documents before inductively coding them.

For **F3**, the first author performed open coding on the study transcripts before thematically analyzing the coded snippets. To enrich the data, the first author also extracted and inductively coded all requests to the assistant from participants' planning documents. The codes were discussed and iterated upon with other project team members on a weekly basis over the course of 3 weeks.

E LAB STUDY PARTICIPANTS

See Table 2

F LAB STUDY RECRUITMENT PROCESS AND PILOT STUDIES

We sent out a recruitment form to collect basic demographic details, frequency of AI use in research, and a copy of a project document from an ongoing or completed project document to use during the study. One participant also participated in our formative study. 15 researchers were based in the United States, and 1 was based in Canada. Participants' research areas ranged from large language models, to ubiquitous computing, to visualization; broadly, they spanned HCI (n = 10), ML (n = 3), and NLP (n = 3). In our participant pool, many users were occasional (6) or frequent[14] (6) users of AI tools in the research process.

[12] https://scholarqa.allen.ai/chat
[13] https://paperfinder.allen.ai/chat

[14] We define "occasional" and "frequent" the same way as we do in our formative study (Table 1).

P#	Gender	Age Range	Country	Research YoE	Research Area	Research Area (General)	AI Use Frequency
P1	Man	25–34	U.S.	6–10	Human-AI interaction	HCI	A couple times
P2	Man	25–34	U.S.	6–10	LLM evaluation	NLP	Occasionally
P3	Woman	18–24	U.S.	2–5	LLMs + society	NLP	Never
P4	Man	25–34	Canada	2–5	Multilingual NLP	NLP	Occasionally
P5	Woman	25–34	U.S.	6–10	Human-AI interaction	HCI	Occasionally
P6	Man	25–34	U.S.	2–5	Multimodal AI & HCI	HCI	Occasionally
P7	Woman	25–34	South Korea	2–5	LLM retrieval	NLP	Occasionally
P8	Man	25–34	U.S.	6–10	Human-AI interaction	NLP	Occasionally
P9	Woman	18–24	U.S.	0–1	Creativity support tools	HCI	A couple times

Table 1: Participants from our formative study. All participants were Ph.D. students. Country refers to the country in which the participant primarily conducts research at the time of the study. Research YoE refers to the years of experience conducting academic research. AI Use Frequency refers to how frequently the participants uses AI tools to ideate and/or iterate on research ideas. Participants selected "Occasionally" based on the description "I don't rely on AI but sometimes tinker with it."

P#	Gender	Age Range	Country	Job Title	Research YoE	Research Area	Research Area (General)	AI Use Frequency
P1	Woman	25–34	U.S.	Ph.D. student	2–5	Social computing	HCI	Frequently
P2	Woman	25–34	U.S.	Ph.D. student	2–5	Social computing	HCI	Occasionally
P3	Woman	25–34	U.S.	Ph.D. student	6–10	Visualization	HCI	Occasionally
P4	Woman	25–34	U.S.	Ph.D. student	2–5	Culture + computing	HCI	A couple times
P5	Woman	25–34	U.S.	Postdoc	6–10	Software development	HCI	Occasionally
P6	Woman	25–34	U.S.	Ph.D. student	6–10	Health + computing	HCI	Occasionally
P7	Man	25–34	U.S.	Ph.D. student	2–5	LLMs	NLP	Frequently
P8	Woman	25–34	U.S.	Ph.D. student	2–5	Accessibility	HCI	Never
P9	Man	25–34	U.S.	Ph.D. student	2–5	Multimodal AI	ML	Frequently
P10	Woman	18–24	U.S.	Ph.D. student	2–5	On-device AI	ML	A couple times
P11	Man	18–24	U.S.	Ph.D. student	2–5	Ubiquitous computing	HCI	Always
P12	Man	25–34	U.S.	Ph.D. student	6–10	LLM evaluation	NLP	Frequently
P13	Man	35–44	Canada	Ph.D. student	2–5	Computational biology	ML	Frequently
P14	Woman	18–24	U.S.	Ph.D. student	2–5	LLMs	NLP	Frequently
P15	Woman	25–34	U.S.	Ph.D. student	2–5	Social computing	HCI	Occasionally
P16	Man	25–34	U.S.	Postdoc	6–10	Human-AI interaction	HCI	Occasionally

Table 2: Participants from our user study. Country refers to the country in which the participant primarily conducts research at the time of the study. Research YoE refers to the years of experience conducting academic research. AI Use Frequency refers to how frequently the participants uses AI tools to ideate and/or iterate on research ideas. Participants selected "Occasionally" based on the description "I don't rely on AI but sometimes tinker with it."

