JumpStarter: Getting Started on Personal Goals with AI-Powered Context Curation

Sitong Wang*
Columbia University
New York, NY, USA
sw3504@columbia.edu

Xuanming Zhang*
Columbia University
New York, NY, USA
xz2995@columbia.edu

Jenny Ma
Columbia University
New York, NY, USA
jm5676@columbia.edu

Alyssa Hwang
University of Pennsylvania
Philadelphia, PA, USA
ahwang16@seas.upenn.edu

Lydia B. Chilton
Columbia University
New York, NY, USA
chilton@cs.columbia.edu

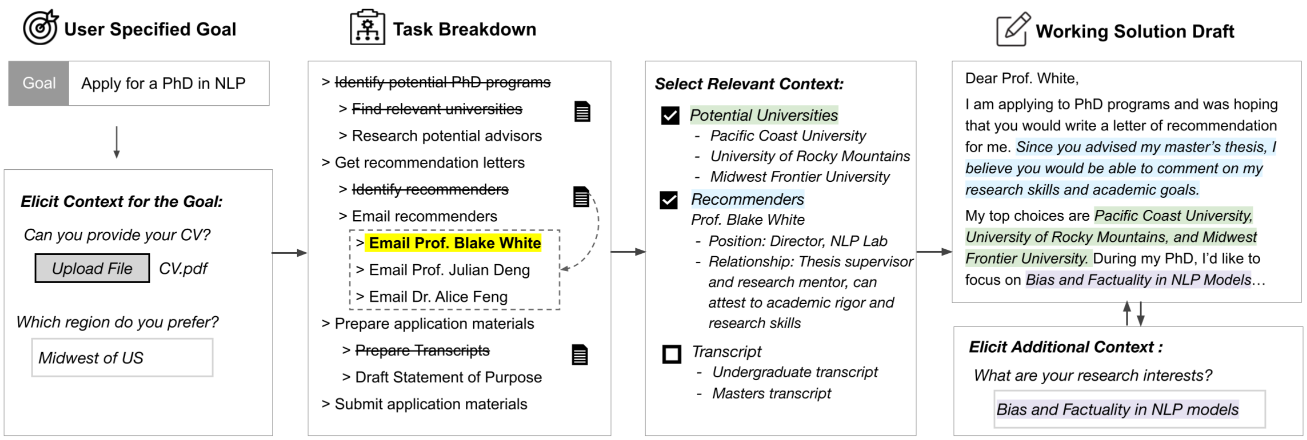


Figure 1: JumpStarter helps users get started on their personal goals through AI-powered context curation. It first takes the user's goal and gathers context for the goal. It then breaks down the goal into actionable subtasks. For each subtask, it helps users choose relevant context and draft working solutions. It also aids users in refining these drafts by eliciting further context.

Abstract

Everyone aspires to achieve personal goals. However, getting started is often complex and daunting, especially for large projects. AI has the potential to create plans and help jumpstart progress, but it often lacks sufficient personal context to be useful. We introduce JumpStarter, a system that uses AI-powered context curation to create action plans and draft personalized working solutions. JumpStarter assists users by posing questions to elicit relevant context, breaking down goals into manageable steps, and selecting appropriate contexts to draft working solutions for each step. A technical evaluation indicates that context curation results in plans and working solutions of higher quality. A user study demonstrates that compared to ChatGPT, JumpStarter significantly reduces users' mental load and increases their efficiency in kickstarting personal projects. We discuss the design implications of AI-powered context curation to facilitate the use of generative AI in complex problem-solving.

CCS Concepts

• **Human-centered computing → Interactive systems and tools**.

*These authors contributed equally to this work.

Keywords

context curation, planning, action initiation, personal goal management, productivity

1 Introduction

Achieving personal goals is a common aspiration. Personal projects are part of everyday life, ranging from moving to a new apartment, to starting a YouTube channel, to applying for PhD programs. Getting started on these projects is crucial for moving from mere intention to actual fulfillment [17, 43]. However, getting started on complex goals can be daunting and challenging. When faced with a new project, one needs to engage in sensemaking, construct a mental model of the process, and adapt it to their specific situation to figure out the actionable steps, all of which can be cognitively demanding [23]. Another major challenge is initiating concrete actions. For example, there is a significant gap between knowing one needs to write a request email and actually being ready to write it [43]. This shift requires concretization—formulating a specific request or language and gathering the necessary information. Identifying and utilizing the right resources to operationalize the plan can be a significant hurdle to getting started.

Generative AI has the potential to create plans and help jumpstart progress, but it often lacks the personal context needed to be useful. Everyone has unique situations and requirements that need to be considered when embarking on personal projects. For example, to get started on the goal of "applying for a PhD," it is important to consider preferred research areas and personal background. Moreover, a complex goal like this must be further decomposed to become actionable. It involves juggling multiple tasks, such as researching universities and requesting recommendation letters, each of which requires further strategic planning and coordination with other tasks. For example, requesting recommendation letters is itself a goal that needs further breakdown—one must first identify recommenders, gather supporting documents, send them individualized email drafts, and follow up accordingly. Popular AI tools like ChatGPT allow users to provide context throughout their interactions. However, it can be mentally demanding for users to recall and articulate the necessary context relevant to the task and to frame the prompt accordingly. Ideally, ChatGPT would remember all the context provided by users in the chat and select the most appropriate ones to support each task. However, this capability is limited by the inherent restrictions on context window size [8, 39, 51], and the process is not transparent to users. Moreover, linear chat interfaces like ChatGPT lack the structure to help people systematically plan their roadmap toward their goals. Although people can ask ChatGPT to break down the steps further, it is easy to lose track of their progress and the context of each task.

In the psychology literature, the management of personal goals is often linked to executive functions such as planning, task initiation, and working memory [14]. Planning involves a step-by-step breakdown of a broader goal [28]. This can be challenging as it requires identifying key tasks, estimating realistic timelines, and foreseeing potential obstacles, all demanding significant effort [46]. While having a plan is crucial, it can sometimes remain abstract and lack the specificity and granularity required for tangible actions [17, 28]. Initiating concrete tasks demands heightened focus and meticulous attention to detail, which significantly increases cognitive load [45]. Effective execution of such tasks necessitates maintaining an active working memory, which aids in retaining and manipulating the relevant information needed to manage the task [3]. Additionally, the Ovsiankina effect [42] indicates that people are more likely to finish the task once they initiate it, underscoring the critical role of action initiation in goal completion.

To better support people in getting started on their personal goals, we introduce a system called JumpStarter. Through AI-powered context curation, the system assists users by 1) decomposing tasks into hierarchical steps and 2) drafting working solutions. Here, the "working solution" refers to any tangible result that enables users to take real-world action. For example, a working solution could be a draft email to send to a recommender or a study schedule for the GRE. The process begins when a user inputs their personal goal. The system then poses questions to elicit the relevant context for the goal. Based on user responses, the system breaks down the goal into subtasks. The user can then start exploring each subtask. The system notifies users if it detects that the subtasks are not actionable enough, prompting further breakdown. If the user opts to decompose further, the system assists in generating subtasks. It also helps detect if "forking" is necessary, which involves breaking down the task based on different entities (such as sending requests to different recommenders), focusing on one at a time. If the user is ready to start on a task, the system selects the relevant context and generates working solutions that incorporate this context. If the user is not satisfied with the suggested working solution, they can request more elicitation questions or provide their own prompts to refine it. Users can select parts of the suggested solution and attach them to the subtask, and the system will add them to the context pool. The user can then move on to the next subtask to explore.

This paper makes the following contributions:

- JumpStarter, an AI-powered system for helping people get started on their personal goals with three main automated features: 1) context curation (including context elicitation, context saving, and context selection), 2) task decomposition and 3) working solution drafting.
- A technical evaluation with and without context curation, showing the importance of context curation for getting plans and working solutions of high quality.
- A user study showing that compared to ChatGPT, JumpStarter significantly reduces users' mental load and increases their efficiency in kickstarting personal projects.

We conclude by discussing the design implications of AI-powered context curation to facilitate the use of generative AI in complex problem-solving.

2 Related Work

2.1 Theory of Goal Achieving

Achieving personal goals is a complex process that involves various aspects of human psychology. This process encompasses numerous psychological factors, including motivation, self-belief, social support, feedback, planning, and more, as highlighted in theories like goal-setting theory [34] and the theory of planned behavior [2]. To better understand how individuals navigate this intricate journey, action phase theory offers a comprehensive framework [17, 18]. It outlines the progression from initial desires to goal completion through distinct stages: pre-decisional, pre-actional (planning), actional, and post-actional phases, each critical in transforming aspirations into accomplishments. Action phase theory also emphasizes the importance of action initiation, as it marks the transition to the actional phase where individuals actively pursue their goals [17, 18].

While other factors such as motivation and self-belief [2, 13, 36] are also crucial, we recognize the potential of computers in supporting the planning and action initiation aspects, especially for goals that primarily involve cognitive tasks or knowledge work (such as applying to PhD programs). In this work, we focus on providing support for individuals during the planning and action initiation phases of achieving their complex cognitive goals, helping them bridge the gap between mere goal intention and taking real-world action.

Planning presents its own set of challenges. In a probe study focusing on managing academic procrastination [5], a majority of participants (13 out of 15) expressed a desire for structured action steps for their tasks. Many participants pointed out the struggles with unstructured or expansive tasks, indicating a need for planning support that includes step-by-step guidelines, structuring recommendations, and adaptive task breakdown. According to the implementation intention theory, plans that specify when, where, and how actions should be taken can transform abstract goals into concrete strategies, bridging the gap between intention and action [19]. Similarly, goal-setting theory emphasizes the need for specific and actionable goals [34]. Personalization is emphasized as another important dimension, suggesting that adapting goals to fit one's real-life situations can significantly enhance the effectiveness of planning [15].

Action initiation is also challenging. A major hurdle is the need to take concrete actions—each task requires specific context to be acted on. However, people often juggle multiple tasks in their daily lives [12]. Frequent context switching between tasks can significantly drain cognitive resources, reducing the efficiency with which one can work on a project [27]. To initiate action, context curation is necessary to create an enabling environment and to aid in concretization. The Fogg Behavior Model suggests that to persuade people to take action, one needs to provide facilitators [16]. An effective facilitator should inform people that the target behavior will not require a resource they do not have at that moment. Context curation—which involves creating a conducive environment tailored specifically for the task at hand—can thus facilitate the initiation of action [48].

2.2 Solutions to Support Action Planning

An action plan refers to a concrete, step-by-step breakdown of a task that helps people complete it [28]. Prior studies show that people prefer not to make action plans for their own tasks [29, 46], as the costs are immediate but benefits are deferred [26]. Therefore, it will be beneficial to provide people with support for action planning.

There are many HCI works focusing on community or crowd-based approaches to help people do a step-by-step breakdown of their goals. For example, TaskGenies [28] decomposes tasks into concrete steps by using online crowds to create new plans and using algorithms to identify and reuse existing ones. PlanHelper [33] helps users construct activity plans (such as for bodybuilding or sightseeing) by processing answer posts in community-based QA platforms. Specifically for writing tasks, a vocabulary [26] is introduced as a cognitive scaffold to enable other people who have the necessary context (such as collaborators), to create action plans for writers. In a collaborative teamwork setting, MixTAPE [40] helps generate action plans for design teams by converting call notes and client briefs into prioritized, assigned tasks using a synthesis algorithm and machine learning models. While not directly supporting action planning, many other works, particularly in the crowdsourced field, focus on decomposing a large task into manageable micro-tasks that help people make progress with limited time and resources [7, 46, 49].

With the advancement of generative AI techniques, large language models (LLMs) like GPT-4 [1] hold the potential to provide intelligent support for planning. There is a series of NLP works that study the potential of automating real-world planning [30, 50, 52]. For example, a multi-agent approach [30] is introduced to determine the steps needed to complete a task (such as how to plant a garden), emphasizing the importance of tailoring the steps to meet users' specific needs (such as planting a garden without using pesticides). However, studies found that the current language agents are not yet capable of automatically handling complex planning tasks [50, 52]; humans still need to be involved to make meaningful plans for real-world tasks.

