Bridging the Gulf of Envisioning: Cognitive Challenges in Prompt Based Interactions with LLMs

Preprint · February 2024

DOI: 10.1145/3613904.3642754

5 authors, including:

Hari Subramonyam
Stanford University
45 PUBLICATIONS **452** CITATIONS
SEE PROFILE

Roy D. Pea
Stanford University
273 PUBLICATIONS **20,452** CITATIONS
SEE PROFILE

Colleen M. Seifert
University of Michigan
155 PUBLICATIONS **7,661** CITATIONS
SEE PROFILE

Bridging the Gulf of Envisioning: Cognitive Challenges in Prompt Based Interactions with LLMs

Hari Subramonyam
Stanford University
USA
harihars@stanford.edu

Roy Pea
Stanford University
USA
roypea@stanford.edu

Christopher Lawrence Pondoc
Stanford University
USA
clpondoc@stanford.edu

Maneesh Agrawala
Stanford University
USA
magrawala@stanford.edu

Colleen Seifert
University of Michigan
USA
seifert@umich.edu

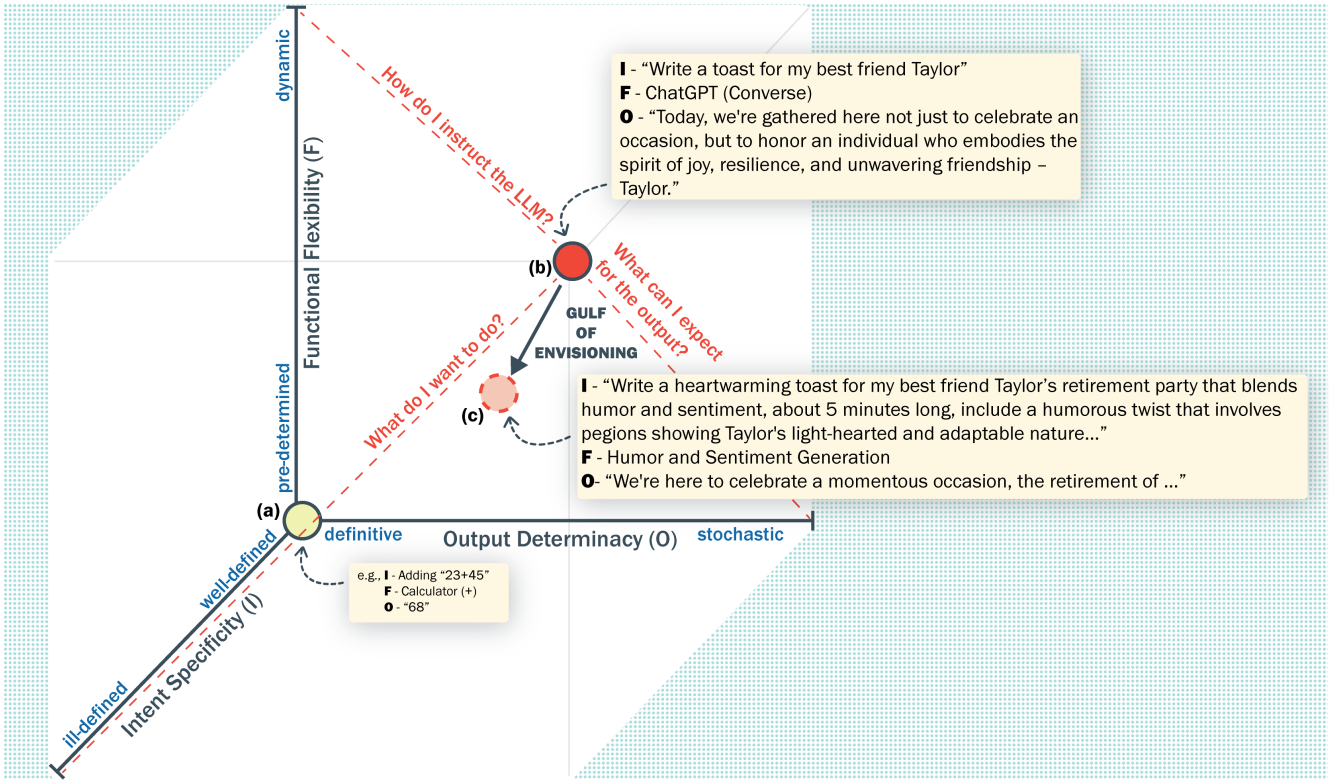

Figure 1: A conceptual framework for LLM-based Interactions along three dimensions: (1) Intent Specificity, (2) Functional Flexibility, and (3) Output Determinacy. Point (a) indicates conventional interfaces such as a Calculator with pre-determined functionality and affordances for interaction. Point (b) represents conversational LLMs such as ChatGPT with dynamic functionality. Gulf of Envisioning is the challenge users face in formulating prompts to generate high-quality outputs, i.e., point (c).

CHI '24, May 11–16, 2024, Honolulu, HI, USA

ACM ISBN 979-8-4007-0330-0/24/05
https://doi.org/10.1145/3613904.3642754

ABSTRACT

Large language models (LLMs) exhibit dynamic capabilities and appear to comprehend complex and ambiguous natural language prompts. However, calibrating LLM interactions is challenging for interface designers and end-users alike. A central issue is our limited grasp of how human cognitive processes begin with a goal and form intentions for executing actions, a blindspot even in established interaction models such as Norman's gulfs of execution and

evaluation. To address this gap, we theorize how end-users 'envision' translating their goals into clear intentions and craft prompts to obtain the desired LLM response. We define a process of *Envisioning* by highlighting three misalignments on not knowing: (1) what the task should be, (2) how to instruct the LLM to do the task, and (3) what to expect for the LLM's output in meeting the goal. Finally, we make recommendations to narrow the gulf of envisioning in human-LLM interactions.

CCS CONCEPTS

• **Human-centered computing** → **Interaction design theory, concepts and paradigms**; **Natural language interfaces**; • **Computing methodologies** → *Natural language generation.*

KEYWORDS

large language models, prompt-based interactions, cognitive psychology

ACM Reference Format:
Hari Subramonyam, Roy Pea, Christopher Lawrence Pondoc, Maneesh Agrawala, and Colleen Seifert. 2024. Bridging the Gulf of Envisioning: Cognitive Challenges in Prompt Based Interactions with LLMs. In *Proceedings of the CHI Conference on Human Factors in Computing Systems (CHI '24), May 11–16, 2024, Honolulu, HI, USA*. ACM, New York, NY, USA, 19 pages. https://doi.org/10.1145/3613904.3642754

1 INTRODUCTION

Large language models (LLMs) such as ChatGPT [124] have demonstrated remarkable capabilities in generating content that is novel, coherent, and contextually relevant. These models can perform a wide array of tasks, from writing comprehensive essays to creating artwork and even producing functional software interfaces, showcasing a high degree of creativity and adaptability. However, they also require careful *guidance* to ensure the generated content is appropriate and in alignment with human goals and intentions. For instance, if an end-user wishes to leverage an LLM to craft a toast and prompts the LLM with *"Write a toast for Taylor"*, the output may be incomplete without providing the desired qualities. The human must be more specific about their *intentions* (such as, "Write a heartwarming toast for my best friend Taylor's retirement party, about 5 minutes long, include a humorous twist, and wish them well on the golf course"). Formal and anecdotal evidence (e.g., [2, 75, 82, 197]) suggests that effectively prompting LLMs to produce outputs similar to human-generated content remains challenging. If intentions are expressed too vaguely or lacking specific detail, the LLM may generate responses that are generic, irrelevant, or off-topic [63, 80, 197]. Iterating with an LLM can correct and progressively guide generation, but playing a "20-questions" or "Hot or Cold" guessing game may be inefficient for longer output and lead to a local minima within the solution space [159]. Further, humans show *fixation* on initial examples that interfere with exploring alternative solutions [71, 93]. **In this work, we draw from theories across HCI and cognitive science to characterize the nature of the cognitive challenges for humans in dialogic interactions with intelligent generative agents.**

As shown in Figure 1, the shift towards LLM-powered interfaces can be characterized along the following three dimensions: (1) Functional Flexibility, (2) Intent Specificity, and (3) Output Determinacy. The vision for artificial general intelligence (AGI) includes LLMs with a robust theory of mind (ToM) of humans (and vice-versa), and would allow effective collaboration across numerous tasks. While not yet at this level of general intelligence [113, 173], current LLMs do exhibit *dynamic* capabilities in that they are able to fulfill a broad range of tasks and generate ad-hoc functionality in response to prompt inputs (e.g., "rewrite these appliance installation instructions for a five-year-old"). This flexibility contrasts with conventional direct manipulation interfaces with pre-determined functionalities (e.g., a calculator). Even contemporary natural language interfaces, while aiming for linguistic variability and conversation-style interactions, remain essentially function-specific, like locating the closest coffee shop [49], updating specific attributes of vector graphics [91], controlling task workflows [166], or constructing data charts [151].