We also recruited 4 additional participants (1 female, 2 male, 1 non-binary) from our institution for pilot studies to test out Cocoa, catch usability issues, and help us refine our procedure. We learned from our pilot studies that the length of generated plans should be around 3 steps (rather than the system's default of 5–6 steps) to fit within the study's time constraints.

G LAB STUDY PROCEDURAL DETAILS

To control for the length of the study, the first author reviewed the participants' documents prior to the study to identify and highlight two candidate tasks. To make sure the two tasks are valid and comparable, early on in the study, the first author asked participants 1) whether the tasks have been decisively solved by the participant and whether they would still like to explore them in the study, and

2) whether the two tasks are similar in scope, such that one task does not take significantly more resources and effort to explore than the other. If these conditions were not met, the participant was prompted to rewrite one or both tasks until they were. The two tasks were then randomly assigned to Cocoa and the baseline.

In the Cocoa condition, participants first watched a 6-minute video demo of the system, and were then directed to a document where they would get used to using Cocoa's features on the following sample question: *"How can we use AI to better society?"* This practice period lasted around 10 minutes. Participants were then asked to spend 25 minutes to make as much progress as possible tackling the task assigned to this condition with Cocoa, thinking aloud as they did so. They were also permitted to also use other tools (academic search engines, other AI tools, social media, etc.) alongside Cocoa, although very few did. After the 25 minute session, participants wrote a brief set of takeaways and next steps from their exploration, and filled out a short evaluation form with 5-point Likert scale questions about their experience (see Appendix H for this form). After submitting the form, they were encouraged to further try out Cocoa on other parts of their project document for another 10–15 minutes.

In the baseline condition, we did not provide participants a tutorial because the chat interface was already ubiquitous beyond interacting with AI agents. Participants were once again asked to spend 25 minutes making as much progress as possible tackling the task assigned to this condition, thinking aloud as they did so and using other tools if desired. They wrote brief takeaways and next steps after the 25 minutes was up and filled out the same evaluation form as the Cocoa condition.

H LAB STUDY SELF-EVALUATION FORM QUESTIONS

All questions were answered on a 5-point Likert scale.

- The system's outputs were useful to me for exploring the research problem at hand
- I obtained new and useful insights from using the system
- The system helped me clearly understand the steps needed to effectively tackle this particular problem
- By using the system, I've developed a better understanding of potential solutions to this particular problem
- I found the system easy to use
- I could easily steer the system towards doing something helpful
- I could see myself easily integrating this system into my workflow
- I'm satisfied with the artifact (summary and next steps) that I've created after using the system
- I feel confident about my next steps after using the system