A popular online tool called Goblin Tools MagicTodo[1] uses AI techniques to support people in breaking down their tasks layer by layer, but the accuracy seems limited. A recent HCI work, ExploreLLM [35], also explores helping users decompose complex tasks into subtasks with LLMs. Their study shows that a schema-like structure is useful for people to do planning tasks, with ChatGPT as a baseline. The system also provides a dedicated input box to prompt users to enter their personal context and preferences for personalization. However, ExploreLLM only examines one layer of task decomposition and relies on users to provide context themselves. In JumpStarter, we explore multi-layer task decomposition and intelligent context curation throughout the process.

2.3 Solutions to Support Action with Context Curation

Curating appropriate context relevant to a task is essential for enabling people to take action on it. However, people often need to juggle multiple tasks in daily life, and context switching is never effortless [27], hindering people's ability to make progress. Therefore, task-specific context management is important and beneficial to support.

Previous system works focus on help people do task-centric information management. For example, UMEA [25] helps users create and manage project-specific work context by providing users dedicated project spaces and monitoring user activities in a desktop environment. Mylar [27] introduces a task context model to help programmers find and identify the information relevant to their task at hand, which is built using the task interaction history. Personal Project Planner [24] introduces an in-context create feature that enables users to produce documents and emails within a project plan, automatically inserting hyperlinks as context to the task. Similarly, Towel [9] allows users to drag any semantic object (project, email, person etc.) onto the to-do list, saving them as resources under the to-do item.

To better support users in taking action on their tasks, the existing context is often not sufficient, and the system may need to elicit more context from the user. For example, PExA [37] introduces an intelligent personal assistant that helps people manage their tasks by suggesting context-appropriate follow-up questions. The use of large language models has also been explored to support context elicitation. For example, the OPEN framework [20] uses Bayesian Optimal Experimental Design to choose informative queries and employs language models to formulate these queries in natural language, enhancing preference elicitation accuracy. Generative Active Task Elicitation (GATE) [31] leverages language models to interactively generate questions and scenarios, guiding users to clarify and refine their intended outcomes in tasks, which helps elicit more precise and contextually rich information from users. In JumpStarter, we explore generative AI-powered context curation, which covers task-specific context elicitation, context saving, and context selection, in the scenario of getting people started on complex personal projects.

[1] https://goblin.tools/

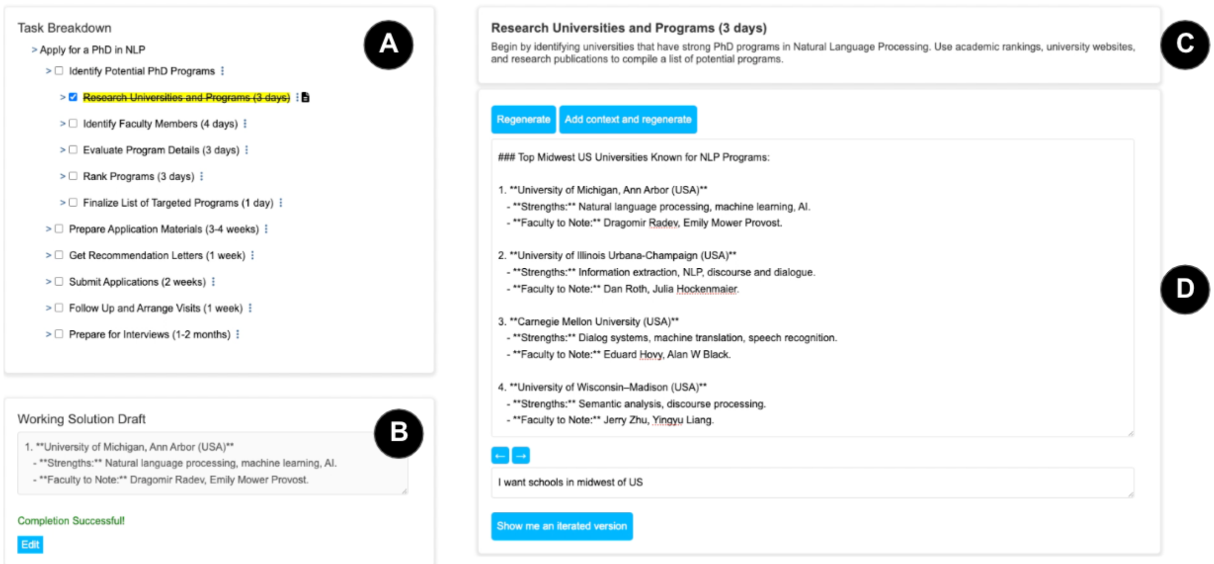


Figure 2: A screenshot of JumpStarter creating plans and drafting working solutions for the goal *Apply for a PhD in NLP*. (A) Task breakdown is shown as a subtask tree, with the goal being the root node. Subtasks decomposed from the same parent node are shown on the same level. (B) Saving the working solution draft. (C) Detailed descriptions of the selected subtask are shown. (D) The working solution is generated, considering the specification from the user – "*I want schools in midwest of US*". Users have three options to improve the draft: regenerate, add context and regenerate, and iterate based on users' new specifications.

3 JumpStarter System

JumpStarter is an interactive system that helps users get started on their personal projects with AI-powered context curation. The system takes the user-specified goal as the input, and outputs plans and working solutions for the user. Here, a working solution refers to any tangible result that can assist users in taking real-world actions related to a task. For instance, a working solution for requesting recommendation letters might be an email draft to solicit letters from advisors. Similarly, for passing the GRE test, a working solution could include a study schedule. JumpStarter is powered by GPT-4[2] and implemented with a Flask/Python web framework. In this section, we describe how our system works with an example walk-through, and the implementation details of the system.

3.1 System Description

As an example, consider John, who aims to apply for a PhD in NLP. In this subsection, we will demonstrate the typical interactions between John and our system through a detailed walk-through.

First, John establishes his goal by entering "*Apply for a PhD in NLP*" into a text box. He then clicks the *Start* button to submit it.

Context Elicitation for Goal. Based on John's goal, the system uses the LLM to generate questions to elicit context from him (see Figure 3). For John, the system generates three questions: "*Can you provide your CV or Resume?*", "*Which universities or programs are you considering for your PhD in NLP?*", and "*Do you have letters of recommendation, or do you need guidance on how to obtain them?*". Since John does not know which universities he might be applying to, and he has not obtained the recommendation letters, he only uploads his CV to the first question. Then, he clicks the *Let's get started* button to proceed. This first step aims to clarify John's goal by eliciting the necessary context that will be used globally throughout John's interaction with JumpStarter, ensuring that the plans and working solutions are created based on John's situations.

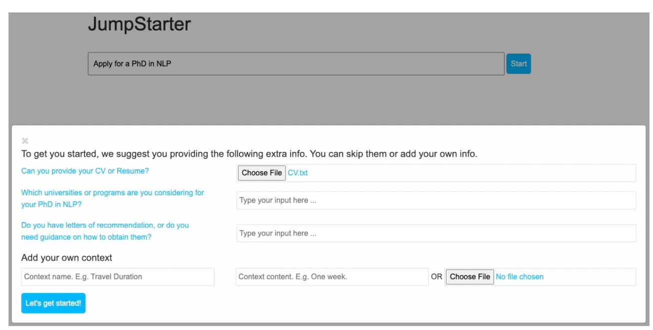


Figure 3: JumpStarter generates questions to elicit necessary context from users to clarify the goal. The user uploads his CV and adds his own context *Preferred location: United States.*

Subtask Suggestion. Leveraging the context obtained from the first step, JumpStarter generates the initial subtask tree for John, as depicted in Figure 4E, which consists of a list of subtasks. It also presents John with an overview of the titles, descriptions, and duration of completion of these subtasks (see Figure 4F). After reviewing them, John gains brief insights into each subtask and decides to explore them in order.

[2]We used `gpt-4-turbo` throughout the system.

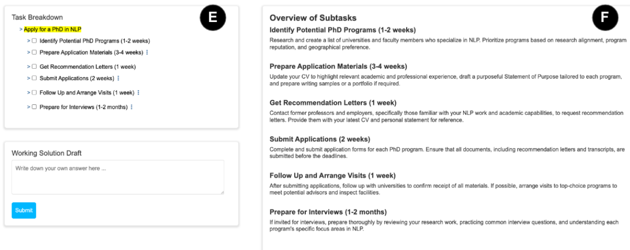


Figure 4: The initial subtask tree and the overview for the goal *Apply for a PhD in NLP.* (E) The task breakdown for the goal. (F) The overview of all the subtasks of the goal, including the titles, descriptions, and duration of completion of the subtasks.

Subtask Detection. After reviewing the initial subtask tree, John thinks the list generally makes sense and proceeds to click on the first node: *Identify Potential PhD Programs: Research and create a list of universities and faculty members who specialize in NLP.* As shown in Figure 5, the node is highlighted, and its title and description are shown on the top right of the interface. At the same time, the system shows a message as it detects this task needs to be further decomposed to make it actionable. The system offers two options for the user to select: the recommended option *Decompose the task* and the alternative option *Directly start drafting*. Clicking on the first button will further break down the selected task into several subtasks, while clicking on the second will generate a working solution for the task. Following the recommendation, John clicks the button *Decompose the task*, after which the subtasks for *Identify Potential PhD Programs* are generated, as shown in Figure 5.

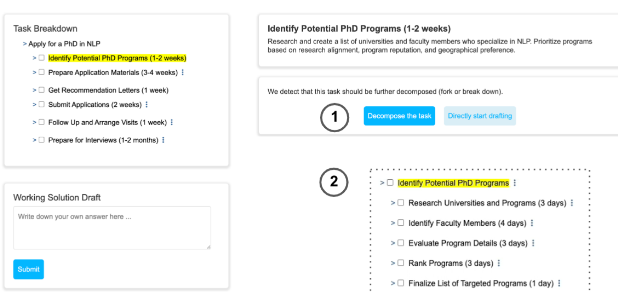
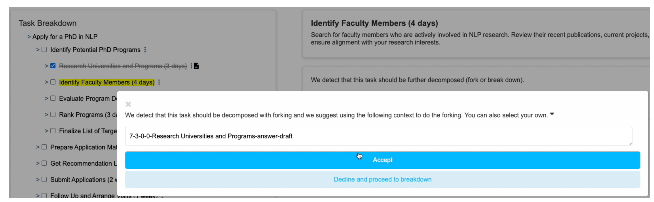
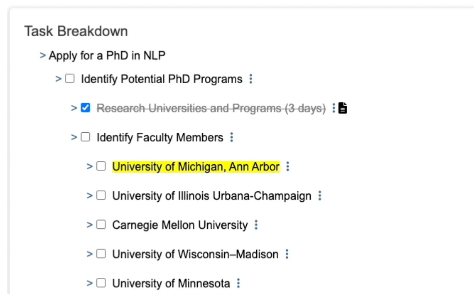


Figure 5: JumpStarter suggests further decomposition for the first subtask *Identify Potential PhD Programs.* (1) John presses the button *Decompose the task.* (2) JumpStarter generates the subtasks for John.

Working Solution Drafting. John reviews all subtasks of *Identify Potential Programs* and proceeds to click on the first subtask *Research Universities and Programs*, which is about "*Identify universities that have strong PhD programs in NLP. Use academic rankings, university websites, and research publications to compile a list of potential programs.*" JumpStarter deems this subtask actionable and suggests John directly start drafting the working solution for it. Following the suggestion, he obtained the initial working solution that contains a list of universities offering PhD programs in NLP. Upon reviewing the list, John realizes it includes many global institutions. However, he prefers schools in the midwest of the United States. To iterate, he enters "*I want schools in midwest of US*" (see part D in Figure 2). After reading it, John is happy to have this list as the first draft of his school application list. Note that John is provided with three alternatives to further improve the draft, as shown in part D of Figure 2: 1) regenerate; 2) add more context and regenerate; and 3) specify details and iterate on the current draft. Subsequently, John copies and pastes the draft into the working solution section of the task *Research Universities and Programs* (see part B in Figure 2) and saves it, which appears as a file icon on the tree (see part A in Figure 2) with the subtask crossed out.