Second, LLMs exhibit open-ended generative characteristics through the vast quantity of linguistic patterns and information learned during training. Their capacity to generate diverse responses to a given input underscores their utility in creative and conversational tasks. While LLMs can produce determinate, correct solutions to closed-ended problems, they offer great value by generating many different solutions to an open-ended problem [119]. As indicated along the *Output Determinacy (O)* dimension in Figure 1, the LLM's capacity to generate varied and unexpected output defines its inherent unpredictability. Humans need to provide oversight and guidance by validating facts, verifying relevance, checking for biases, and evaluating output quality. Third, end-users can converse with LLMs through prompts expressing goals and intentions at any level of specificity. As indicated in the *Intent Specificity (I)* dimension in Figure 1, structure and convention in the input are not imposed by the LLM, leaving end-users uncertain about how to formulate input to improve LLM generation. The new capabilities of LLMs greatly extend human-machine interactions along these three dimensions into new territory through an interaction model where (1) operational scope is not restricted to pre-programmed tasks, (2) all input intentions are allowed with an "anything goes" approach, and (3) outputs are probabilistic rather than determinate.

In this work, we examine the transformative impact of generative AI systems for human-machine interaction, focusing on how this shift from conventional interfaces alters the design and usability of interactions on these three dimensions. Hutchins et al. [68] offer a model of interface-design-challenges in software systems, including an "execution gulf" between user intentions and system actions and an "evaluation gulf" between system output and user understanding of its genesis. Their general principle is that as the *distance* between the human's intentions and the system's interface increases, the costs of interaction increase. LLMs transform human-machine interaction to substantially reduce this distance, and therefore the costs, by defining interaction as human formulation of intentions through natural language dialogue leading to desired output [81, 197]. LLMs narrow the gulf of execution by eliminating conventional needs for *action specification* and *execution* [68], leaving only *intention* to the user. However, the gulf of

evaluation may increase by challenges to *perceive*, *interpret*, and *evaluate* output [68] given the LLMs' probabilistic process.

What are the consequences of these new LLM features enabling success on complex tasks – flexibility in functional scope, variation in intention specificity, and probabilistic processes and outputs – on the nature and costs of human interaction? We suggest this new LLM interaction process poses new challenges for people, which we call, "*the gulf of envisioning*." Concretely, the gulf of envisioning characterizes the *distance* between the human's initial intentions and their formulation of a prompt that foresees how LLM capabilities and training data can be leveraged to generate high-quality output. Envisioning includes at least three challenges for humans interacting with LLM systems: (1) how to set my goals and intentions such that the LLM can accomplish the task – *the capability gap*, (2) how to best instruct an LLM about my goals (i.e., prompt engineering) – *the instruction gap*, and (3) what to expect for the LLM's output – *the intentionality gap*.

In this paper, we formulate a new model of interaction for human-LLM interfaces in which *intentions are the actions.* Our key contributions include (1) a characterization of how transformative LLM natural language interfaces yield both expansive functionality and new challenges in bridging intentions and outcomes; (2) an updated model of human-machine interaction identifying the process of envisioning execution; and (3) a set of design patterns and guidelines for human-LLM interfaces along with an analysis of interfaces for three types of generative tasks.

2 INTENTIONS AND INTERACTIONS IN CONVENTIONAL SOFTWARE SYSTEMS

A primary focus of HCI is designing interfaces that mediate the *interactive* relationship between an end-user and a computational system to accomplish a human goal. To this end, researchers and practitioners have conceived several different interaction paradigms (summarized in [65]), proposed frameworks to understand challenges in human-machine interactions [68, 120], and identified ways to solve those challenges through the use of affordances, feedback mechanisms, task-oriented design, etc. [34, 53, 183].

However, an intriguing question remaining underexplored is **how end-users conceive intentions when engaging with the interface**. In Norman's seven-stage model [120], interaction consists of (1) Establishing the Goal, (2) Forming the Intention, (3) Specifying the Action Sequence, (4) Executing the Action on the System's Interface, (5) Perceiving the System's State as a Response to the Action, (6) Interpreting the State, and (7) Evaluating the System State with respect to the Goals and Iterating until the goal is achieved (see Figure 2). In much of HCI work, stage two – *forming the intention* – is assumed as given [65]. For instance, when cutting and pasting a paragraph of text in a word processor or clicking on the 'Bold' font button, how much do we know (or care) about the underlying intentions leading a user to execute those actions? We posit that while this gap from goal to intention has been inadvertently bypassed in traditional design approaches, it emerges as a critical challenge that must be addressed in human-LLM interactions. In this section, we explore this overlooked aspect of intention formation during interactions and postulate its role in LLM-powered interfaces.

2.1 Defining Intentions

At the highest level, an intention is a *situated pursuit* of a goal that is attainable through the *execution of a process* [14, 70] of a certain sequence of actions conceived as leading towards a goal. An intention is an intermediate cognitive state that translates the abstract desire (goal) into concrete actions. Research in cognitive task analysis suggests that intentions are not just spontaneously generated but arise from a foundation of knowledge, thought processes, and goal configurations [28]. Intentions include aspects such as declarative knowledge (understanding "what" needs to be done), procedural knowledge (knowing the "how-to" of a task), decision points (key moments in reaching a goal where decisions are necessary), and cognitive skills (the mental abilities required to carry out the task). Further, intentions are tied to goals through complex hierarchical structures, and they emerge as the user works towards achieving those goals. This key cognitive science insight was advanced in seminal works by Karl Lashley [92] and Miller et al. [109] in their focus on the hierarchical structure of nested subroutines in human action, opposing behaviorist conceptions of sequential behavior as a chain of stimulus-response associations, and it is further elaborated in current work in cognitive and computational neuroscience [18].

2.2 Role of Intentions in Interactions

How does a user formulate intentions and then specify them as actions to be executed by a system? Let's start with the more familiar aspect of this question, which is action specification, before delving into intentions. According to Norman, users have (or rather acquire) in their mind, mentally represented models of the target system that provide them with the "predictive and explanatory powers" to understand interactions [121]. We refer to these representations about systems that **help with action specification as system mental models**. These models primarily comprise knowledge about how the system operates, including its constituent parts and their interrelations, their inherent processes, and their impact on the system output [24]. While many researchers have characterized the nature of system mental models (e.g., [12, 103, 112, 194]), the common purpose pertaining to HCI is to equip users to determine which action to execute by allowing them to mentally *simulate* the action [162]. For instance, imagine that a writer (the user) is about to type a long and detailed section header. Before committing, they run a *mental simulation* using the system mental model: they envision typing the header in full length, foreseeing it might take up too much space or look cumbersome in the document. Consequently, they may consider changing the header size or shortening the text. This *mental rehearsal* helps them anticipate potential readability issues or aesthetic concerns and select their action sequence accordingly.

In the above example, intention formation has not been considered. That is, when mentally simulating the typing of the header, how did the user conceive the header text (i.e., input to the system mental model) in the first place? Of course, this pertains to the user's goal; let us assume it is to produce a Wikipedia article on Chocolate. A specific intention is the user's decision or plan to type a long, detailed section header in the document, say *"Ethical and environmental implications of cacao bean farming in various tropical regions and their socio-economic impact."* The formulation of this intention is tied to the underlying *cognitive task processes* of writing

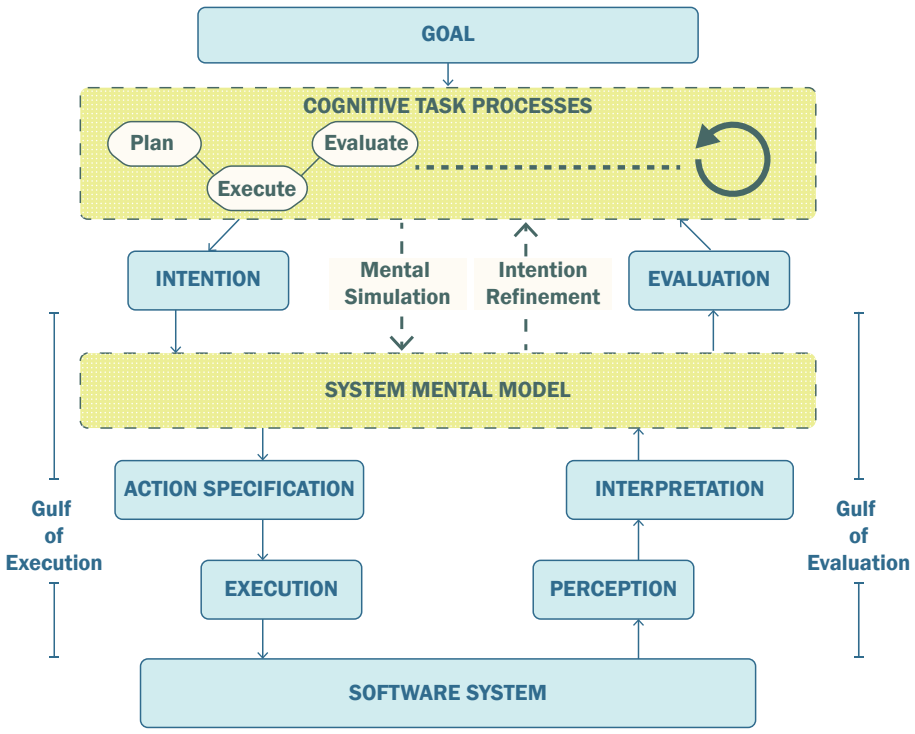

Figure 2: Expanded View of Norman's seven-stage model of interaction. Norman's seven-stage model (blue blocks) is a valuable tool when designing interfaces. However, while the second stage – forming the intention – has often been overlooked in most HCI research, it is a crucial part of human-LLM interactions. We add the cognitive task and system mental blocks (in green) to illustrate the mechanism for intention formation and its interaction with action specification.