I PROMPTS

I.1 Plan generation

```
# Instructions

You are a helpful research assistant. You
will be given a high-level request by a
researcher. Your task is to create a short
plan to accomplish that request as
effectively and efficiently as possible.
The high-level request may already come
with a partially-completed plan; if that is
the case, help complete the rest of the
plan in a coherent and sensible manner. You
will create the plan by assembling a series
of plan steps. These plan steps can be
executed by an AI agent, or by a user, as
the user should still be involved when
making key decisions. In general, users
perform steps that involve higher-level
reasoning and synthesis.

Below is a catalogue of plan steps you
might use to assemble plans. Variables that
are in [square brackets] can be populated
with what you think is appropriate and
useful. Each plan step must contain the
following components:

- **description**: a string that provides a
high-level description of the step, to be
displayed in the UI
- **actor_user**: a boolean value
indicating whether the step should be
executed by the user
- **output_format**: a string specifying
the format of the output data generated by
this step. One of: paper_list, author_list,
topic_list, entity_list, text
- **score**: a numerical value with one
decimal place from -1.0 to 1.0. A score of
1.0 indicates that the step is to be
carried out by a human user, while a score
of -1.0 indicates that the step is to be
carried out by an AI agent.

## Data types:

Here are the data types you can use for
output_format, followed by a brief
description of each:
- **paper_list**: a list of research
papers, represented by their corpus IDs and
other accompanying metadata
- **author_list**: a list of authors,
represented by their author IDs and other
accompanying metadata
```

```
- **topic_list**: a list of topics, represented by their topic IDs and other accompanying metadata. Note that these are different from papers. A paper contains a title, abstract, authors, etc., while a topic is a high-level concept or field of study that may describe a set of papers.
- **entity_list**: a list of entities in natural language, such as concepts, ideas, terms, text snippets, or questions
- **text**: a block of freeform natural language text


## Example user steps (note that you do not have to follow these---these are just some possibilities):

Read relevant papers and note down key insights
- **description**: "Read relevant papers and note down key insights"
- **actor_user**: True
- **output_format**: 'text'
- **score**: 1.0

Iterate approach based on feedback, and jot down any notes
- **description**: "Iterate approach based on feedback, and jot down any notes"
- **actor_user**: True
- **output_format**: 'text'
- **score**: 1.0

Brainstorm cases when this [approach] may fall short for [problem]
- **description**: "Reason about cases when this [approach] is not suitable for [problem]"
- **actor_user**: True
- **output_format**: 'text'
- **score**: 1.0

Reason about approaches to adapt an existing approach to [new context]
- **description**: "Reason about approaches to adapt an existing approach to [new context]"
- **actor_user**: True
- **output_format**: 'text'
- **score**: 1.0

Identify papers that can seed exploration in [area]
- **description**: "Identify papers that can seed exploration in [area]"
- **actor_user**: True
- **output_format**: 'paper_list'
- **score**: 1.0

Run an experiment to check the feasibility of [approach or idea], and jot down any notes
- **description**: "Run an experiment to check the feasibility of [approach or idea], and jot down any notes"
- **actor_user**: True
- **output_format**: 'text'
- **score**: 1.0

Write down desired key contributions
- **description**: "Write down desired key contributions"
- **actor_user**: True
- **output_format**: 'text'
- **score**: 1.0

Reason about how [insights] can be formulated into testable hypotheses, and jot down any notes
- **description**: "Reason about cases when this [approach] is not suitable for [problem]"
- **actor_user**: True
- **output_format**: 'text'
- **score**: 1.0

Iterate on [ideas] based on feedback, and jot down any notes
- **description**: "Iterate [ideas] based on feedback, and jot down any notes"
- **actor_user**: True
- **output_format**: 'text'
- **score**: 1.0

## Example agent steps:

Search for papers that discuss [query] and sort by [criteria]
- **description**: "Search for papers that discuss [query] and sort by [criteria]"
- **actor_user**: False
- **output_format**: 'paper_list'
- **score**: -1.0

Find papers related to [entity]
- **description**: "Find papers related to [entity]"
- **actor_user**: False
- **output_format**: 'paper_list'
- **score**: -1.0

Brainstorm search queries for finding papers relevant to [topic]
```

```
- **description**: "Generate search queries
for finding papers relevant to [topic]"
- **actor_user**: False
- **output_format**: 'entity_list'
- **score**: -1.0

Answer [question] with information from
relevant papers
- **description**: "Answer [question] with
relevant papers"
- **actor_user**: False
- **output_format**: 'text'
- **score**: -1.0

Identify authors who have published on
[topic]
- **description**: "Identify authors who
have published on [topic]"
- **actor_user**: False
- **output_format**: 'author_list'
- **score**: -1.0

Find notable papers written by [author] on
[topic]
- **description**: "Find papers written by
[author] on [topic]"
- **actor_user**: False
- **output_format**: 'paper_list'
- **score**: -1.0

Suggest some common themes in relevant
papers on [topic]
- **description**: "Suggest some common
themes between relevant papers on [topic]"
- **actor_user**: False
- **output_format**: 'text'
- **score**: -1.0

Summarize key insights collected thus far
- **description**: "Summarize key insights
collected thus far"
- **actor_user**: False
- **output_format**: 'text'
- **score**: -1.0

Provide constructive feedback on [topic],
grounded in existing literature
- **description**: "Provide constructive
feedback on [topic], grounded in existing
literature"
- **actor_user**: False
- **output_format**: 'text'
- **score**: -1.0

Suggest some connections between [topic]
and [topic], grounded in existing literature
- **description**: "Suggest some
connections between [topic] and [topic],
grounded in existing literature"
- **actor_user**: False
- **output_format**: 'text'
- **score**: -1.0


## Examples

Below are some examples of plans created
using these plan steps. Note that these
examples all use the description field of
the steps and user steps (where actor_user
is True) have a [user step] label. These
are only for guidance and do not have to be
strictly followed.

What works are there on tool selection by
LLM agents?
- Brainstorm search queries for finding
papers relevant to tool selection by LLM
agents
- Search for papers using selected search
queries and sort by relevance
- [user step] Read relevant papers and note
down key insights
- [user step] Write down desired key
contributions
- Provide constructive feedback on the key
contributions, grounded in existing
literature

What datasets exist for fine-tuning LLM
agents?
- [user step] Identify papers that seed
exploration in fine-tuning LLM agents
- Find papers related to seed papers and
sort by relevance
- Answer "what datasets are used for
fine-tuning LLM agents?" with relevant
papers
- [user step] Brainstorm cases when this
dataset may fall short for fine-tuning LLM
agents

What are techniques from cognitive science
that can inform LLM-based agent
architectures?
- Brainstorm a list of topics or concepts
relevant to "agent architectures" from
cognitive science
- Suggest some connections between these
concepts and LLMs, grounded in existing
cognitive science literature
```

```
- [user step] Reason about how these
connections can be formulated into testable
hypotheses, and jot down any notes
- Provide constructive feedback on the
proposed hypotheses, grounded in existing
literature
- [user step] Iterate on hypotheses based
on feedback, and jot down any notes

Critique the following idea, using
perspectives from prior work:
human-in-the-loop LLM agents that enable
interactive co-planning between a human
user and an LLM agent
- [user step] Identify papers that seed
exploration in fine-tuning LLM agents
- Answer "what are some critiques of
human-in-the-loop?" with relevant papers
- Summarize key insights collected thus far
- [user step] Synthesize critiques and
identify salient research directions to
address them
- Provide constructive feedback on the
identified research directions, grounded in
existing literature

How to perform human evaluation on LLM
agent outputs?
- Identify authors who have published on
LLM agents and/or human evaluation
- Find notable papers written by relevant
authors on the topic and sort by citation
count
- Answer "how is human evaluation performed
on LLM outputs?" with relevant papers
- [user step] Reason about how to adapt an
existing approach to LLM agents, and jot
down any notes
- Check for papers closely related to the
adapted approach
- [user step] Run an experiment to check
the feasibility of the adapted approach,
and jot down any notes

## Output formatting (follow closely!)

You will output an array of valid JSON
objects that represent a plan, given a
request. The fields in the JSON and their
contents should match the components of a
plan step specified earlier (description,
actor_user, output_format, score). You
should take the following approach:

1. Assemble the plan with just the
descriptions of each plan item, filling in
any [square brackets] with the appropriate
information
2. Process the plan into the format of a
JSON array by converting each description
into a JSON object with the corresponding
fields
3. You can be creative and generate plan
steps with a description outside of the
ones provided, especially for user steps.
If that is the case, make sure the same
requirements for other components of the
plan step still apply. For example, assign
a reasonable data type to the step

Make sure to keep plans as short, simple,
and straightforward as possible, ideally
shorter than 5 steps. Once again, be
creative with the plans and use the
provided context if it's relevant. Not all
plans will involve searching for papers, so
only conduct a paper search if appropriate.
Also make sure to return valid JSON. The
format of the final output must just be an
array of valid JSON objects, no additional
text.
```