(a) Selecting relevant context for forking.



(b) Entity-based task decomposition (forking).

Figure 6: Context selection for forking and the task decomposition after applying forking on *Identify Faculty Members.*

Forking Detection and Context Selection. After obtaining the initial working solution for the first subtask, John moves on to the next subtask titled *Identify Faculty Members.* JumpStarter suggests that the subtask lacks the specificity needed for direct action and suggests further breakdown. John agrees and clicks the *Decompose the task* button. Following this, JumpStarter determines that the subtask should be forked to address the specifics of each program identified earlier. It then selects and presents the relevant context, specifically the working solution from the previously completed step *Research Universities and Programs* that includes a list of universities (see Figure 6a). John agrees with the system's suggestion and clicks the *Accept* button to proceed with forking using the selected context. Note that if John disagrees with the suggested context, he can select other context variables from a dropdown button; alternatively, he can also *Decline and proceed with breakdown* if he thinks that the subtask should not be forked. The updated subtask tree after forking is displayed in Figure 6b. With the new subtasks of evaluating different programs, John proceeds to explore each program in greater detail by clicking on them sequentially.

With all of these functionalities of JumpStarter, John manages to generate a plan and working solutions that he is satisfied with. This enables him to progress further and ultimately achieve his goal of applying for a PhD in NLP.

3.2 Data Representation

In our system, each context is represented as a key-value pair, where the key denotes the context's name, and the value is its content. For instance, the pair "Location preference: Midwest of US" indicates the user's preference for a specific location - Midwest of US.

There are two types of context in our system: global context and local context. Global context refers to context that is universally applied by default throughout the system. Specifically, it consists of all context elicited from the questions the system asks at the very beginning after the user enter their goal. The local context includes all saved working solution drafts and the context added by the user throughout their interaction with the system.

Additionally, we represent lists of subtask nodes generated by our system in the form of a tree, with the user's goal serving as the root node. Each subtask node stores task-specific attributes, including working solution drafts, task titles, task descriptions, and estimated duration of completion, among others.

3.3 Feature Implementation

In this section, we describe the implementation details of key features of our system. We introduce each feature's implementation in the order they appeared in Section 3.1. Note that the user's goal and the global context are integrated into the system prompt, which we use to implement all of our key features.

3.3.1 Context Elicitation for Goal. Based on the user's input goal, JumpStarter poses questions to elicit context relevant to that goal from the user. To achieve this, we instruct GPT-4 to assess which existing documents the user might possess that could serve as the context. GPT-4 will suggest what file to upload along with an elicitation question. Our system then lists that question in the UI and allows the user to upload their file. Alternatively, if GPT-4 generates a question that does not require any documents from the user, an input text box will be provided in the UI, allowing users to directly type their responses. See the GPT prompts we used in Appendix A.1.

3.3.2 Subtask Suggestion. Given a task that the user decides to break down, JumpStarter decomposes it into a list of subtasks and attaches them as child nodes to the task node. We instruct GPT-4 to generate the list, where each subtask should have a name, associated descriptions, and an estimated duration for completion. This helps users understand the scope and duration of each task. To minimize the occurrence of repetitive subtask nodes across the tree, we prompt GPT-4 to consider the overall tree structure and suggest subtasks that naturally complement the existing hierarchy. See the GPT prompts we used in Appendix A.4.

3.3.3 Subtask Detection. Given a task that the user selects, JumpStarter evaluates whether it is actionable and sufficiently detailed. If it is not actionable, JumpStarter recommends further breakdown. If it is actionable, JumpStarter recommends that the user directly drafts the working solution. We use GPT-4 to perform subtask detection. To determine the optimal prompting method, we experimented with different prompting techniques. To optimize both the accuracy and user experience in terms of loading time, we decided to adopt Chain of Thought (CoT) prompting with few-shot examples and integrate the tree level of the target task into the prompt. More details about the experiments and results can be found in Section 4.1. See the GPT prompts we used in Appendix B.

3.3.4 Working Solution Drafting. Given a task that the user decides to work on, JumpStarter helps users draft working solutions, which contains the following parts.

Context Selection for Working Solution Generation. JumpStarter starts by selecting the relevant context for the given task. Specifically, given the title and detailed descriptions of the task, we prompt GPT-4 to select the most relevant context from the current local context collection. The selected context keys are presented as a checklist in a pop-up window in the system. Users are allowed to de-select any context key if necessary. Users can also add any context they deem useful from a drop-down list. Note that since the local context collection is empty for John when he drafts his first subtask, no local context can be selected, and thus, no pop-up window appears in his story in Section 3.1. We adopt the CoT prompting method to achieve this feature. See the prompt we used in Appendix A.3.1.

Working Solution Generation. The selected context is then applied to generate the working solution draft. We directly prompt GPT-4 to help with the task in a zero-shot manner by providing the task title, its descriptions and selected context as the input (along with the user's goal and the global context in the system prompt). See the prompt we used in Appendix A.7.

Context Elicitation for Working Solution Iteration. As mentioned in Section 3.1, users can click on the "Add context and regenerate" button if they attempt to iterate on a working solution draft but are not sure how. The system will then perform context elicitation for working solution iteration for the user. Similar to context elicitation for goal (see Section 3.3.1), the system will provide questions to elicit more context from users, hopefully enabling them to get a better draft. To achieve this, we prompt GPT-4 in the CoT manner. See the prompt we used in Appendix A.2. With the newly elicited context, the system then triggers the "working solution generation" feature to generate an iterated draft for users.

3.3.5 Forking Detection and Context Selection.

Forking Detection. Given a task that requires further breakdown (as determined by the "Subtask Detection" feature), JumpStarter will then assess whether it requires forking, i.e., whether it needs to be decomposed based on different entities. Unlike the typical task breakdown that treats subtasks as sequential, forking focuses on decomposing subtasks in a way that allows them to be completed in parallel, independent of the order in which they are generated. If it is not forking, JumpStarter directly goes into a typical task breakdown without any changes in the UI. If it is forking, JumpStarter then triggers the context selection for forking feature (see the paragraph below). We prompt GPT-4 in a CoT fashion with four in-context examples. See the GPT prompts we used in Appendix A.6.

Prompting techniques	Accuracy		Statistics	
	Mean	SD	p	Sig.
Zero-shot	.35	.000		
Few-shot	.58	.040		
+ CoT	.62	.050	.405	-
+ CoT + Tree	.69	.020	.004	**
+ CoT + Draft	.72	.020	.009	**
+ CoT + Tree + Draft	.87	.040	.000	***

Table 1: The technical evaluation results for Subtask Detection comparing different prompting techniques, where the p-values ($-$: $p > .100$, $+$: $.050 < p < .100$, $*$: $p < .050$, $**$: $p < .010$, $***$: $p < .001$) are reported. Note that the p-values are computed against the few-shot-only baseline. Few-shot combined with CoT+Tree+Draft achieved the best accuracy.

Context Selection for Forking. Given a task that requires forking (as determined by the "Forking Detection" feature), we instruct GPT-4 to select the most relevant context keys for forking. See the prompt we used in Appendix A.3.2.

4 Technical Evaluation

JumpStarter helps people create a hierarchical action plan and draft personalized working solutions by detecting subtasks and curating context. In this section, we describe how we evaluated the accuracy of subtask detection and the impact of context curation on generated plans and working solutions.

4.1 Subtask Detection Experiments

4.1.1 Experiment settings. Subtask detection involves determining whether a task needs further decomposition to become actionable. With vanilla zero-shot and few-shot prompting as our baselines, we experimented with three methods for detecting subtasks with GPT-4 (see Appendix B for detailed prompts). We first experimented with few-shot prompting in the Chain-of-Thought ("CoT") style [47]. We then integrated the tree levels of the task nodes into the prompt, hypothesizing that nodes at shallower levels (e.g., the first level) would be less likely to be actionable than those at deeper levels ("Tree"). We also explored incorporating the initial solution draft into the prompt to encourage the system to suggest task decomposition depending on the quality of the draft ("Draft"). We evaluated the accuracy of these methods by constructing a test suite of 20 examples from real and diverse use cases, which were labeled by expert users. In all experiments, the temperature was set to 1, the max tokens were set to 2048, and *top-P* was set to 1. We report the average across five runs for each experimental setting.

4.1.2 Results and Findings. As summarized in Table 1, the zero-shot method yielded the lowest accuracy of only 0.35, which also implies an inherent difficulty in the task itself. We observe enhanced performance with few-shot prompting (0.58), with accuracy rising even more when using the CoT prompting paradigm (0.62). Combining CoT few-shot prompting with the task node tree levels (0.69), initial solution draft (0.72), or both (0.87), all significantly led to enhanced performance. Generating the initial solution draft introduced a trade-off between latency and accuracy, so we opted for the slightly less performant few-shot CoT with tree levels for a better user experience.

Participant	Personal goal	Condition order
E1	Apply to a fellowship	(1) → (2) → (3)
E2		(2) → (3) → (1)
E3	Get a driver's license	(3) → (1) → (2)
E4		(1) → (2) → (3)
E5	Organize a team event	(2) → (3) → (1)
E6		(3) → (1) → (2)

Table 2: Overview of expert participants for the comparative study. Six experts were assigned one of three goals to evaluate under all three conditions, which were presented in shuffled order to avoid biasing the results.

4.2 Comparative Study with and without Context Curation

To evaluate how context curation affects the quality of JumpStarter's proposed plans and working solutions, we conducted a controlled within-subjects lab study for the following conditions:

(1) *Context dumping*: context saving only.
(2) *Context filtering*: context saving and context selection.
(3) *Context curation*: context saving, context selection, and context elicitation.

Besides ablating the use of both context selection and context elicitation in Condition (1) and context elicitation alone in Condition (2), all features and UI were kept exactly the same. We hypothesized that action plans and working solutions developed with the full context curation method, represented by Condition (3), would be judged as higher quality than those created without it.

4.2.1 Participants and Procedure. We evaluated the effect of context curation on subtask and working solution quality for three preselected personal goals inspired by [32]: (a) Apply to a fellowship, (b) Get a driver's license, and (c) Organize a team event. We then used a university mailing list and word of mouth to recruit participants. Each goal was assigned two "expert participants"—those who reported achieving the goal in the past six months. Overall, we recruited six expert participants for this study (average age=25.8, three female, three male). Participants were compensated $20 per hour, with sessions lasting about 1.5 hours each.

During the study, we first introduced the expert participants to the concept of action plans and working solutions with examples. We instructed the participants that they would be using three versions of the same system for their respective goal, from start to finish. The system versions represented the three experimental conditions, which were presented in shuffled order among participants to counterbalance the learning effect (see Table 2). We demonstrated how to use each version before each participant used it. We asked the participants to generate the subtasks and working solutions exactly once. They then rated the quality on a seven-point Likert scale while providing a verbal explanation for their choice.

	Subtask quality	Working solution quality
(2) vs. (1)	$p = .436$ -	$p = .001$ **
(3) vs. (1)	$p < .001$ ***	$p < .001$ ***
(3) vs. (2)	$p < .001$ ***	$p < .001$ ***

Table 3: The statistical test results comparing three conditions, where the p-values (−: $p > .100$, +: $.050 < p < .100$, ∗: $p < .050$, ∗∗: $p < .010$, ∗ ∗ ∗: $p < .001$) are reported. Our full context curation method (3) outperformed context saving alone (1) and context saving and selection (2).

4.2.2 Results and Findings. Overall, 46 subtasks, or plan items, were generated for Condition 1 (*context dumping*: context saving only), 50 for Condition 2 (*context filtering*: context saving and context selection), and 50 for Condition 3 (*context curation*: context saving, context selection and context elicitation). An equal number of corresponding working solution drafts were also produced, as we asked participants to generate a draft only once for each subtask. Our method of *context curation*, represented by Condition (3), performed the best for both subtask quality and working solution quality. For subtask quality, *context curation* ($\mu = 6.12$, $\sigma = 0.95$) was rated higher than the similarly performing *context dumping* ($\mu = 5.26$, $\sigma = 1.26$) and *context filtering* ($\mu = 5.28$, $\sigma = 1.25$) methods. For the quality of working solutions, *context curation* ($\mu = 6.36$, $\sigma = 0.95$) outperformed *context filtering* ($\mu = 5.68$, $\sigma = 1.01$) by a similar margin, which also outperformed *context dumping* ($\mu = 5.04$, $\sigma = 1.04$). These results and corresponding p-values are reported in Table 3.