that are independent of the technological system. Generally, they concern how people accomplish mental tasks through component processes, such as retrieval from memory, transforming and combining ideas, and using procedures, etc. Specific to writing, such a cognitive task process is formulated by Hayes and Flower, and it involves three *intertwined* cognitive stages, including planning, translating, and reviewing [61]. The specifics of this model are less important at this point, and we will discuss cognitive task processes in detail in Section 3. But what it is important to understand is that these **cognitive task processes support intention formation.** Even for a simpler goal, such as setting a wake-up alarm, the end user must mentally determine the time they need to leave for work, the time it might take to get ready, plan their commute, etc. This is all part of the cognitive task processes.

Critically, these two stages – formulating intentions through cognitive task processes and specifying actions through system mental model simulations – *interact*. In order to generate their intentions for a given goal, the user must be engaged in the cognitive processes needed for that goal. To specify the actions for the system to take based on formulated intentions, the user must simulate the system's mental model to determine how selected actions may influence the execution of their intention. In reality, users constantly adapt their intentions based on both their dynamic cognitive task processes (what they want to accomplish) and their system mental model (what they believe can be accomplished through system actions). In the Wikipedia example, the mental simulation of typing a long header may result in the *reformulation* of the intention to a shorter title, say "Ethical impacts of Cacao farming." Thus, interaction requires (1) intention (formulation and refinement), (2) system mental models that allow a cognitive walkthrough of how the intention might play out at the end of the interaction, and (3) the ability to mentally project the results of their intended actions (i.e., anticipate the eventual outcome of the interaction).

2.3 Role of Design in Mapping Intentions to Actions

Until LLMs, HCI approaches have successfully **interwoven intentions and actions** during interaction design. Through human-centered design approaches, interaction designers strive to ensure that every potential action available on an interface aligns intuitively with the users' goal-based intentions [121, 183]. Even for complex cognitive goals such as reading, writing, reasoning, and sense-making, researchers have developed interactive interfaces by understanding the underlying cognitive process model (e.g., [43, 86, 168]). For instance, in texSketch [168], they build interactions around the cognitive processes in reading, such as selection, organization, and integrated comprehension. This alignment fosters a symbiotic relationship where users feel the system is an extension

of their thought processes, making the interaction feel natural and efficient.

Second, design also plays a critical role in helping users to formulate a mental model of the system, and through it, learn to anticipate system outcomes. In conventional HCI, designers invent a *conceptual model* – an abstract representation or framework to communicate how a system operates – based on human-centered design practices [121]. For instance, the designer may analyze users' past experiences and prior knowledge and draw from familiar analogies and metaphors (e.g., the desktop metaphor) to shape the presentation and behavior of the system [116]. According to Hutchins [67], conceptual models function as metaphors on several levels. At the broadest level, they're shaped by the users' primary goals. In a word-processing tool, this could mean a "blank page" metaphor aligning with users' primary goal of creating a document. Moving in more granularly, metaphors at the interaction level inform users about their computer interactions and often remain consistent across tasks (e.g., a "clipboard" for copying and pasting). Lastly, metaphors at the task domain level provide an understanding of how tasks are organized and structured, such as the use of tabs or headers and footers to indicate document structuring during writing. By establishing this foundation by means of HCI design, users can more readily form a coherent internal representation of the system's operations and functionalities. This representation enhances their ability to plan and execute complex tasks while seamlessly aligning their goal intentions with the appropriate actions.

With LLM systems, the **link between user intentions and system actions is less clear, and end-users lack adequate mental models of LLMs.** Consequently, LLM interactions become challenging for users. In the next section, we postulate how these constraints challenge forming intentions for interactions.

3 ENVISIONING INTENTIONS IN LLM INTERACTIONS

LLMs represent a significant leap forward in the evolution of natural language processing (NLP) capabilities. From the perspective of interactive system design, LLMs obviate the need for structured interfaces with preset actions for implementing intentions in favor of *unconstrained* use of natural language (note that we address specific interface designs for LLMs in Section 4). With a dynamic operational scope lacking explicit interface actions, how do users formulate their intentions and then express them as prompt inputs? One proposed solution is to treat LLMs as if they are human and engage in a conversation with them [33, 37, 136, 197]. Here, we examine this approach to intention formation by considering a specific generative task for LLMs – writing – in three aspects of human-to-human interactions: models of communication, roles and expertise, and theory of minds.

3.1 Cognitive Processes in Generative Tasks

What happens when humans perform generative tasks? Studies show that human performance of 'generative' tasks, such as writing, creating new ideas, coding, and reasoning, appears endlessly variable. At a high level, human task performance takes the form of repetitive cycles of cognitive processes where a change in process, failure, or success is not predictable. A general observation is task performance proceeds through cycles of *"plan a little"* and *"do a little"* [35, 107]. There is no definitive cognitive process for how to accomplish a generative task, though much of education is aimed at instruction on exactly these tasks. Models offer few distinctions among tasks despite distinctive aims, and a high degree of variation in cognitive processes is observed for the same task in different people and for the same person repeating a task.

The predominant model of generative tasks is Newell & Simon's (1972) model of problem-solving [119]. They defined a problem space, including the current state, a desired goal state, and all available actions or operators. The process of problem-solving is defined by actions taken to bridge the gap between nearby states, describing a path toward a solution. For instance, a means-ends analysis process identifies differences between current and desired states and selects an action to decrease their distance [117]. The path from each state is viewed as traversing a solution space containing all possible outcomes. Problem space models describe well-defined problems where these elements are known, and branching can identify all possible combinations of actions. This approach is successful as a model for machine problem solving [118], but when applied to human problem solving, deterministic solution paths with well-defined goals, operators, and evaluation of options are rare [149]. Typically, great variability is observed in cognitive processes, including strategies like back-tracking from a goal to a current state, mental simulation of potential actions, and generation of novel actions.

As a consequence, **cognitive models of generative tasks describe a more variable process** with ill-defined initial states, goal states, and available actions. For example, a four-stage model of human problem solving describes cognitive processes of 1) problem identification, 2) planning, 3) implementation, and evaluation [41]. *Defining a problem* is far from a determinate process yet central to success. *Planning* is defined as identifying and organizing subgoals, intentions, and actions [108, 129] to achieve a goal. Next, an *execution* phase turns intentions into actions. In addition to "doing" the task, people coordinate other cognitive processes, such as monitoring for errors and making real-time adjustments based on feedback. During the *evaluation* phase, users are engaged in a comparative judgment of the actions' outcome in relation to a goal state. Evaluation also includes self-regulation, metacognitive awareness, and value functions, as well as explaining errors to plan subsequent intentions and actions [21, 146, 182]. These stages are so loosely defined that the 4-stage model is ubiquitous in accounts of thinking, creativity, design, and other generative tasks [41, 66]. When applied to an instance of task performance, the cognitive processes are indeterminate, with the progression and order of stages varying, and stages are often independently repeated. *Iteration* is assumed for all stages in these models because the cognitive processes are also defined over subtasks as needed [10].

3.1.1 A Cognitive Task Process for Writing Tasks. A cognitive process model for writing offers a more specific account of the steps required for completing a task [61]. First, the writer defines the task environment, including the audience, the purpose, and tools. Next, the act of writing involves three *intertwined* cognitive stages for planning, translating, and reviewing. The *planning* phase creates a mental roadmap for writing. It involves retrieving ideas or content

related to the topic, articulating objectives such as the intent to inform, persuade, or entertain, structuring ideas in a coherent order, and determining how to transition from one idea to the next. The *translating* stage involves navigating the roadmap and putting ideas into words by converting cognitive representations into linguistic expressions. After a segment (or an entire piece) is written, a third stage involves *Reviewing* it to determine whether it aligns with initial goals, such as coherency, compellingness, and clarity. Based on this evaluation, they may choose to revise the text through restructuring, adding new information, or deleting content. Crucially, this cognitive process during a writing task is not a linear progression from one stage to the next, but an iterative, dynamic process of constant repetition, refinement, and reflection as writers respond to their ongoing assessment of their created output. In this more specific model of cognitive processing in writing texts, the same general steps (planning, implementing, and evaluating) and prominent iteration are evident. In addition, a role for continual monitoring of what is produced and how it meets the goals suggests constant oversight of the generation process.