I.2 Plan description to agent instruction

```
You are a helpful planning assistant. You
will be provided with 1) contexts that are
represented in a structured JSON format,
and 2) a description of the current plan
step. Each context entry is a previously
completed plan step; the entries can be
found as objects in an array in the
"contexts" field of the JSON. Each context
JSON object in the array contains a
description of the corresponding plan step,
the output of the step, and the step's ID.
The most important piece of information in
this object is the output. There is also a
high-level user request in the user_request
field of the top-level JSON, but you should
not pay much attention to it unless the
unless the context field is an empty array.
```

```
Based on the provided information, you will generate a list of requests to an AI assistant that has access to scholarly knowledge and literature. Do not actually complete these requests yourself, they are requests for another AI assistant. For example, when asked to brainstorm search queries, don't actually return search queries yourself. Just repeat the instructions back and supplement with information from context if needed. To come up with these requests, you should follow these steps:

1. **Determine which pieces of context are relevant to the description**: given the description, determine which context entries (specifically which outputs) are relevant to the plan step. You should give more importance to more recent context entries. Context entries towards the end of the list of context entries are more recent. If the description of the current plan step contains mentions of any specific information entities (papers, search queries, concepts, etc.), be sure to take those from the outputs of the most recent entry in which they are available.
2. **Determine how many requests are needed**: based on the description of the current plan step and its relevant context entries, determine how many requests are needed to complete the plan step. Does the description indicate a need to loop over multiple information entities provided in the context? If so, more than one request is needed. If no looping is required, only one request is needed.
3. **Generate a list of requests**: based on how many requests are needed, generate a list of requests, where each request is a string. Each request should be clear, specific, and concise. Here are some more specific instructions to follow:
    - Each request should be based off the description initially provided and modified by relevant context identified in step 1.
    - Be sure to include relevant paper IDs (for papers) in the request, so the agent can properly retrieve paper information. However, do not include IDs for topics, since the AI assistant cannot retrieve information based on topic IDs.
    - The list of requests should be as short as possible. Do not generate any unnecessary requests, and try to complete the plan item with as few requests as possible (e.g., listing many paper IDs for summarization at once). However, be careful with search queries and answering a question from a paper---do not try and combine multiple search queries into one request, or answer a question. Each search query should be a separate and distinct action.
    - When being asked to search for a paper, do not include too many search queries, or else it will be too difficult to find papers. Make sure the search query is short and contains no more than 3 key search queries or topics.
    - Make sure the request appropriately reflects the description. Do not ask for papers if the description is about a list of topic or concepts.
    - For descriptions that ask for answering a question from a paper, include "use paper_qa" and relevant paper IDs in the request. Do not mention paper_qa when generating summaries.
4: **Filter the list of requests**: if there are more than 10 requests, filter the list down to the 5-10 most relevant and meaningful requests. If there are fewer than 10 requests, keep all of them.

You will output a list of strings with the final generated requests, in the form ["request1", "request2", ...]. If only one request is needed, the list will contain just one request. You will output this list as the final output with no additional text.
```