Our results show that the context selection feature significantly enhances the quality of working solutions compared to the baseline. Unlike context saving only, which keeps all context in the context window all the time, context selection requires the LLM to explicitly choose relevant context from the available pool. During sessions under this condition, participants often remarked that the generated solutions appeared to take into account what they had input in previous subtasks. This was particularly evident in "summarizing" tasks, where the system could provide a personalized checklist for tasks like applying for scholarships and driver's licenses, or an overall itinerary for a team event. E5 referred to the event itinerary they got as a *"very useful synthesis of everything I've explored."* In contrast, the context saving only condition tended to produce only general tips for creating the itinerary. As another example, E1 mentioned that the email draft generated with context selection was more personalized than that created without it. E1 stated, *"I like the recommendation letter request email draft it gives me, as it considers much of my background that I saved in the previous 'update your CV' task. I did the same thing in the previous round [context dumping] but did not feel it was as effective."*

In addition to context selection, context elicitation improves the quality of both subtasks and working solutions, outperforming both *context dumping* and *context filtering* conditions. Participants reported that the elicitation questions posed at the beginning *"provided the right plan to start with."* (E4) For instance, E2 uploaded the fellowship requirements document as initially suggested by the system, later rating the generated subtasks a perfect 7/7. *"It captured the requirements quite accurately, unlike what it gave me just now [with context filtering],"* E2 noted. *"The subtasks were precise, fitting the unique aspects of the fellowship I am applying to, which requires only one recommendation letter, though typically more are needed."* Similarly, elicitation questions about which state to obtain a driver's license in (for the "get a driver license" goal) and how many people are in the team (for the "organize a team event" goal) both resulted in subtasks that were better tailored to participants' individual situations. In addition, preference elicitation questions such as "What type of vehicle do you intend to drive?" and "When do you prefer to hold the event? Weekday or weekend? Noon or night?" prompted participants to provide answers as personalized context, ultimately resulting in more tailored working solutions that they rated highly.

5 User Study

To understand how JumpStarter might assist users in getting started on their personal goals, we conducted a within-subjects study with ten participants, comparing JumpStarter to ChatGPT. We chose ChatGPT as the baseline because it is a popular AI tool that many people are familiar with, and it allows users to feasibly add contextual information throughout the chat session.

During the study, each participant was asked to explore one personal goal they would like to achieve in the near future using each system. After finishing the exploration with each system, participants were asked to evaluate the system through questionnaires on task load, outcome satisfaction, and confidence level in taking the next steps on their goal. We also conducted semi-structured interviews to understand their experiences with each system. At the end of each study session, we conducted another interview to get feedback about their overall experience and system preferences.

5.1 Hypotheses

In the study, we investigate the following hypotheses:

- *H1* Compared to ChatGPT, JumpStarter significantly **lowers users' task load (H1)** for mental demand (H1a), temporal demand (H1b), performance (H1c), effort (H1d) and frustration (H1e).
- *H2* Compared to ChatGPT, JumpStarter significantly **increases users' exploration efficiency (H2)** in terms of the number of plan items explored (H2a) and the number of working solutions drafted (H2b) within the given time.
- *H3* Compared to ChatGPT, JumpStarter significantly **increases users' satisfaction level with the quality of results (H3)** in terms of plan quality (H3a) and working solution quality (H3b).
- *H4* Compared to ChatGPT, JumpStarter significantly **increases users' confidence in taking the next steps on their personal projects (H4)**.

5.2 Participants and Procedure

We recruited ten participants (average age=23.8, six female, four male) through a university mailing list and word of mouth. Each participant reported that they are familiar or very familiar with the use of ChatGPT. Before the study, we asked each participant to pick one personal goal they wanted to achieve in the next six months. The personal goals that each participant picked can be found in Table 4.

Participant	Personal goal	Goal type
P1	Start a side job	Career
P2	Organize a weekly game night	Life
P3	Land a job offer	Career
P4	Prepare for the LSAT	Academia
P5	Manage social media accounts	Creativity
P6	Move to a new apartment	Life
P7	Create a portfolio website	Creativity
P8	Prepare to deliver a tutorial	Academia
P9	Start a YouTube channel	Creativity
P10	Organize a family reunion	Life

Table 4: Overview of personal goals picked by participants in the user study.

During the session, we first introduced participants to the concept of action plans and working solutions with examples. Then, each participant was asked to create an action plan for their goal and draft working solutions for as many plan items as possible, using ChatGPT and JumpStarter. Participants were randomly assigned to a condition order (ChatGPT first and then JumpStarter, or JumpStarter first and then ChatGPT) that was counterbalanced to prevent a learning effect. Before participants started using JumpStarter, we used an example case to guide them through the system and familiarize them with the interface. Then, participants were given a maximum of 25 minutes to complete each task. They also had access to a Google Doc to note their results if needed. The entire process took approximately 1.5 hours. Participants were compensated $20/hour.

5.3 Results and Findings

We collected participants' ratings on a 1-7 point scale through questionnaires (see Table 5 for results). Participants rated their task load, outcome satisfaction, and confidence level in taking the next steps on their goal by using JumpStarter and ChatGPT. We recorded the number of plan items explored and the working solution drafted by each participant in each task. During the interviews, we asked them follow-up questions to understand the reasons behind their scores. We applied the thematic analysis method [6] to analyze the interview transcripts. We report the key findings in this section.

5.3.1 Task load. In the NASA TLX dimensions [21], working with JumpStarter was significantly less demanding in terms of mental demand (p=.005, Z=-2.52), temporal demand (p=.012, Z=-2.26), performance (p=.007, Z=-2.47), effort (p=.007, Z=-2.45), and frustration (p=.009, Z=-2.34).

JumpStarter reduces the mental load by providing an easier way for users to visualize the structure and track progress, unlike ChatGPT's linear chat interface. For example, P2 commented, *"ChatGPT info dumps a lot, and I have to keep the structure in my brain, whereas JumpStarter gave me a structure that I could easily follow."* Additionally, JumpStarter eases users' mental load by posing targeted questions to help them specify their context. For instance, P10 noted, *"I appreciate the questions the system (JumpStarter) asked when I felt stuck about how to iterate on drafting events [for a family reunion]. Entering the ages of the family members actually provided many better choices."* In contrast, in ChatGPT, users must devise the context themselves and frame the prompts accordingly, as P8 said, *"In ChatGPT, the information load is high—I have to think very hard about what information I should provide in order to get things that work for me."*

Moreover, participants reported putting significantly more effort into the task while using ChatGPT, although this did not necessarily result in better performance. For example, feeling that the current response was too general and niche, P1 asked ChatGPT to generate questions they could answer to improve the response. ChatGPT then provided eight questions, which were, according to P1, *"hard to answer as there were many and they were a bit too abstract."* However, even after P1 made lots of effort to answer all eight questions, they still felt that the responses were *"a bit too general and not useful."* Additionally, P1 noted that *"ChatGPT seemed to forget these eight answers soon after in my following chat with it,"* leaving them feeling that their efforts were not valued as ChatGPT failed to maintain and use this context.

5.3.2 Exploration efficiency. During the study session, participants explored significantly more plan items using JumpStarter (mean=4.5, SD=1.43) than they did with ChatGPT (mean=2.9, SD=0.74), and they drafted significantly more working solutions with JumpStarter (mean=5.3, SD=1.89) compared to ChatGPT (mean=3.3, SD=1.16).

In ChatGPT, it can be easy for users to become fixated on a single plan item, whereas JumpStarter helps users maintain an overview of the entire plan. For example, P2 went through nine iterations of the survey questions for organizing the weekly game night, stating, *"I got the survey questions I like in the end [after using ChatGPT], but it took me nine iterations on my own. I really hoped ChatGPT would guide me to the right place, but I had to direct myself. Because I got too involved in this survey thing, I totally forgot I needed to work on other planning items."* Similarly, P8 spent all the time digging into details like designing the visual content for tutorial slides. They reflected *"[With ChatGPT,] I went into a detailed level that was not necessary very quickly, and I just forgot to do the general planning."*

Furthermore, JumpStarter's structure enables users to jump between tasks and focus on one manageable task at a time, making it easier for them to arrive at a working solution. P4 summarized it well: *"I feel JumpStarter has a more flexible structure compared to ChatGPT. I like that I can easily jump between tasks and pick the ones I care about the most. Also, I like the visualization for each task—seeing the title, description, and context provides everything I need and helps me focus on exploring that task."* P10 echoed this sentiment, commenting, *"JumpStarter automatically manages and considers my drafts of previous tasks, which is good because then I can focus on the current task."*

5.3.3 Perceived quality of outcome. Plans created with JumpStarter were perceived to be of significantly higher quality (mean=5.9, SD=0.88) compared to those created with ChatGPT (mean=3.3, SD=1.49). Moreover, the working solution quality was also perceived to be significantly higher with JumpStarter (mean=5.8, SD=0.92) compared to ChatGPT (mean=4.7, SD=1.25), based on participants' self-reported results.

JumpStarter often poses important questions to clarify users' needs and incorporates that information into its planning. While ChatGPT tends to provide more general plans that might not align

Category	Factor	JumpStarter		ChatGPT		Statistics		Hypotheses
		Mean	SD	Mean	SD	p	Sig.	
Task load	Mental demand	3.3	1.64	5.3	1.06	.005	**	*H1a* accepted
	Temporal demand	2.4	.97	4.7	1.83	.012	*	*H1b* accepted
	Performance	2.7	.95	4.6	1.26	.007	**	*H1c* accepted
	Effort	2.8	1.69	4.8	1.23	.007	**	*H1d* accepted
	Frustration	2.1	1.60	4.5	1.90	.009	**	*H1e* accepted
Exploration efficiency	Plan items explored	4.5	1.43	2.9	.74	.002	**	*H2a* accepted
	Working solutions drafted	5.3	1.89	3.3	1.16	.047	*	*H2b* accepted
Satisfaction	Plan quality	5.9	.88	3.3	1.49	.004	**	*H3a* accepted
	Working solution quality	5.8	.92	4.7	1.25	.017	*	*H3b* accepted
Confidence		5.6	1.07	3.9	.99	.007	**	*H4* accepted

Table 5: The statistical test results comparing JumpStarter with ChatGPT, where the p-values (−: $p > .100$, +: $.050 < p < .100$, $*$: $p < .050$, $**$: $p < .010$, $***$: $p < .001$) are reported.

with specific user situations. For example, P9, who wanted to start their YouTube channel, shared, *"Identifying the target audience was not listed while I was using ChatGPT—I had to ask about it separately. Meanwhile, in JumpStarter, I was asked at the very beginning if I already knew my target audience. I said I needed help, and then it listed identifying the target audience as the first step in my plan, which was nice."* In addition, P9 mentioned that the plan generated with ChatGPT focused heavily on video editing, but as a video editing expert, they did not require much preparation for that task. Instead, *"JumpStarter asked about my familiarity with video editing at the very beginning, and I told them I was an expert. I feel that this was reflected well in the plan—like it assigned a shorter duration for the video editing task, which better suited my situation."*

5.3.4 Confidence level in taking the next steps on the goal. Participants reported significantly higher confidence in taking the next steps on their personal projects using JumpStarter (mean=5.6, SD=1.07) compared to using ChatGPT (mean=3.9, SD=.99).