3.2 Generative Task Interactions with an LLM Agent

Based on this understanding of cognitive task processes, let us revisit the goal of writing a toast for Taylor's retirement. If recruiting a human to help, one might choose an expert who knows Taylor well, and has been to many such events, or an assistant with strong writing skills based on prior knowledge of the expertise required for the task and a theory of specific minds reflecting their knowledge states. Based on this understanding, we may directly ask the expert to *"Write a retirement toast."* With an LLM agent, we can assume training on everything posted on the internet, a generation process by next-word prediction, and conversational interaction to iterate on output. Based on this assumption, the most straightforward path is to simply state the goal to the LLM in their own words, as shown in Figure 3- PATH 1. Such an approach aligns with the theory of computational rationality, in which interactive behaviors fundamentally depend on the principle of expected utility [126]. That is, users tend towards behaviors that maximize expected utility within specific constraints, called bounded rationality [158]. Constraints can emerge internally from cognitive or bodily capacities, or externally from the environment. A key element in this determination is the perception of effort required to improve the input quality and the potential gain in output quality. Given that LLMs are likely trained on databases containing toasts, the resulting output may satisfice [157, 172], and generic or adequate answers may be readily available.

Experienced users may learn that instead of directly stating the goal, a greater *effort* to formulate intentions within prompts can increase the utility by arriving at better or quicker solutions. For example, while it is possible to iterate based on output evaluation, the time to read and reformulate prompts has costs. If the iterative steps can be combined in fewer steps, taking more care in formulating intentions may be cost-effective. For instance, in creating a prompt for the LLM to write a toast for Taylor, the user may engage in planning how they would do the task by identifying topics to include, a structure and organization, constraints for format and length, tone and writing style, desirable qualities, etc. The more developed and specific the intentions, the better the LLM's output should satisfy the user. We call this process envisioning (Figure 3- PATH 2). However, envisioning more developed intention specifications is effortful and cognitively demanding. To craft a more specific prompt for the LLM, the user must further develop their intention by mentally exploring potential plans and values as if – but not actually – executing them, and then adding what is discovered to the prompt. This often leads to taking the Path 1 approach without envisioning. If "an answer" or even a "satisfactory answer" is needed, envisioning may not be. But, if aiming for a high-quality answer, a user may better exploit the system's vast knowledge by envisioning possible and very desirable output. In what follows, we discuss specific cognitive gaps in envisioning LLM interactions:

3.2.1 Capability Gap: Recall from our earlier definition that an intention includes specifications about how to execute task processes. The *capability gap* concerns the users' **inability to formulate "how to" procedures to implement their intentions.** Defining when, where, and how one wants to take action on intentions requires added cognitive effort [128, 150], but has been shown to enhance the rate of successful execution [55]. While LLMs, with their extensive training data, can theoretically understand and generate a wide variety of bespoke task content, their very strength can also be a source of complexity for users. Take, for instance, entering a prompt for summarizing a 2-page text in 1 page. For a human writer, this task involves a myriad of choices about what to prioritize, which nuances to retain, and which details to omit. How can a user identify the right specification of intention for the LLM?

For most generation tasks, a **cognitive process is not already well-formed** in memory but can be made more explicit (or newly generated) through planning. As described in the section on cognitive models, generative tasks are not determinate and vary with each instance of generation and each intention (every story is written with a different specific process). On the other hand, when tasks are well practiced through experience with execution, people may have "scripts" or well-developed routines to complete a generative task [144]. However, some tasks may be **too well learned to summarize as instructions;** for example, asking someone how to make a peanut butter and jelly sandwich without actually making it[1] demands generating actions without feedback from execution. "Doing" offers reminders of where you take added steps to avoid errors. Without this step (since LLMs are doing the generation), users are forced to take a trial-and-error approach with differing abstractions and variations of their intentions, and then iterate based on evaluating the LLM's output. But unlike deterministic systems where a given input always produces the same output, an LLM can produce countless variations from the same input, and there is, by definition, no one correct answer in generative tasks. This **indeterminacy in LLM responses**, coupled with a **lack of transparency in how inputs lead to LLM outputs**, can hinder users' attempts to formulate prompts.

Of course, human-LLM interactions are designed to be conversational and iterative, allowing users to evaluate output, revise prompts, and further refine intentions until satisfied. However,

[1]The EXACT INSTRUCTIONS CHALLENGE video: https://youtu.be/Ct-lOOUqmyY

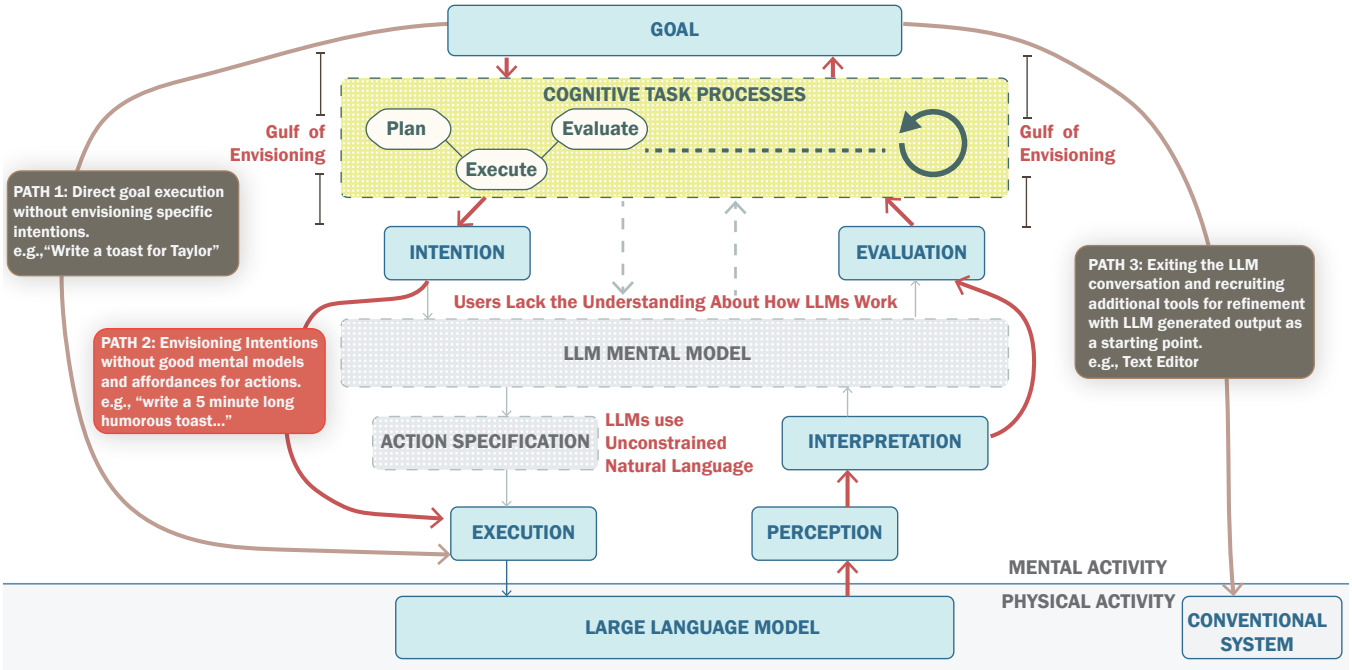

Figure 3: In the context of Norman's seven-stage model action, we highlight what is missing during human-LLM interactions. Further, there are three pathways to interactions: (1) directly state their goal to the LLM, (2) formulate their intentions and provide them to the model through prompt engineering, and (3) take the LLM output and transition to a dedicated interface and system (e.g., switching from ChatGPT to a Word Processor based on an LLM generated draft).

there are **costs associated with iteration from user experience, social exchange expectations** [29], and generative process perspectives. In envisioning intentions, users must find a balance between "batching" intentions in prompts and development through iteration. Intention setting in advance allows users to carefully consider their goals without potential change or interference from the LLM. For example, if writing about personal experience or emotions, iterations may draw the eventual outcome further away from the user's internal perspective. Repeated iteration over many small changes to prompts may **violate a user's conversational expectations** for "chunking" of information in turn-taking. Most importantly, iteration poses the **danger of fixation** on the presented output. Humans tend to become fixated on their own or others' initial proposed solutions [71, 93, 101] and struggle with overcoming attachment to early ideas [30], limiting options for solutions through fixation on a particular type of idea or concept [132], resulting in a lack of exploration of alternatives. If a first output is viewed, the user may "anchor" on it as a solution and focus on local refinements to local minima, thereby missing opportunities in more distant solution spaces. Those with less developed intentions may be more likely to show a stronger fixation on an initial LLM output due to a lack of knowledge about desirable outcomes. Experienced users may learn *when* greater effort to formulate intentions within initial prompts is worthwhile in arriving at better solutions or quicker system interactions. For example, while it is possible to iterate based on output evaluation, the time to read and reformulate prompts has costs. If the iterative steps can be combined in fewer steps, taking more care in formulating intentions may be cost-effective.