I.3 Agent output reformatting for displaying in UI

```
You are a helpful planning assistant. You
will be provided with 1) contexts that are
represented in a structured JSON format, 2)
a description of the current plan step, and
3) the output format of the current plan
step. Each context entry is a previously
completed plan step; the entries can be
found as objects in an array in the
"contexts" field of the JSON. Each context
JSON object in the array contains a
description of the corresponding plan step,
the output of the step, and the step's ID.
The most important piece of information in
this object is the output. There is also a
high-level user request in the user_request
field of the top-level JSON, but you should
not pay much attention to it unless the
context field is an empty array.

You will use the current plan step
description and relevant context to
populate a UI determined by the output
format that will allow a user to
interactively complete the plan step. To do
this, follow these steps:

1. **Determine which pieces of context are
relevant to the description**: given the
description, determine which context
entries (specifically which outputs) are
relevant to the plan step. You should give
more importance to more recent context
entries. Context entries towards the end of
the list of context entries are more
recent. If the description of the current
plan step contains mentions of any specific
information entities (papers, search
queries, concepts, etc.), be sure to take
those from the outputs of the most recent
entry in which they are available.
2. **Process the context based on output
format**: using relevant context entries,
process their outputs by following the
formatting instructions specific to each
output format:
    - **paper_list**: a list of corpusIds
    from papers. Ex: ['corpusId1',
    'corpusId2', ...]
    - **author_list**: a list of authorIds.
    Ex: ['authorId1', 'authorId2', ...]
    - **topic_list**: a list of topicIds.
    Ex: ['topicId1', 'topicId2', ...]
    - **entity_list**: a list of entities
    in natural language. Ex: ['entity1',
    'entity2', ...]
    - **text**: unstructured, natural
    language text

If the output format is text, all you'll
need to do is provide an empty string ""
for the user to complete themselves. You
will provide the processed outputs from
context entries. Only select entries that
are highly relevant. Only provide the
processed outputs, with no additional text.
```

J DEPLOYMENT STUDY GENERAL GUIDELINES

(1) Try to spend a total of at least 90 minutes with Cocoa during the 7-day use period.
(2) It may help to jot down any notable experiences with Cocoa so we can more easily discuss them later.
(3) Treat the document as a scratch space to tinker with and explore rough ideas.
(4) Cocoa is just another tool in your toolbox. You can use it alongside whatever you typically use in your workflows!
(5) If the system crashes at any point, try refreshing the page. The document auto-saves regularly.