Participants reported that ChatGPT helped validate their thoughts with commonsense knowledge and sometimes provided surprising or useful tips. As P7 said, *"Sitting down and planning things out itself is very helpful. I used ChatGPT as a cross-reference, checking to make sure I'm on the right track - thinking similarly to other people. And sometimes answers to low-level tasks covered things I did not really know, which is good."* However, they reported that they still always felt they might miss something important while using ChatGPT, whereas with JumpStarter, they feel more secure (P3, P4, P9), as P4 commented, *"I love that I can break things down further if I want, so I don't feel like I miss anything."*

In addition, JumpStarter can provide very personalized and actionable next steps that greatly increase users' confidence in taking action on their projects. For example, P1, with the goal of starting a side job, shared that *"The schedule JumpStarter helped me generate is very personalized, and I can directly use it to take real action - before, I felt worried about launching this idea as I had very limited time, now I feel like I can really start doing it."* P10 also liked that JumpStarter provided them with a specific and personalized itinerary for organizing the family reunion - *"I like that it summarizes everything I saved in the previous tasks - I can use it in the real world."*

5.3.5 Tool preference. 8 out of 10 participants reported they prefer to use our tool in the future, compared to ChatGPT. The main reasons given include that JumpStarter can provide more customized responses with less cognitive load. The users feel that they do not have to think hard about what information to provide (P1), are guided by the system (P2), can more easily consume the information (P3) or track the plan (P6), and can get their personal details efficiently organized, framed, and utilized (P8, P9, P10).

The other two participants (P4 and P5) mentioned that their choice depended on how familiar they were with the project they wanted to work on. If it was a topic they already had a clear understanding of, they preferred the chatbot interaction to help them figure out the details. Otherwise, they would prefer to use JumpStarter as it offers more structure. We further our discussion in terms of choices between structure and chat in Section 6.2.

5.3.6 Improvement feedback. Participants also provided insights on how to improve JumpStarter. The main feedback included "make the subtask outline and task descriptions editable" (P1, P3, P6, P7, P8, P9), "format the suggested working solution to be easier to read" (P4, P8, P9), "enable users to add or edit context whenever they want" (P3, P7), and "add a synthesis button to summarize what has been explored so far" (P1, P2). P8 also suggested embedding a search agent to collect data and ensure credibility. We discuss limitations and future work further in Section 6.3.

6 Discussion

6.1 AI-powered Context Curation in Complex Problem Solving

In JumpStarter, we studied how context curation aids people in exploring complex goals. Our studies showed that context curation is essential for devising plans and working solutions of high quality. AI-driven context curation significantly reduces users' mental burden and enhances their exploration efficiency. Context curation in

JumpStarter, besides the necessary context saving, also incorporates two key components: context elicitation and context selection.

Users greatly appreciated the elicitation questions posted by the system. Compared to the “pull” mode in ChatGPT, where users had to identify all requirements and constraints themselves to formulate a prompt, they tend to prefer the "push" mode in JumpStarter. In this mode, they can review questions posted by JumpStarter and select which ones to answer. This preference aligns with findings in cognitive psychology literature, suggesting that recalling information is more cognitively demanding than recognizing it [11]. Additionally, when users invest more effort into creating the prompt themselves, as observed in ChatGPT, they are more likely to be critical of the responses they receive. This phenomenon exemplifies the expectation confirmation theory [38], which suggests that dissatisfaction arises when results do not meet the standards anticipated based on the effort invested.

Users also appreciated the system assisting them in selecting context from the pool and transparently showing it before drafting a solution, giving them a good sense of control. However, it is worth noting that a few participants mentioned their desire for the system to remember and use all the context they have provided if possible. Ideally, tools like ChatGPT would maintain and use a memory of all user-provided context throughout the chat session [41]. Our observations of users’ interactions with ChatGPT, however, often revealed that it did not effectively use the context provided to deliver more personalized responses, or even worse, it sometimes provided conflicting responses to the user. This presents an interesting challenge in updating context accordingly—as the user progresses in their problem-solving journey, their answers for some context, such as the amount of time they want to allocate each week for LSAT preparation, might change as they realize the extent of content they need to cover. In such cases, it is crucial for the system to update the user preferences in the context pool, which we aim to explore in the future.

Last but not least, a major limitation of our system is that all context curation must occur within the system. Users are still required to manually upload files or input information. In line with earlier explorations [25, 27], it would be more ideal if the system could perform more “in-situ” context capturing, integrating content from users’ everyday tools like email inboxes and desktop files. An add-on that integrates into users’ existing workflows would be ideal, as suggested by the theory of invisible technology [4]. One can also imagine that such an add-on could directly place the email drafts it suggests into the email application as editable drafts, ready to be refined and sent.

6.2 Blending Chat and Structure for Interacting with LLMs

Since the advent of ChatGPT, chatbots have become a mainstream method for interacting with LLMs. However, our user study indicates that a structured approach is beneficial for handling more complex tasks. This is in line with findings from recent HCI work [35], which shows that structure aids in generating structured thoughts and personalized responses. Furthermore, a structured format helps users to more easily understand the information provided by GPT, avoiding the overwhelming experience of endless scrolling, as suggested in recent studies [22, 44]. The subtask tree structure in our system additionally aids users in grasping the overall scope of tasks, tracking their progress, managing context, and focusing on one individual task at a time.

However, does this mean that we should always prioritize structure over chat when using LLMs for complex problem-solving? During our studies with JumpStarter, we observed that users show an interest in engaging with a chatbot at certain stages of their exploration journey. For example, when users encounter unfamiliar terms such as “research proposal” or “back-end coding” in the plan or working solutions, they prefer to consult a context-aware chatbot to understand these terms. They find the chatbot feels more natural and allows them to quickly ask follow-up questions if anything is unclear. Similarly, when users have strong pre-existing opinions about their plans, they tend to favor the chatbot for its interactivity and immediacy. This allows them to quickly receive feedback that aligns with their own thinking flow. For instance, if they have concerns about aspects such as the market size for their side job product ideas, they can immediately drill into that with the chatbot’s assistance.

In addition, our system currently uses a “form fill-in” method for users to answer the elicitation questions. Although this structured approach helps the system efficiently gather and save the context, implementing it in a chat interaction might be more natural and friendly. One everyday metaphor is when a doctor wants to learn more about your health condition, they can either give you a long form to fill in or ask in a conversational way that might make you feel more comfortable. They can follow up immediately if you find some questions unclear or if the doctor finds your answers not making sense. However, structured interaction has many benefits that chatbot interaction does not provide, as we discussed in our study findings. It is an interesting future direction to explore how we might blend structure and interaction together to complement their pros and cons and provide a better way to interact with LLMs.

6.3 Limitations and Future Work

JumpStarter primarily assists users in figuring out “how” to achieve their personal goals, prompting real-world actions by increasing simplicity, as suggested by the Fogg Behavior Model [16]. However, the “why” problem—motivation (such as self-regulation or emotional challenges)—is another very important dimension that JumpStarter does not address. For example, one participant shared that their confidence dropped after exploring the goal, as they realized how much they had to do to make it happen. Emotional support at this time would be very valuable. But as prior studies suggested [5], we should be very careful about using LLMs to provide emotional support, which deserves further investigation.

JumpStarter was designed to support people with personal goals that primarily involve cognitive or knowledge work, encompassing a wide range of goals across academia, career, creativity, and life (see examples in Table 4). However, there are many other personal goals that JumpStarter does not cover, such as physical goals (e.g., losing weight), behavioral goals (e.g., overcoming shyness), and spiritual goals (e.g., coming to terms with one’s faith), as outlined in [32]. Many of these goals require motivational support, a topic explored in previous HCI works [10, 15]. Future iterations of JumpStarter could potentially expand to include these areas to accommodate a broader spectrum of personal goals.

JumpStarter utilizes GPT-4 as its core engine for providing information. While LLMs like GPT-4 are adept at synthesizing information from the Internet and can sometimes offer valid and useful references, they are also prone to generating inaccurate or hallucinated information [1]. Hence, it is crucial to integrate search agents into our system to enhance the credibility of the information provided. Also, when pursuing creative goals such as starting a YouTube channel, users require support in collecting and analyzing real-world data. Developing a search agent specifically tailored to platforms like YouTube could be a valuable direction to explore.

JumpStarter currently requires users to manually add context to the system. As discussed in Section 6.1, adopting a richer and more flexible method for capturing context that integrates into users' current workflows would be beneficial. Additionally, JumpStarter currently can only handle context in text form, but many context are not in text, such as YouTube channel icons or slide visual layouts. As visual understanding models like GPT-4V [1] evolve, it would be interesting to expand JumpStarter to handle multimodal context.

7 Conclusion

In this work, we present JumpStarter, a system designed to support people in getting started with their personal goals through AI-powered context curation. Our technical evaluation indicates that context curation plays a crucial role in generating plans and working solutions of higher quality. Our user study reveals that, compared to ChatGPT, JumpStarter enables users to explore their goals more efficiently and with less mental effort. We discuss the design implications of incorporating AI-powered context curation into complex problem-solving. We also provide insights into how we might blend structured and conversational approaches to better support people's interaction with LLMs.

References

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774* (2023).

[2] Icek Ajzen. 1991. The Theory of planned behavior. *Organizational Behavior and Human Decision Processes* (1991).

[3] Alan Baddeley. 1992. Working memory. *Science* 255, 5044 (1992), 556–559.

[4] Eric Bergman and Donald A Norman. 2000. Making Technology Invisible. *Information Appliances and Beyond: Interaction Design for Consumer Products* (2000).

[5] Ananya Bhattacharjee, Yuchen Zeng, Sarah Yi Xu, Dana Kulzhabayeva, Minyi Ma, Rachel Kornfield, Syed Ishtiaque Ahmed, Alex Mariakakis, Mary P Czerwinski, Anastasia Kuzminykh, et al. 2024. Understanding the Role of Large Language Models in Personalizing and Scaffolding Strategies to Combat Academic Procrastination. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*. 1–18.

[6] Virginia Braun and Victoria Clarke. 2006. Using thematic analysis in psychology. *Qualitative research in psychology* 3, 2 (2006), 77–101.

[7] Carrie J. Cai, Philip J. Guo, James Glass, and Robert C. Miller. 2014. Wait-learning: leveraging conversational dead time for second language education. In *CHI '14 Extended Abstracts on Human Factors in Computing Systems* (Toronto, Ontario, Canada) *(CHI EA '14)*. Association for Computing Machinery, New York, NY, USA, 2239–2244. https://doi.org/10.1145/2559206.2581183

[8] Shouyuan Chen, Sherman Wong, Liangjian Chen, and Yuandong Tian. 2023. Extending context window of large language models via positional interpolation. *arXiv preprint arXiv:2306.15595* (2023).

[9] Kenneth Conley and James Carpenter. 2007. Towel: Towards an Intelligent To-Do List.. In *AAAI Spring Symposium: Interaction Challenges for Intelligent Assistants*. 26–32.

[10] Sunny Consolvo, Predrag Klasnja, David W McDonald, and James A Landay. 2009. Goal-setting considerations for persuasive technologies that encourage physical activity. In *Proceedings of the 4th international Conference on Persuasive Technology*. 1–8.

[11] Fergus IM Craik and Robert S Lockhart. 1972. Levels of processing: A framework for memory research. *Journal of verbal learning and verbal behavior* 11, 6 (1972), 671–684.

[12] Allen Cypher. 1986. The structure of users' activities. *User centered system design* (1986), 243–263.

[13] Edward L Deci and Richard M Ryan. 2012. Self-determination theory. *Handbook of theories of social psychology* 1, 20 (2012), 416–436.

[14] Adele Diamond. 2013. Executive functions. *Annual review of psychology* 64, 1 (2013), 135–168.

[15] Tina Ekhtiar, Armağan Karahanoğlu, Rúben Gouveia, and Geke Ludden. 2023. Goals for goal setting: a scoping review on personal informatics. In *Proceedings of the 2023 ACM Designing Interactive Systems Conference*. 2625–2641.

[16] Brian J Fogg. 2009. A behavior model for persuasive design. In *Proceedings of the 4th international Conference on Persuasive Technology*. 1–7.

[17] Peter Gollwitzer. 1996. The volitional benefits of planning. In *The psychology of action: Linking cognition and motivation to behavior*. Guilford Press, 287–312.

[18] Peter M Gollwitzer. 1990. Action phases and mind-sets. *Handbook of motivation and cognition: Foundations of social behavior* 2, 53-92 (1990), 2.

[19] Peter M Gollwitzer. 1999. Implementation intentions: strong effects of simple plans. *American psychologist* 54, 7 (1999), 493.