3.2.2 Instruction Gap: The *instruction gap* refers to the user's challenges in clearly and effectively expressing their intentions in the interface as text prompts, leading to a potential mismatch between the user's intentions and what the LLM perceives and then produces. One benefit to the natural language interface for LLMs for end-users is a **wide-open input space** for specifying intentions. In writing the toast, the user's intentions as described in the text prompt can *vary* across language expressions; for example, the role assumed (e.g., "act as an event emcee," "act as a best friend"), desirable qualities of output (e.g., "make it funny," "make it heartfelt"), and related text content as a starting point for generation (e.g., "golf," "Vegas"). Unfortunately, the LLMs' algorithms for learning from text corpora create a dependency on **language precision**. While human language use tolerates a wide range of expressions communicating a similar meaning, even slight changes in words can lead LLMs to produce significantly different outputs. It is possible that substantial experience with LLM use will improve a user's understanding of the precise mapping between prompt contents, at least in a domain area. For instance, more expert users may have an understanding of transformer architectures, and experienced users may know how specific keywords in the prompt (e.g., "punchy") influence the attention mechanism. "Instructing" is a much more deliberate task with individual variation, and there are few examples of product instructions done well despite their ubiquity (but see [3]). Users **challenged in performing improvisational linguistic**

dexterity may not obtain satisfactory outcomes solely through trial and error iteration of prompts. Further, linguistic sensitivity makes it **difficult for users to script their intentions for the LLM upfront by anticipating potential system interpretations**. Therefore, users are likely to need a multi-turn interaction with an LLM session to determine alternative specifications for their intentions leading to better outcomes.

3.2.3 Intentionality Gap: In PATH-1 interaction with the LLM, the user states their goal directly and forgoes any cognitive task processes anticipated in doing the tasks themselves (i.e., planning what to write, executing it, and evaluating it). Instead, they start with evaluation of the LLM-generated text. Does it meet their goal? This evaluation is cognitively demanding because no "bridging" steps between goal and output are provided. By skipping directly from goal to evaluating output, users may lack a cohesive understanding of the task context and required content, **hindering their ability to make comprehensive judgments about the adequacy of the output.** Drawing from philosophical terminology, users encounter an *intentionality gap* while assessing texts generated by LLMs. In philosophy, intentionality refers to the capacity of mental states to be about, directed at, or represent something in the world outside of themselves [147]. In the LLM context, users may not have a nuanced "aboutness" of the specific intentions or contexts they are considering and, therefore, may not have a clear sense of what the content should be "about" from their own perspective. Consequently, **without internal cognitive benchmarks of the generation process for comparison, they may become over-reliant on the LLM and avoid evaluation against thier self-generated goals.** Further, engaging in subsequent planning to identify and rectify any deviations from a desired outcome can be challenging. In other words, users producing only a goal statement will have difficulty answering questions such as, *"Can the LLM do better?"* and *"How can the LLM do better?"*

Note that this intentionality gap is different from the gulf of evaluation in Norman's model [120]. With traditional HCI tasks, deterministic outcomes are easier to perceive and interpret, whether completed successfully or not. Feedback is often immediate, as with a direct manipulation user interface in which "…users can immediately see if their actions are furthering their goals [156]." The evaluation gulf stems from ambiguous system feedback at the interface level, which may make it challenging to perceive and interpret the produced outcome. Addressing the gulf of evaluation involves improving system feedback, representation, and visibility. In contrast, the intentionality gap lies at the cognitive level due to the user's failure to create clear intentions. Bridging this intentionality gap with LLMs may require interfaces to scaffold users in creating clearer task process prompts and developing their contextualized understanding of the qualities desired in the generated text.

The idea that under-specified intentions lead to challenges in evaluation surfaces in Karl Duncker's [41] work on the process of finding a solution through a continual restructuring of a problem over time to develop the "essential" properties of a desired solution. Problems intentionally left open-ended, as in many creative tasks, are termed "ill-defined" [159] in that *incomplete* information is provided about the problem. For example, design intentions require a great deal of construction by the designer [138] to identify valuable qualities of potential solutions. Identifying the nature of a problem is key to solving problems, creative work, and design thinking [39, 42, 60]. Exploring problems provides perspectives on values important in solutions [40, 145]. The process of developing intentions helps one learn to "see" the desirable attributes of non-existent outcomes [39] and specify important solution qualities. The power of exploring intention is illustrated by an early study of fine artists asked to create still-life drawings: those showing "discovery-oriented" behavior (rearranging objects, changing perspectives, touching objects) produced works with greater originality and higher quality (as judged by experts), and even experienced greater professional recognition and income years later. **Those who showed more consideration of the qualities of their intended piece produced better outcomes than those focused on execution** [31, 32].

3.3 Recruiting New Pathways for Interacting With LLMs

In the Intentionality Gap, users interact directly with the LLM in evaluation mode. This means they're often trying to make judgments about the generated output and whether changes to the input may produce different outputs more aligned with their goal. **With LLMs, they must attempt these cognitive maneuvers without a foundational context or a cohesive mental model of the LLM functions.** In a third proposed interaction pathway (Figure 3-PATH 3), we suggest a dedicated interface tool with support for the user to further develop intentions and actions to more clearly direct the LLM toward their task goals. This shift to an interface tool that knows specific complex generative tasks – how to write a screenplay or code in Java – facilitates the user's identification of a functional basis for the task process, enabling the user to more precisely specify, calibrate, and synchronize their intentions with corresponding plan descriptions. Critically, **the scaffolding provided by such a "prompt prompter" can facilitate prompt entry and completeness** by identifying expected elements of task functions that are missing in a user prompt and requesting needed information (e.g., "who is your audience?"). Another alternative pathway is integrating LLMs within existing functional tools, as we will see in section 4. The use of a specialized interface tool can help scaffold human interactions with LLMs by dynamically providing structured templates to guide users in clarifying their intentions and fostering clearer directions. This added structure serves as a scaffold for helping users frame their output evaluations with a more informed perspective on what is required in a prompt to generate a successful output, reducing the intentionality gap. While expert users can develop their own cognitive task models through LLM use to identify their own scaffolds, sharing that expertise can greatly help occasional and novice users. Furthermore, a specialized tool can guide users in specifying procedural aspects of their intentions, thereby reducing the need for repeated trial-and-error. Importantly, this prompting scaffold can change dynamically within an LLM session to better reflect the intentions as the user develops them.

As discussed earlier, a cognitive task model such as Hayes and Flowers [61] describes how a writer moves from a goal to the executed text output. While this process is not the only viable way

to generate a story, it is an approach that people often use and, therefore, may be evident in the stories captured in the textual data corpus employed in training LLMs. While the LLM was not trained to write stories, this underlying cognitive model becomes implicitly embedded in the LLM through its training set of words, phrases, and stories, and emerges in the LLM's generation performance. While the LLM does not have an explicit cognitive model of how to generate a story, it is built through statistical analysis of a story corpus created by human writers. That is why it is **helpful for a user to mentally envision what is required for them to perform the generative task: Doing so likely engages the same task knowledge used by other writers with similar intentions.** Then, describing their developed intentions to the LLM using those cognitively informed specifications makes accessing stories with similar intent more likely through the lexical indexes built into transformer networks.

To traverse the extreme database of connections encoded within large learning models and generate a desired outcome, a user must attempt to describe not only *what* others wrote about, but also, *why* they wrote. Without understanding intentions well, it is impossible to access plans from past tasks leading to good outcomes, and impossible to assess outcomes from LLM systems as meeting one's goals. **The value of generative outputs depends entirely on the intentions linking goals to their execution in output.** The three pathways shown in Figure 3 build on human knowledge of successful task completion and vary in the effort and process required. The three gaps, including the capability gap, instruction gap, and intentionality gap, together comprise the gulf of envisioning.

4 EXAMINING THE ENVISIONING GAP IN THREE LLM INTERFACES

Up until now, we have characterized the envisioning gulf based on the core capabilities of LLMs (i.e., language understanding and text generation) with a focus on writing tasks. We now analyze the design of three existing LLM interfaces (see Figure 4) across different generative tasks to pinpoint how the three gaps manifest during interactions. Concretely, we look at how LLM interface designs support the specific cognitive task processes they're built for – using the framework of *planning, execution, and evaluation* – and the specific features they implement to minimize the three gaps in our revised interaction model.