[20] Kunal Handa, Yarin Gal, Ellie Pavlick, Noah Goodman, Jacob Andreas, Alex Tamkin, and Belinda Z Li. 2024. Bayesian preference elicitation with language models. *arXiv preprint arXiv:2403.05534* (2024).

[21] Sandra G Hart. 2006. NASA-task load index (NASA-TLX); 20 years later. In *Proceedings of the human factors and ergonomics society annual meeting*, Vol. 50. Sage publications Sage CA: Los Angeles, CA, 904–908.

[22] Peiling Jiang, Jude Rayan, Steven P Dow, and Haijun Xia. 2023. Graphologue: Exploring large language model responses with interactive diagrams. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*. 1–20.

[23] Philip Nicholas Johnson-Laird. 1983. *Mental models: Towards a cognitive science of language, inference, and consciousness*. Number 6. Harvard University Press.

[24] William Jones, Predrag Klasnja, Andrea Civan, and Michael L Adcock. 2008. The personal project planner: Planning to organize personal information. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 681–684.

[25] Victor Kaptelinin. 2003. UMEA: translating interaction histories into project contexts. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. 353–360.

[26] Harmanpreet Kaur, Alex C Williams, Anne Loomis Thompson, Walter S Lasecki, Shamsi T Iqbal, and Jaime Teevan. 2018. Creating better action plans for writing tasks via vocabulary-based planning. *Proceedings of the ACM on Human-Computer Interaction* 2, CSCW (2018), 1–22.

[27] Mik Kersten and Gail C Murphy. 2006. Using task context to improve programmer productivity. In *Proceedings of the 14th ACM SIGSOFT international symposium on Foundations of software engineering*. 1–11.

[28] Nicolas Kokkalis, Thomas Köhn, Johannes Huebner, Moontae Lee, Florian Schulze, and Scott R Klemmer. 2013. Taskgenies: Automatically providing action plans helps people complete tasks. *ACM Transactions on Computer-Human Interaction (TOCHI)* 20, 5 (2013), 1–25.

[29] Michel Krieger, Emily Margarete Stark, and Scott R Klemmer. 2009. Coordinating tasks on the commons: designing for personal goals, expertise and serendipity. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 1485–1494.

[30] Yash Kumar Lal, Li Zhang, Faeze Brahman, Bodhisattwa Prasad Majumder, Peter Clark, and Niket Tandon. 2024. Tailoring with Targeted Precision: Edit-Based Agents for Open-Domain Procedure Customization. In *Findings of the Association for Computational Linguistics ACL 2024*. 15597–15611.

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

[32] Brian R Little. 1983. Personal projects: A rationale and method for investigation. *Environment and behavior* 15, 3 (1983), 273–309.

[33] Chengzhong Liu, Zeyu Huang, Dingdong Liu, Shixu Zhou, Zhenhui Peng, and Xiaojuan Ma. 2022. PlanHelper: Supporting Activity Plan Construction with Answer Posts in Community-Based QA Platforms. *Proceedings of the ACM on Human-Computer Interaction* 6, CSCW2 (2022), 1–26.

[34] Edwin A Locke and Gary P Latham. 2002. Building a practically useful theory of goal setting and task motivation: A 35-year odyssey. *American psychologist* 57, 9 (2002), 705.

[35] Xiao Ma, Swaroop Mishra, Ariel Liu, Sophie Ying Su, Jilin Chen, Chinmay Kulkarni, Heng-Tze Cheng, Quoc Le, and Ed Chi. 2024. Beyond chatbots: Explorellm for structured thoughts and personalized model responses. In *Extended Abstracts of the CHI Conference on Human Factors in Computing Systems*. 1–12.

[36] James E Maddux. 1995. Self-efficacy theory: An introduction. In *Self-efficacy, adaptation, and adjustment: Theory, research, and application*. Springer, 3–33.

[37] Karen Myers, Pauline Berry, Jim Blythe, Ken Conley, Melinda Gervasio, Deborah L McGuinness, David Morley, Avi Pfeffer, Martha Pollack, and Milind Tambe. 2007. An intelligent personal assistant for task and time management. *AI Magazine* 28, 2 (2007), 47–47.

[38] Richard L Oliver. 1980. A cognitive model of the antecedents and consequences of satisfaction decisions. *Journal of marketing research* 17, 4 (1980), 460–469.

[39] Bowen Peng, Jeffrey Quesnelle, Honglu Fan, and Enrico Shippole. 2023. Yarn: Efficient context window extension of large language models. *arXiv preprint arXiv:2309.00071* (2023).

[40] Sajjadur Rahman, Pao Siangliulue, and Adam Marcus. 2020. Mixtape: Mixed-initiative team action plan creation through semi-structured notes, automatic task generation, and task classification. *Proceedings of the ACM on Human-Computer Interaction* 4, CSCW2 (2020), 1–26.

[41] Partha Pratim Ray. 2023. ChatGPT: A comprehensive review on background, applications, key challenges, bias, ethics, limitations and future scope. *Internet of Things and Cyber-Physical Systems* 3 (2023), 121–154.

[42] Maria A Rickers-Ovsiankina. 1928. *Die wiederaufnahme unterbrochener handlungen*. Ph. D. Dissertation. Verlagsbuchhandlung Julius Springer.

[43] Paschal Sheeran and Thomas L Webb. 2016. The intention–behavior gap. *Social and personality psychology compass* 10, 9 (2016), 503–518.

[44] Sangho Suh, Bryan Min, Srishti Palani, and Haijun Xia. 2023. Sensecape: Enabling multilevel exploration and sensemaking with large language models. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*. 1–18.

[45] John Sweller. 2011. Cognitive load theory. In *Psychology of learning and motivation*. Vol. 55. Elsevier, 37–76.

[46] Jaime Teevan, Shamsi T Iqbal, Carrie J Cai, Jeffrey P Bigham, Michael S Bernstein, and Elizabeth M Gerber. 2016. Productivity decomposed: Getting big things done with little microtasks. In *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems*. 3500–3507.

[47] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems* 35 (2022), 24824–24837.

[48] Deena Skolnick Weisberg, Kathy Hirsh-Pasek, Roberta Michnick Golinkoff, and Bruce D McCandliss. 2014. Mise en place: Setting the stage for thought and action. *Trends in Cognitive Sciences* 18, 6 (2014), 276–278.

[49] Alex C Williams, Harmanpreet Kaur, Shamsi Iqbal, Ryen W White, Jaime Teevan, and Adam Fourney. 2019. Mercury: Empowering Programmers' Mobile Work Practices with Microproductivity. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology*. 81–94.

[50] Jian Xie, Kai Zhang, Jiangjie Chen, Tinghui Zhu, Renze Lou, Yuandong Tian, Yanghua Xiao, and Yu Su. 2024. Travelplanner: A benchmark for real-world planning with language agents. *arXiv preprint arXiv:2402.01622* (2024).

[51] Howard Yen, Tianyu Gao, and Danqi Chen. 2024. Long-context language modeling with parallel context encoding. *arXiv preprint arXiv:2402.16617* (2024).

[52] Huaixiu Steven Zheng, Swaroop Mishra, Hugh Zhang, Xinyun Chen, Minmin Chen, Azade Nova, Le Hou, Heng-Tze Cheng, Quoc V Le, Ed H Chi, et al. 2024. NATURAL PLAN: Benchmarking LLMs on Natural Language Planning. *arXiv preprint arXiv:2406.04520* (2024).

A GPT-4 Prompts used in JumpStarter

A.1 Context Elicitation for the Goal

See Figure 7.

System prompt

You are a helpful assistant in generating at most three questions to elicit more context from the user in order to accomplish a task the user is involved. You will be given the main purpose of the task, and the current context history from the user. You need to judge if the current context information is enough to do this task. If not, what existing doc do you think they already have and can be provided as the context document for this task? Alternatively, what existing most important extra information do you think they should provide as the context information for this task? Please try your best to start by asking the potential existing doc first, and then ask for the potential info. Do not ask the question that can be possibly answered from the suggested doc in the first question. You can also just ask for the potential info if no doc is needed from the user. Please directly generate the questions for users to answer, provide the reason for the question, identify if it is DOC or INFO, and provide a name for the question. Otherwise, output "Ready".

Here are two examples:
Input: My user has a main purpose: Apply for a drivers license by 12/15/2024. The current context history from the user is empty.
Output:
1. Reason: Driver's license requirements vary significantly depending on the location. Knowing the specific state or country would allow for tailored advice regarding local rules, tests, and documentation required -> Question: Which state or country are you applying for? -> Type: INFO -> Name: State or Country
2. Reason: Different age groups may have different requirements or steps in the licensing process. For example, minors often have to go through graduated license programs. -> Question: How old are you -> Type: INFO -> Name: Age of the User

Input: My user has a main purpose: Apply for a PhD program by 12/15/2024. The current context history from the user is empty.
Output:
1. Reason: Curriculum Vitae (CV) or Resume could be helpful as it would likely contain detailed information about their educational background and any research experiences or academic achievements, which are critical for applying to PhD programs. -> Question: What CV or Resume can you provide for the PhD application? -> Type: DOC -> Name: CV or Resume

Now, start prediction:

User Prompt

Input: My user has a main purpose: {task_input}. The current context history from the user is empty.

Figure 7: The prompt for Context Elicitation for the goal.

A.2 Context Elicitation for Drafting Working Solutions

See Figure 8.

My user has a main purpose: {main_purpose}. My user is working on the task {task_name}: {task_description}. The current context history from the user is {context_history}. Please judge if the current context information is enough to do this task. If not, what existing most important extra information do you think they should provide as the context information for this task? Please directly generate questions for users to answer as extra info. Otherwise, output "Ready"
Format the response like this: 1. <question 1> : <reason for asking question 1> -> title of question 1
2. <question 2> : <reason for asking question 2> -> title of question 2
3. <question 3> : <reason for asking question 3> -> title of question 3

Figure 8: The prompt for Context Elicitation for drafting working solutions.

A.3 Context Selection

A.3.1 Context Selection for Drafting. See Figure 9.

A.3.2 Context Selection for Forking. The prompt is as follows:

- *My user has a main purpose: {main_purpose}. My user is working on the task {task_name}: {task_description}. My user needs to break down the task into sub-tasks. Here is the current context history from the user: {context_history}. Please select the most relevant context key from the current context history that can be used to better decompose the current task into several sub-tasks for the user to get started. Do not help the user to break down the task. Please also provide explanations. Format the response like this: <context_key>: <reasons>. Replace the context_key with the actual key in the context history.*

System prompt

Given the user's main purpose and the task they are working on, select the most relevant context keys from the current context history that can be used to draft good responses for the user to complete the task. Please also provide explanations. \n The current context history is shown as one or more key-value pairs. Please select only the keys from the 'key' part of the context history. Do not select the keys from the 'value' part of the context history \n Format the response like this: number. <context_keys>: <reasons for selecting context_keys>. Replace the context_keys with the actual keys as shown in the context history. Please directly give the answers and do not provide extra summarization sentences at the end.

User Prompt

My user has a main purpose: {main_purpose}. My user is working on the task {task_name}: {task_description}. Here is the current context history in JSON format (with 'key':'value' pairs) from the user: {context_history}

Figure 9: The prompt for Context Selection for drafting working solutions.

A.4 Subtask Suggestion

See Figure 10.

My user has a main purpose:{main purpose}.
Please consider the following context information from my user: {user_context}

Please break down the task below into three to six manageable subtasks: {current task}
The existing step structure is shown as follows: {existing tree step structure}.

Please directly give the response that fills in the current subtask: {current task} in the provided task structure.

Format the response like this: 1. [Duration for subtask1] {subtask1 title}: {subtask1 detailed description} 2. [Duration for subtask2] {subtask2 title}: {subtask2 detailed description} 3. [Duration for subtask3] {subtask3 title}: {subtask3 detailed description}.
Please specify the duration for each subtask in terms of days, weeks or months. For example, [1 week], [2-4 weeks], [1 month], and [1-2 months]. Please do not include other texts for duration such as [Ongoing]. Please do not include ** in the subtask title. Please directly give the response and do not start with "{current subtask title}:"

Figure 10: The prompt for Subtask Suggestion.