4.1 Writing using ChatGPT

ChatGPT (Figure 4-A) is an LLM with a corresponding web-based interface developed by OpenAI [124]. Through the interactive chat interface, users can supply prompts and engage in dynamic conversations with the model. In the context of a writing task, ChatGPT can quickly produce drafts or outlines based on a given topic, elaborate on an outline, perform grammar and style checks, and paraphrase and synthesize. Naturally, the full spectrum of writing support it offers is diverse and dynamically evolving.

4.1.1 Capability Gap. To support users in understanding the action space, the ChatGPT landing page currently provides *example prompts* and *tasks* the model can perform. These affordances begin to reduce the capability gap during interactions. Yet the tasks depicted often lack the granularity required for users to devise concrete plans. Further, the tool also allows users to start different "chats" for different lines of planning and execution cycles. However, the key cognitive issue in planning is determining how to break down their goals into specific, actionable steps with the LLM. Simply having different chats may help with the organization, but it does not close the gap in helping users know how to best to formulate their intentions.

4.1.2 Instruction gap. Users mainly discover how LLMs interpret prompts within ChatGPT through trial and error. These trial and error actions subsume features such as regeneration, editing the original prompt, and managing different "chats" with ChatGPT on the left sidebar. Yet ChatGPT does not provide a history of previous outputs, making it difficult to compare the quality of regeneration from run to run. If users cannot see how slight language changes affect outputs, they might struggle to learn from their linguistic adjustments to the prompt. In other words, the lack of feedback inhibits their ability to understand how language nuances influence the model's interpretations, making the Instruction gap even more pronounced. Separately, ChatGPT allows users to set "custom instructions," which specify details and guidelines when engaging in a dialogue with the model. However, this feature requires users to foresee these intentions upfront before interacting with the model and a more general understanding of instruction utility across tasks.

4.1.3 Intentionality Gap. The intentionality gap reflects the challenge users face when evaluating the LLM-generated text because they bypassed the planning and execution processes. Custom instructions are the primary mode for aligning user values with model output, acting somewhat as "base prompts." There are two parts to defining custom instructions: (1) what you want ChatGPT to know about you as well as (2) how you would like ChatGPT to respond. The sum of these two mimics domain-specific prompts seen in other LLM-enabled systems, as the former asks the model to play a role in narrowing its scope while the latter helps steer the model's generation. Overall, this feature ensures that users no longer need to specify such context when prompting, allowing them to focus on crafting a good prompt based on their specific goals and intentions. However, even with this alignment, users may still encounter challenges in assessing the output because they may lack a comprehensive mental model of the content. Finally, at evaluation time, users can ask the LLM questions about its previous inputs, regenerate a response from the LLM, or even directly edit an old prompt to see how a model changes its answer. These features aim to give users more clarity on the LLM's outputs, offering ways to refine and adjust the content to better match their intentions.

In essence, while these features of ChatGPT offer initial guidance and organization, they don't fully equip users to carry out the planning and evaluation tasks.

4.2 Creative Coding using Spellburst

Next, we look at Spellburst (Figure 4-B), a creativity support tool developed as a research artifact [5]. We selected this tool as it aims to support the artists' "exploratory creative coding" workflow to address cognitive challenges in creative work. Specifically, they focus on bridging the artist's creative intents expressed in natural

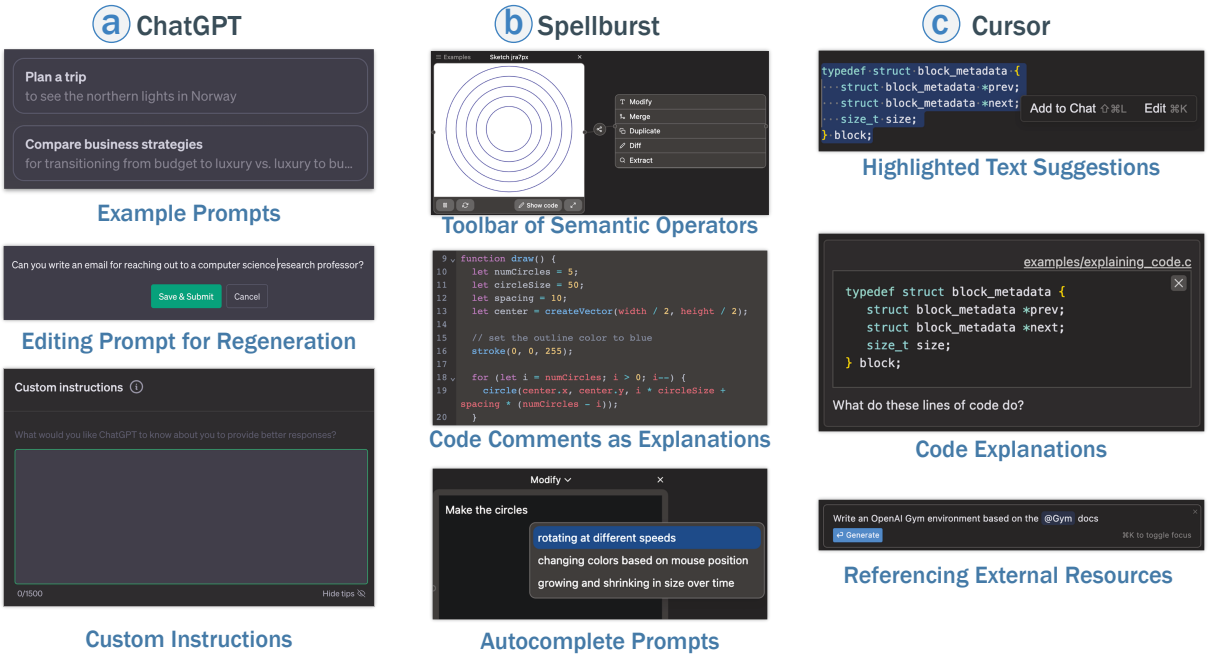

Figure 4: Example affordances for how (A.) ChatGPT [124], (B.) Spellburst [5], and (C.) Cursor [7] bridges the capability (top), language (middle), and intentionality (bottom) gaps.

language with the implementation of those intents as code using LLM.

Briefly, they provide a node-based interface that allows artists to prompt an LLM – namely, ChatGPT – to generate new computational artwork (called a sketch) and also execute creative operations such as merging different outputs or creating variations through branching. The tool provides example sketches in code to iterate from, a toolbar for each node with a set of possible operators, and autocomplete suggestions when authoring prompts. Users primarily execute the task by prompting the LLM to generate code; however, users can also manually write and edit LLM-generated code for use in further ideation. Finally, Spellburst offers various ways of evaluation, including interleaving comments in the LLM's output and allowing users to ask questions about the output through the diffing and extraction operators. The node-based canvas of the interface is also well-suited for exploring different lines of iteration after evaluation.

4.2.1 Capability Gap. To minimize the capability gap, the interface first provides a series of example sketches. These examples, which are snippets of p5.js [105] code for sketches such as "Bouncing Balls" and "Fractal Tree," provide the user with ideas for what types of tasks are best suited for the interface. In addition, the interface shortens the distance by helping users author prompts through *autocomplete* based on their crowd-sourced taxonomy. This helps them understand what is possible with Spellburst, especially for users who may have an initial idea for a semantic jump. Finally, the set of semantic operators on the right side of each node, such as "Modify," "Diff," and "Extract," showcase the different types of ways users can extend their current line of thinking using the LLM.

4.2.2 Instruction gap. Spellburst also uses a variety of techniques to bridge the Instruction gap. After the interface is prompted to generate a sketch, the model interleaves *comments* within the code of the sketch that point to the specific change in the code based on the user's prompt. These comments enable users to see how the model's inputs prompt its outputs. As an example, if a user modified a sketch to turn its outline from black to blue, the model would output the sketch with a code comment in the location where the outline color was changed. There are also numerous ways to experiment and see how the model may interpret different prompts, including the duplication and modification of a sketch node or the creation of a new branch within the interface entirely, representing a new line of semantic and syntactic jumps. Finally, two semantic operators help make the model's output more explainable by asking the LLM to describe its own output – diffing and extracting. *Diffing* allows users to see both syntactic (code-level) and semantic (prompt-based) differences between two sketches, while *extracting* enables users to ask the model questions about a sketch.

4.2.3 Intentionality Gap. Based on their provided prompts, Spellburst asks the LLM to "act as an expert creative coder." Through this approach, they are setting a high-level intentionality of how the LLM should be thinking about generating the output. Furthermore, the prompts help ensure that the model generates code that runs and compiles correctly. The team behind the system also crowd-sourced a set of image transformations to develop a creative coding taxonomy. This taxonomy drives the autocomplete suggestions, as users can orient their intentions alongside this shared vocabulary when prompting the model, improving the quality of the LLM's output.