A.5 Subtask Detection

See Figure 11.

A.6 Fork Detection

See Figure 12.

A.7 Working Solution Drafts

The prompt to generate the working draft is shown as follows:

- *My user has a main purpose: {main purpose}. Please consider the following context information from my user: {user_context}. My user needs help with the current task {current task}: {task description}*

System prompt

You are a useful assistance to detect if the current task needs to be further decomposed if it is not actionable and the primary goal of the task can not be viewed as a singular, distinctive deliverable. Specifically, an overall goal is decomposed into a tree of subtasks. Each of these tasks is positioned on the tree structure and is associated with a level. Based on the user prompt, and the current level of the task node on the tree, please output Yes if it needs to be decomposed; No otherwise meaning it is actionable and does not require task decomposition. Please also provide explanations for your choice.
Here are some examples:
User: My user is working on the task Research on Prospective Companies and Positions: Conduct a comprehensive search on potential companies and specific research scientist internship positions in the NLP field. Understand what each role entails, identify skill requirements, and evaluate how they align with your research interests. The current node level of the task is 1. My user needs to know if the current task is specific and actionable
Reason: this task needs to be further decomposed as it is positioned on the first level of the tree and it involves more than one deliverables: search on companies and search on positions. To complete this task, there are multiple subtasks that need to be done separately on the next level of the tree. These include conducting a comprehensive search on potential companies, searching for specific research scientist internship positions, an analysis of what each role entails, identifying skill requirements, and evaluation of alignment with the user's research interests.
Answer: Yes

User: My user is working on the task Identify Potential Universities: Create a list of universities that offer PhD programs in HCI. The selection can be based on factors such as reputation, HCI research focus, published HCI research papers, faculty expertise etc. The current node level of the task is 2. My user needs to know if the current task is specific and actionable.
Reason: This task does not need to be further decomposed as it is positioned on the second level of the tree and it just involves one deliverable: create a list of schools that offer PhD programs in HCI. Although it may require several steps to create the list, the end goal of this task is to get a list. Therefore the task is actionable.
Answer: No

User: My user is working on the task Identify Required Documents: Reseach and confirm all the necessary documents required for the non-driver ID application, ensuring to list all forms of acceptable proofs such as a birth certificate or passport for identity, Social Security Card or W-2 form for Social Security number, and utility bills or lease agreement for proof of residency. The current node level of the task is 1. My user needs to know if the current task is specific and actionable.
Reason: Although the task is positioned on the first level of the tree, the task does not need to be further decomposed as the primary goal of the task - identify necessary documents for the non-driver ID application - can be viewed as a singular, cohesive deliverable. Despite involving various types of documents, the task is focused on compiling a comprehensive list, which makes it actionable as a single unit. The distinction lies in the focus on gathering all necessary documentation, a clear and direct objective.
Answer: No

Now, let's start prediction:

User Prompt

User: My user is working on the task {task_title}: {task_descriptions}. The current node level of the task is {tree_level}. My user needs to know if the current task needs to be decomposed

Figure 11: The prompt for Subtask Detection.

System prompt

Given the queried task, determine if a "for" loop is needed to complete the task. You will be given a question Q. Please provide the reasoning and then respond with "Yes" or "No".
Here are some examples:
Q: Research the specific HCI PhD programs at each university from the initial list. Focus on aspects such as program curriculum, research opportunities, faculty expertise, and available resources.
Reason: The task requires a "for" loop to complete as there already exists an initial list of entities (i.e. universities) to research. Specifically, the goal of this task is to research the program curriculum, research opportunities, faculty expertise, and available resources for each university from the initial list. It is not possible to complete the task directly without a "for" loop.
A: Yes

Q: Make a list of potential recommenders including former supervisors, academic advisors, and professors who are familiar with your academic and research abilities.
Reason: The task does not require a "for" loop to complete as there does not exist a list of potential recommenders. The goal of this task therefore is to construct the list of recommenders based on certain criteria.
A: No

Q: Reach out to the individuals on your list via email or phone, providing them with the necessary documents and details about the HCI programs, and formally request their letters of recommendation
Reason: This task needs a "for" loop to complete as you have already obtained your list of individuals. You should reach out to each entity (i.e. individuals) on the list to complete the task.
A: Yes

Q: Gather information on different universities offering PhD programs in Human-Computer Interaction. Create an initial list based on general information such as program recognition, location, and basic offerings.
Reason: The task does not require a "for" loop to complete as there does not exist a list of universities offering PhD programs in HCI that can be used to iterate on.
A: No

User Prompt

Q: {task_description}

Figure 12: The prompt for Fork Detection.

B Prompts for Technical Evaluation of Subtask Detection

B.1 Zero-shot Prompting

The prompt for zero-shot for the task of subtask detection is demonstrated below:

- System prompt: *You are a useful assistance to detect if the current task needs to be further decomposed if it is not actionable and the primary goal of the task can not be viewed as a singular, distinctive deliverable. Based on the user prompt, please output Yes if it needs to be decomposed; No otherwise meaning it is actionable and does not require task decomposition.*
- User Prompt: *My user is working on the task {task title}: {task description}. My user needs to know if the current task needs to be decomposed.*

B.2 Few-shot Prompting

The prompt for few-shot-only prompting is shown in Figure 13. Note that we used three in-context examples in the prompt.

System prompt

You are a useful assistance to detect if the current task needs to be further decomposed if it is not actionable and the primary goal of the task can not be viewed as a singular, distinctive deliverable. Based on the user prompt, please output Yes if it needs to be decomposed; No otherwise meaning it is actionable and does not require task decomposition
Here are some examples:
User: My user is working on the task Research on Prospective Companies and Positions: Conduct a comprehensive search on potential companies and specific research scientist internship positions in the NLP field. Understand what each role entails, identify skill requirements, and evaluate how they align with your research interests. My user needs to know if the current task is specific and actionable
Answer: Yes

User: My user is working on the task Identify Potential Universities: Create a list of universities that offer PhD programs in HCI. The selection can be based on factors such as reputation, HCI research focus, published HCI research papers, faculty expertise etc. My user needs to know if the current task is specific and actionable.
Answer: No

User: My user is working on the task Identify Required Documents: Reseach and confirm all the necessary documents required for the non-driver ID application, ensuring to list all forms of acceptable proofs such as a birth certificate or passport for identity, Social Security Card or W-2 form for Social Security number, and utility bills or lease agreement for proof of residency. My user needs to know if the current task is specific and actionable.
Answer: No

Now, let's start prediction:

User prompt

My user is working on the task {task title}: {task description}. My user needs to know if the current task needs to be decomposed.

Figure 13: The few-shot-only prompt for Subtask Detection.

B.3 Few-shot + CoT

We constructed the prompt in a Chain-of-Thought fashion, where GPT-4 is instructed to first generate the reasoning and then the answer. The prompt is shown in Figure 14.

B.4 Few-shot + CoT + Draft

We experimented with incorporating both CoT and the initial working solution draft into the prompt. The system prompt is shown in Figure 15. For the user prompt, before detecting subtasks. we first generated the initial working solution draft for the current task. The user prompt is shown below:

- *My user is working on the task {task title}: {task description}. The GPT response to the task is: {Draft}. My user needs to know if the current task is specific and actionable.*

B.5 Few-shot + CoT + Tree + Draft

To construct the system prompt for this setting, we incorporate the tree level of each task into the prompt. The prompt is shown in Figure 16. Additionally, for the current task at hand, its tree-level information is also presented in the user prompt, as shown below:

System prompt

You are a useful assistance to detect if the current task needs to be further decomposed if it is not actionable and the primary goal of the task can not be viewed as a singular, distinctive deliverable. Based on the user prompt, and the GPT response, please output Yes if it needs to be decomposed; No otherwise meaning it is actionable and does not require task decomposition. Please also provide explanations for your choice.

Here are some examples:
User: My user is working on the task Research on Prospective Companies and Positions: Conduct a comprehensive search on potential companies and specific research scientist internship positions in the NLP field. Understand what each role entails, identify skill requirements, and evaluate how they align with your research interests. The GPT response to the task is:
Sure, I will list some promising companies that offer research scientist internship positions in Natural Language Processing (NLP) and briefly describe what each role entails:

1. **Google AI**:
- *Role*: As a research intern, you can work in areas such as machine intelligence, machine perception, natural language processing, and quantum computing. Google typically looks for research experience, ability to design and execute a research agenda, and coding skills.

2. **Facebook AI Research (FAIR)**:
- *Role*: Natural Language Processing Internship roles at FAIR involve developing novel techniques for NLP tasks which include generating human-like text. They require experience with machine learning, AI, deep learning, dialogue management, and computer vision.

3. **Amazon Alexa AI**:
- *Role*: Interns in this role generally work on language understanding capabilities of Alexa, advancing state-of-the-art on tasks like entity resolution or text classification. Knowledge in machine learning, especially deep learning for NLP, is key for this role.

4. **OpenAI**:
- *Role*: Research internships at OpenAI are highly selective. Interns are expected to have published in machine learning or AI and can spend a significant portion of their time on original research.

5. **IBM Watson**:
- *Role*: Watson Interns can work on various AI/NLP tasks from fact extraction, relation extraction, disambiguation, to summarization, based on the team you join. A strong AI/ML foundation and coding skills are required.

6. **Apple AI/ML**:
- *Role*: Apple offers internships where you can work on Siri's language understanding capabilities. They typically require machine learning knowledge and experience, especially in deep learning.

7. **Microsoft Research**:
- *Role*: Interns in this role could work on a variety of NLP challenges, including developing algorithms that enable, for example, conversational interactions with AI. Strong machine learning knowledge, particularly in deep learning, is expected.

8. **Baidu Research**:
- *Role*: Interns can work on various novel NLP tasks such as language understanding, language generation, and more. Baidu looks for individuals with robust knowledge in machine learning, deep learning, and coding abilities.

9. **NVIDIA - AI Research**:
- *Role*: Interns have an opportunity to work on projects in the realms of machine learning, deep learning, computer vision, and natural language processing. They seek candidates with strong ML and DL backgrounds.

When selecting a specific position, I recommend considering the following:
- Does the role align with your career objectives?
- Does the role play to your strengths or offer opportunities to develop in areas of interest?
- Does the role match your skill set? If not, can you acquire those skills?

Remember, a role can be an opportunity to learn and grow, so don't be deterred if you lack a few skills. If the role excites you, go for it!
My user needs to know if the current task is specific and actionable
Reason: this task needs to be further decomposed as the GPT response is not good enough to help user complete the task. It involves more than one deliverables: search on companies and search on positions. To complete this task, there are multiple subtasks that need to be done separately. These include conducting a comprehensive search on potential companies, searching for specific research scientist internship positions, an analysis of what each role entails, identifying skill requirements, and evaluation of alignment with the user's research interests.
Answer: Yes

User: My user is working on the task Identify Potential Universities: Create a list of universities that offer PhD programs in HCI. The selection can be based on factors such as reputation, HCI research focus, published HCI research papers, faculty expertise etc. The GPT response to the task is:
Here is a list of universities that offer PhD programs in HCI:
1. Stanford University, USA
2. Massachusetts Institute of Technology (MIT), USA
3. Carnegie Mellon University (CMU), USA
4. University of California, Berkeley, USA
5. University of Washington, USA
6. University of Michigan, USA
7. University of California, San Diego, USA
8. Cornell University, USA
9. University of Toronto, Canada
10. University of College London, UK
My user needs to know if the current task is specific and actionable.
Reason: The GPT response is good enough and it provides direct answer to complete the task. Additionally, this task does not need to be further decomposed as it just involves one deliverable: create a list of schools that offer PhD programs in HCI. Although it may require several steps to create the list, the end goal of this task is to get a list. Therefore the task is actionable.
Answer: No

User: My user is working on the task Identify Required Documents: Reseach and confirm all the necessary documents required for the non-driver ID application, ensuring to list all forms of acceptable proofs such as a birth certificate or passport for identity, Social Security Card or W-2 form for Social Security number, and utility bills or lease agreement for proof of residency. The GPT response to the task is:
To apply for a non-driver ID in [redacted] you need to provide several documents to prove your identity, Social Security Number (SSN), and residency. Below, I have listed the acceptable documents for each category based on the latest guidelines provided by the [redacted] Department of Motor Vehicles (DMV).