4.3 Software Development using Cursor

Cursor (Figure 4-C) is a code editor that helps users pair program with AI [7]. The system uses an LLM – users can choose between GPT-4 [125] or ChatGPT [124] – to help developers chat with their project, make code changes, and address bugs. Given the growing interest in utilizing AI to supercharge the software development process [54, 69], we analyze how Cursor implemented a human-LLM interface for the complex cognitive task process of programming.

As a fork of VSCode [106], Cursor has numerous tool affordances critical to supporting programming, including a file explorer, line completion, and a search bar. However, there are many features specific to Cursor. When planning, users are given several examples of tasks they can accomplish. In addition, there are various ways to converse with the LLM, including when selecting the text, using keyboard shortcuts within the main editor, and the right sidebar dedicated to chatting with the model. During execution, the user can either write code normally in the editor or prompt the model to generate code. Finally, the system provides several features for evaluation, including regenerating model responses, chatting with a model about code, and auto-debugging.

4.3.1 Capability Gap. Cursor's first way of showcasing what tasks are available within the system is through a set of example files when a user first opens the editor. Each of these files is named after a possible task, and the comments in the file give instructions regarding what keyboard shortcuts and prompt to use to complete the goal. Similar to ChatGPT, though, since these tasks are high-level (i.e., "fix a bug," "explain code"), they are not helpful for more specific planning of goals and intentions. When interacting with code across the various panels of the interface – including the code editor and the chat interface on the right-hand side – the tooltip shows both what actions can be taken with an LLM and how to establish context. For instance, highlighting text within a code editor gives users the option to either add the code to the right-hand chat interface or edit directly with a prompt. Likewise, inside of the chat interface, buttons appear for users to reference a file or change the scope of the user's question within a prompt.

4.3.2 Instruction gap. To help users understand how LLMs interpret language, Cursor allows users to ask models questions about their codebase. Topics for these questions can range from the inner workings of a particular function to the overall flow of an entire project. Since this feature can give answers about model-generated and user-written code, developers can get a better idea of how LLMs interpret their programs"under the hood." A similar feature that can help decipher how models understand code is prompting the model to generate inline comments. Finally, despite no direct functionality for regenerating output outside of feeding the model the same prompt, there is an option to "rerun without context" to see the effects of adding code or references to other materials alongside a prompt. The lack of regeneration history, however, makes it more complicated for users to map changes in their inputs to changes in model outputs.

4.3.3 Intentionality Gap. The primary way Cursor aligns user intent with model output is through references of code snippets, files, and documentation. When authoring a prompt, users can attach an existing file of code they've written, a function within the code editor, or even documentation of third-party libraries to steer the output of the model. This functionality has many use cases, including prompting the model to adopt the same style or asking the LLM to use a particular method or function from a library. While this feature reduces the amount of effort involved in establishing context with the LLM, this affordance does not inherently give users a mental model of how the LLM works, which means that users may still run into issues. Furthermore, to help evaluate output, user can ask Cursor questions about their codebase. The questions can be about any piece of code within the code editor – either model- or user-generated – to help users understand how to make their prompts more optimal.

5 RECOMMENDATIONS FOR DESIGNING INTERFACES FOR ENVISIONING

In addition to the three interfaces described in detail in the previous section, we conducted a qualitative analysis of 12 systems to identify design patterns that would potentially support the process of envisioning. We only selected tools that were either directly accessible or at least had a video demonstration and accompanying technical description of the features, such as in research papers. For each tool, we identified affordances along the three main operators in generative tasks, namely planning, execution, and evaluation. We then clustered the identified features based on their functional similarity to develop a set of interface design patterns and corresponding tenets for teams building human-LLM systems. We do not claim a comprehensive categorization but provide a starting point for the design of interface affordances for envisioning LLM interactions.

Design Pattern 1 – Visually Track Prompts and Outputs: LLM interactions are often iterative and take a trial-and-error approach. We observed that some tools provided users with a visual interface for capturing their prompting and divergent thinking through alternative pathways. A popular choice is node-based interfaces, which allow users to visualize multiple different outputs (nodes) and trace how they are connected (edges). As mentioned before, in Spellburst [5], each node represents a sketch, and edges showcase different iterations of the sketch through merging, diffing, and other semantic operators. In PromptChainer [185], a system that helps users chain together LLM prompts for complex tasks, each node represents an individual prompt and corresponding output while each edge represents the use of the output as context for another prompt. Such external representations of their thought process not only lower cognitive load but also help users better engage in more deliberate prompt authoring. For instance, users can readily see if their previous line of thinking was fruitful or make adjustments to subsequent prompts based on prior outputs. These affordances correspond to the tenet that: *Users need guidance navigating the cognitive task space for prompt authoring, discerning which paths lead to desired or poor outcomes [T1].*

Design Pattern 2 – Suggest Ideas for Prompting: Many systems proactively offer users prompt suggestions. Some of them are aimed at assisting users who may not be as familiar with LLM-enabled interfaces, while others serve as ideation partners in the cognitive task workflow. In addition to providing ideas for prompts, examples

and suggestions emphasize the importance of clarity and precision in language. If these suggestions come from examples where the model is fine-tuned, they can also help better align a user's intentions with model behavior. For example, ChatGPT [124] provides standalone prompt examples to showcase its utility. On the other hand, Notion [122], which is a tool for knowledge management, not only offers possible ways to use AI to improve your writing (i.e., "Change tone"): it also gives suggestions on how to prompt an LLM to steer these improvements in a certain direction (i.e., "Professional," "Casual," "Straightforward"). These patterns lead to two Tenets, namely: *LLMs can serve as cognitive partners in task formulation [T2]* , and *A focus on clear, precise written language will help to bridge the gap between human intention and LLM output [T3].*

Design Pattern 3 - Provide Multiple Outputs: Rather than generate one output based on a user prompt, LLM systems may provide numerous outputs. This can be achieved by setting the *temperature* of the model – a parameter that dictates randomness – greater than 0 and giving the model the same prompt or explicitly asking the model to give more than one example. This feature allows users to view multiple options to see which best fits their intentions. Furthermore, providing multiple outputs helps users link the effects of changes in prompts to changes in the final output of the model. Some systems also support grouping and clustering model outputs to make this process easier. For instance, BotDesigner [197] lets users manually assign a tag to model outputs, while Sensecape [169] groups relevant topics together semantically based on a high-level topic. These features support the tenet that *LLMs should support users through their divergent thinking strategies [T4].*

Design Pattern 4 - Make the Output Explainable: Some systems prompt LLMs to explain their outputs or make them more interpretable. This design pattern allows users to better understand how LLMs interpret certain prompts and makes model outputs easier to use for manual editing. How this technique is applied in practice can differ depending on the task domain. Replit [137], a browser-based code editor, has an AI assistant named Ghostwriter that generates in-line comments within its code responses. Another code editor, Cursor [7], does not always provide code comments but does allow users to ask LLMs about the code they generate. In contrast, Sensecape [169], which is designed for exploration and sensemaking, prompts an LLM to return a response at different levels of detail, such as through summaries and keywords. These features help users address their intentionality gaps and better assess the model output. This pattern supports the tenet *An error in human-LLM interaction is not just a user error or LLM failure but signals a breakdown in the distributed cognitive system that requires collaborative repair [T5].* However, in designing for explainability and drawing causal inferences between prompt inputs and outputs, design should account for users' overreliance on explanations without careful validation [48].

Design Pattern 5 - Use domain-specific prompting strategies: Outside of standard prompt engineering techniques, most systems use a custom prompting strategy depending on their task. These methods help steer the outputs from LLMs into something usable for the end goal while also minimizing the output ambiguity that may arise in trying different prompts. As an example, Graphologue [74], which is designed to turn text-based responses from LLMs into graphical diagrams, uses prompting techniques to have models annotate entities and relationships within their outputs to create diagrams in real-time. Coding Steps [78], a web-based application to help novices learn Python, prompts models with static examples, then user code, then the user prompt, to ensure that the level of output is appropriate for beginners. These strategies allow designers to implement conceptual tasks for users and consequently allow them to build task-specific system mental models. The corresponding tenet is that, *Users favor working with a system mental model leading to actions when working within a defined task domain [T6].*

Design Pattern 6 - Allow manual control of output: Many systems afford users the opportunity to manually edit the outputs and interactions with LLMs. Since many LLM-enabled systems are built for exploration and ideation, direct manipulation can help users better incorporate their values and intentions into the model. Oftentimes, manual editing is introduced when one output serves as input to another LLM. For instance, while LIDA [36], a tool for generating visualizations and infographics, prompts an LLM to output goals for dataset exploration, users are also allowed to enter their own goals and adjust the model's suggestions. Likewise, Mirror [187] – an NLI for data querying and summarization – gives users the ability to edit the SQL queries generated by a pre-trained code model to add human expertise. These features align with the tenet, *If tasks are well-defined, people prefer dedicated interfaces over dynamic interfaces [T7].*

6 DISCUSSION

In this work, we have theorized about cognitive challenges emerging in the transition from conventional software paradigms to prompt-based interactions powered by generative models. Based on prior empirical evidence on challenges with prompting [80, 196, 197], we have applied cognitive science and HCI perspectives to characterize significant HCI design challenges with prompt-based interactions. Given the advanced cognitive capabilities of LLMs, people are now able to express in natural language their bespoke task goals and ask the LLM to perform those goals for them. At the same time, they lack the specific affordances of conventional systems in formulating their intentions and task plans and evaluating the LLM outputs. Given the shift in the operational scope from deterministic functions to dynamic intelligent agents, we have identified new cognitive process models for specifying actions through intentions, i.e., the process of envisioning. In reasoning about envisioning intentions with LLMs, we have also identified three specific gaps including the capability gap, instruction gap, and intentionality gap, and we have provided initial recommendations for interface designers to scaffold prompting. However, a number of open questions remain about designing prompt-based interfaces. Here, we propose open questions for future research as we consider future development and applications of generative models.