1. Proof of Identity and Date of Birth (must total 6 points according to [redacted] DMV's point system):
- U.S. passport or U.S. passport card (valid or expired within the past five years) – 4 points
- Birth certificate (U.S., Puerto Rican, Canadian, or Mexican) - 2 points
- Employment Authorization Card (I-766) – 3 points
- Permanent Resident Card (I-551) – 3 points
- Foreign passport with valid U.S. Visa and I-94 – 3 points

Note: At least one document must show your date of birth and at least one document must have your full Social Security Number.

2. Proof of Social Security Number:
- Social Security card
- W-2 form
- SSA-1099 form
- Non-SSA-1099 form
- Pay stub with your name and SSN on it

3. Proof of [redacted] Residency (at least two documents):
- State or federal tax return
- Utility bill (must be dated within the last 90 days)
- Lease agreement or mortgage statement
- Bank statement (must be dated within the last 90 days)
- Any mail from any government agency (dated within the last 90 days)

Additional Notes:
- All documents must be originals or certified copies.
- If your name has changed due to marriage, divorce, or court order, you will need to provide documents that link your current name to your former name.
- You must also complete an "Application for Permit, Driver License or Non-Driver ID Card" (Form MV-44).

Before going to the DMV, it's recommended to use the [redacted] DMV Document Guide which provides a personalized checklist of the required documents based on the specifics of your application. This can be found on the official [redacted] DMV website.

Make sure to double-check with the DMV for the latest requirements or any specific circumstances that may affect your application to ensure a smooth process.
My user needs to know if the current task is specific and actionable.
Reason: The task does not need to be further decomposed for two reasons. The first reason is that the GPT response provides direct answer to the task and generates a list of all forms of acceptable proofs for the documents required to get a non-driver ID. The second reason is that the primary goal of the task - identify necessary documents for the non-driver ID application - can be viewed as a singular, cohesive deliverable. Despite involving various types of documents, the task is focused on compiling a comprehensive list, which makes it actionable as a single unit. The distinction lies in the focus on gathering all necessary documentation, a clear and direct objective.
Answer: No

Now, let's start prediction:

Figure 15: The prompt for few-shot + CoT + Draft for Subtask Detection.

You are a useful assistance to detect if the current task needs to be further decomposed if it is not actionable and the primary goal of the task can not be viewed as a singular, distinctive deliverable. Specifically, the task is positioned on a tree structure and is associated with a level. Based on the user prompt, the current node level of the task on the tree, and the GPT response, please output Yes if it needs to be decomposed; No otherwise meaning it is actionable and does not require task decomposition. Please also provide explanations for your choice.

Here are some examples:
User: My user is working on the task Research on Prospective Companies and Positions: Conduct a comprehensive search on potential companies and specific research scientist internship positions in the NLP field. Understand what each role entails, identify skill requirements, and evaluate how they align with your research interests. The current node level of the task is 1. The GPT response to the task is:
Sure, I will list some promising companies that offer research scientist internship positions in Natural Language Processing (NLP) and briefly describe what each role entails:

1. **Google AI**:
- *Role*: As a research intern, you can work in areas such as machine intelligence, machine perception, natural language processing, and quantum computing. Google typically looks for research experience, ability to design and execute a research agenda, and coding skills.

2. **Facebook AI Research (FAIR)**:
- *Role*: Natural Language Processing Internship roles at FAIR involve developing novel techniques for NLP tasks which include generating human-like text. They require experience with machine learning, AI, deep learning, dialogue management, and computer vision.

3. **Amazon Alexa AI**:
- *Role*: Interns in this role generally work on language understanding capabilities of Alexa, advancing state-of-the-art on tasks like entity resolution or text classification. Knowledge in machine learning, especially deep learning for NLP, is key for this role.

4. **OpenAI**:
- *Role*: Research internships at OpenAI are highly selective. Interns are expected to have published in machine learning or AI and can spend a significant portion of their time on original research.

5. **IBM Watson**:
- *Role*: Watson Interns can work on various AI/NLP tasks from fact extraction, relation extraction, disambiguation, to summarization, based on the team you join. A strong AI/ML foundation and coding skills are required.

6. **Apple AI/ML**:
- *Role*: Apple offers internships where you can work on Siri's language understanding capabilities. They typically require machine learning knowledge and experience, especially in deep learning.

7. **Microsoft Research**:
- *Role*: Interns in this role could work on a variety of NLP challenges, including developing algorithms that enable, for example, conversational interactions with AI. Strong machine learning knowledge, particularly in deep learning, is expected.

8. **Baidu Research**:
- *Role*: Interns can work on various novel NLP tasks such as language understanding, language generation, and more. Baidu looks for individuals with robust knowledge in machine learning, deep learning, and coding abilities.

9. **NVIDIA - AI Research**:
- *Role*: Interns have an opportunity to work on projects in the realms of machine learning, deep learning, computer vision, and natural language processing. They seek candidates with strong ML and DL backgrounds.

When selecting a specific position, I recommend considering the following:
- Does the role align with your career objectives?
- Does the role play to your strengths or offer opportunities to develop in areas of interest?
- Does the role match your skill set? If not, can you acquire those skills?

Remember, a role can be an opportunity to learn and grow, so don't be deterred if you lack a few skills. If the role excites you, go for it!
My user needs to know if the current task is specific and actionable
Reason: this task needs to be further decomposed as the GPT response is not good enough to help user complete the task. Moreover, It is positioned on the first level of the tree and it involves more than one deliverables: search on companies and search on positions. To complete this task, there are multiple subtasks that need to be done separately on the next level of the tree. These include conducting a comprehensive search on potential companies, searching for specific research scientist internship positions, an analysis of what each role entails, identifying skill requirements, and evaluation of alignment with the user's research interests.
Answer: Yes

User: My user is working on the task Identify Potential Universities: Create a list of universities that offer PhD programs in HCI. The selection can be based on factors such as reputation, HCI research focus, published HCI research papers, faculty expertise etc. The current node level of the task is 2. The GPT response to the task is:
Here is a list of universities that offer PhD programs in HCI:
1. Stanford University, USA
2. Massachusetts Institute of Technology (MIT), USA
3. Carnegie Mellon University (CMU), USA
4. University of California, Berkeley, USA
5. University of Washington, USA
6. University of Michigan, USA
7. University of California, San Diego, USA
8. Cornell University, USA
9. University of Toronto, Canada
10. University of College London, UK
My user needs to know if the current task is specific and actionable.
Reason: The GPT response is good enough and it provides direct answer to complete the task. Additionally, this task does not need to be further decomposed as it is positioned on the second level of the tree and it just involves one deliverable: create a list of schools that offer PhD programs in HCI. Although it may require several steps to create the list, the end goal of this task is to get a list. Therefore the task is actionable.
Answer: No

User: My user is working on the task Identify Required Documents: Reseach and confirm all the necessary documents required for the non-driver ID application, ensuring to list all forms of acceptable proofs such as a birth certificate or passport for identity, Social Security Card or W-2 form for Social Security number, and utility bills or lease agreement for proof of residency. The current node level of the task is 1. The GPT response to the task is:
To apply for a non-driver ID in [redacted] you need to provide several documents to prove your identity, Social Security Number (SSN), and residency. Below, I have listed the acceptable documents for each category based on the latest guidelines provided by the [redacted] Department of Motor Vehicles (DMV).

1. Proof of Identity and Date of Birth (must total 6 points according to [redacted] DMV's point system):
- U.S. passport or U.S. passport card (valid or expired within the past five years) – 4 points
- Birth certificate (U.S., Puerto Rican, Canadian, or Mexican) - 2 points
- Employment Authorization Card (I-766) – 3 points
- Permanent Resident Card (I-551) – 3 points
- Foreign passport with valid U.S. Visa and I-94 – 3 points

Note: At least one document must show your date of birth and at least one document must have your full Social Security Number.

2. Proof of Social Security Number:
- Social Security card
- W-2 form
- SSA-1099 form
- Non-SSA-1099 form
- Pay stub with your name and SSN on it

3. Proof of [redacted] Residency (at least two documents):
- State or federal tax return
- Utility bill (must be dated within the last 90 days)
- Lease agreement or mortgage statement
- Bank statement (must be dated within the last 90 days)
- Any mail from any government agency (dated within the last 90 days)

Additional Notes:
- All documents must be originals or certified copies.
- If your name has changed due to marriage, divorce, or court order, you will need to provide documents that link your current name to your former name.
- You must also complete an "Application for Permit, Driver License or Non-Driver ID Card" (Form MV-44).

Before going to the DMV, it's recommended to use the [redacted] DMV Document Guide which provides a personalized checklist of the required documents based on the specifics of your application. This can be found on the official [redacted] DMV website.

Make sure to double-check with the DMV for the latest requirements or any specific circumstances that may affect your application to ensure a smooth process.
My user needs to know if the current task is specific and actionable.
Reason: Although the task is positioned on the first level of the tree, the task does not need to be further decomposed for two reasons. The first reason is that the GPT response provides direct answer to the task and generates a list of all forms of acceptable proofs for the documents required to get a non-driver ID. The second reason is that the primary goal of the task - identify necessary documents for the non-driver ID application - can be viewed as a singular, cohesive deliverable. Despite involving various types of documents, the task is focused on compiling a comprehensive list, which makes it actionable as a single unit. The distinction lies in the focus on gathering all necessary documentation, a clear and direct objective.
Answer: No

Now, let's start prediction:

Figure 16: The prompt for few-shot + CoT + Tree + Draft for Subtask Detection.

System prompt

You are a useful assistance to detect if the current task needs to be further decomposed if it is not actionable and the primary goal of the task can not be viewed as a singular, distinctive deliverable. Based on the user prompt, please output Yes if it needs to be decomposed; No otherwise meaning it is actionable and does not require task decomposition. Please also provide explanations for your choice.
Here are some examples:
User: My user is working on the task Research on Prospective Companies and Positions: Conduct a comprehensive search on potential companies and specific research scientist internship positions in the NLP field. Understand what each role entails, identify skill requirements, and evaluate how they align with your research interests. My user needs to know if the current task is specific and actionable
Reason: this task needs to be further decomposed as it involves more than one deliverables: search on companies and search on positions. To complete this task, there are multiple subtasks that need to be done separately. These include conducting a comprehensive search on potential companies, searching for specific research scientist internship positions, an analysis of what each role entails, identifying skill requirements, and evaluation of alignment with the user's research interests.
Answer: Yes

User: My user is working on the task Identify Potential Universities: Create a list of universities that offer PhD programs in HCI. The selection can be based on factors such as reputation, HCI research focus, published HCI research papers, faculty expertise etc. My user needs to know if the current task is specific and actionable.
Reason: This task does not need to be further decomposed as it just involves one deliverable: create a list of schools that offer PhD programs in HCI. Although it may require several steps to create the list, the end goal of this task is to get a list. Therefore the task is actionable.
Answer: No

User: My user is working on the task Identify Required Documents: Reseach and confirm all the necessary documents required for the non-driver ID application, ensuring to list all forms of acceptable proofs such as a birth certificate or passport for identity, Social Security Card or W-2 form for Social Security number, and utility bills or lease agreement for proof of residency. My user needs to know if the current task is specific and actionable.
Reason: The primary goal of the task is to identify necessary documents for the non-driver ID application, which can be viewed as a singular, cohesive deliverable. Despite involving various types of documents, the task is focused on compiling a comprehensive list, which makes it actionable as a single unit. The distinction lies in the focus on gathering all necessary documentation, a clear and direct objective.
Answer: No

Now, let's start prediction:

User prompt

My user is working on the task {task title}: {task description}. My user needs to know if the current task needs to be decomposed.

Figure 14: The prompt for few-shot + CoT for Subtask Detection.

- *My user is working on the task {task title}: {task description}. The current node level of the task is {level}. The GPT response to the task is: {Draft}. My user needs to know if the current task is specific and actionable.*