6.1 Open Questions for Designing Human-LLM Interactions

6.1.1 How should we model conversational interactions between humans and LLMs? Given the human-like conversational abilities of LLM interactions, designers must consider how to effectively model conversations. Recent work on designing natural language interfaces (e.g., [38, 152]) has primarily considered the 'Recipient Design' approach based on Grice's Cooperative principles [20, 58, 164]. This approach models communication as sensitive to context and individual characteristics, and it recognizes that speakers tailor their speech and communicative behavior to meet the needs of their listeners. As the influential literary theorist Mikhail Bahktin theorizes [9], any utterance has both addressability (every word always addressed to someone) and answerability (every word directed toward and anticipating an answer from someone). Sociolinguists describe these features as manifest in the *recipient design* of an utterance [56]. The maxims of quantity, quality, relation, and manner guide how information is conveyed and interpreted to ensure clarity, relevance, and truthfulness. However, with LLM dialog, such an inductive approach can be challenging for users as these maxims depend on each conversationalist tracking a theory of mind [50] regarding what their partner thinks and knows. An AI capable of a complex theory of mind for individual users across sessions will likely require extensive development and testing for feasibility. An alternative model is the Transactional Model of Communication [141], which offers a more dynamic view of interchange. Rather than precision, a deductive process allows for repairs and adjustments of misunderstanding and miscommunication, and consider the contextual and continuous nature of communication. Applying this model for human-LLM interaction, rather than emphasizing the quality of communication, in interfaces offers opportunities for repair and feedback, e.g., editing a previous prompt in the ChatGPT interface.

Other models of communication, such as the Socio-Cognitive approach to Pragmatics [79] and Speech Act Theory [148], may also be useful for design. The socio-cognitive approach accounts for how the speaker's and listener's background, intentions, and situational contexts contribute to constructing meaning. This approach may provide guidance for characterizing the agentic roles of LLMs in personalization and adaptability, modeling intentions and expectations, code-switching, and ethical and responsible interactions. Further, focusing on the 'functions' of language, i.e., speech acts, may allow design for conversational rules and conventions. Lastly, according to social exchange theory [29], communication behaviors are influenced by design to maximize benefits and minimize the cost of interactions. Future research should investigate the balance and trade-offs between high-quality intuitive prompting and the adaptability offered by more deductive conversational approaches. This exploration should focus on how varying models of communication can enhance LLMs' ability to understand and adapt to diverse user backgrounds, intentions, and contexts. By integrating insights from these communication models, HCI research can develop LLM interfaces that not only facilitate efficient exchanges but also support ethical, personalized, and contextually sensitive interactions.

6.1.2 What is a useful theory of mind for LLMs to support effective envisioning? In conventional systems, users' mental models of systems provide a functional account of how the system produces its output, and that knowledge is used to generate necessary action specifications and evaluate whether the output is " good enough." In the case of LLMs, novice users likely do not have a system mental model that can sufficiently describe what happens at a process level, or that understanding does not allow the prediction of output from input. The requisite mental model to account for an LLM's performance includes both the training process and the algorithms used to learn; further, it must include a theory of *what* was learned by the LLM. If we consider LLMs from a purely human-like perspective, as in human-human communication, we require a cognitive understanding of how communication works, including forming assumptions and expectations, norms, shared knowledge, feedback processing, symbolic understanding, perspective taking, reciprocity and feedback processing, etc. For another human, I can use my own mind to generate predictions of what output will likely be produced by another mind using my own mind as a guide; for example, the colors red, white, and blue make me think of America. However, no other minds have the equivalent dataset of an LLM. Its scope defies the predictions one can generate from the information encoded in one human mind. Without a sense of the outcome of its learning process, it is impossible for another mind to predict LLM output. That is why LLMs are exciting generators different from humans but connected through text descriptions of human experiences. It is sufficient to engage in a conversational LLM interaction and feel like talking with another person who has a theory of mind about the individual and associated beliefs, feelings, and goals through the power of intentions.

Future research should explore how to bridge the gap between the complex, often opaque inner workings of LLMs and the intuitive understanding of its users. This work will involve creating models that accurately represent the LLM's operations in a user-friendly manner, aligning them with the system's actual functionality. The challenge lies in simplifying the complex mechanisms of LLMs without sacrificing essential details that users need to predict the outcomes of their interactions. Prior research on end-user programming has studied the challenges novices face in envisioning simple interactive features and ways to support requirement specification, debugging, and verification [81]. Similarly, for LLMs, theory should account for the evolving nature of user requirements and the emergent design process, acknowledging that users often learn and refine their understanding of LLMs through experience. Efforts should be concentrated on enhancing the visibility of the LLM's processing pathways, perhaps through interactive visualizations or simplified explanatory frameworks to reinforce user understanding through better prompting, iteration, and evaluation.

6.1.3 What is the optimal "sweet spot" for Human-LLM interaction along the three interaction dimensions? Reconsidering the three dimensions in Figure 1, LLMs are appealing because the inputs can be abstract, complex, and vague (i.e., underspecified), and they can still produce outputs that are good. As mentioned earlier, if the quality of the answer matters, end users will want to iteratively explore the generative features to reach a high-quality answer. In such cases, envisioning can be challenging, and in both our revised model and

as seen in prior work [196], we need better guidance in designing interfaces to make envisioning easier for users. When powering interfaces with LLM capabilities, how do we balance Intent Specificity, Functional Flexibility, and Output Determinacy in ways that leverage the strengths of the LLM while aligning with the user's needs for quality and relevance? For Intent Specificity, a semi-structured (templated) approach is advantageous. While LLMs excel at interpreting and generating responses to open-ended queries, a certain level of specificity in the user's intent can guide the LLM to produce more targeted and pertinent content. Functional Flexibility should be oriented towards dynamic responsiveness. The LLM's ability to pivot across different tasks and domains should be fully utilized within a framework that is shaped by the context of the interaction. Future research should investigate efficient pathways and optimal points along these three dimensions. Such inquiries will be likely to include novel interfaces that suggest modifications to vague inputs and feedback systems that learn from each interaction to refine future responses. Moreover, future research should also consider the evolving nature of user expectations and the continuous advancements in LLM capabilities, ensuring that the "sweet spot" for AI and interface design remains a moving target that adapts to the growing sophistication of both users and technology.

6.2 Limitations and Future Work

Of course, LLMs have been game-changing for AI, and have launched a wildly diverging portfolio of applications. They evidence the fact that much of everyday human task performance is rote, standardized, and repetitive. Strong patterns across the text database reveal how little originality exists in the accumulated human text products available in digital form. It is quite possible that the deep cognitive processing proposed here as a means of probing LLMs to produce output more similar to desired human outcomes is rare. However, its value when it does occur suggests it is well worth developing models of the cognitive processes shared across cognitive tasks.

Another limitation of our model is the obvious differences among people in their ability to identify intentions from goals. Divergent thinking [59] is a minor part of academic training across the school curriculum, whereas converging on a single correct answer dominates learning. People show major differences in their ability to solve open-ended problems and complete generative tasks, potentially attributable to differences in cognitive capacities, including memory but also imagination. A common test of creativity, the unusual uses test, asks for different ways to use a common object, such as a brick [59]. People often fixate on functions such as using the brick as a weight, and generate uses like an anchor, paperweight, and balloon holder. Less often, they identify unusual functions, such as using its material as a dye for crayons or lipstick. Perhaps using human minds as the "key" limits solutions to only those generated by the human mind. While the LLMs are currently making use of the text-based products of human minds, it is possible that future systems will encode different forms of data less dependent on human goals and intentions and more content generated by AI models. Further, more systems with corpora combining products of humans and AIs may diverge further away from cognitive models as explanations of links between input and output.

Finally, this work is intended to propose a direction for the development of new approaches to HCI. Further pathways for human-LLM interaction can be identified, and new interface supports for LLM use based on prompting guidance are growing daily. While other approaches may be quicker (and dirtier) so as to plug obvious holes in current LLMs, the promise of this work is to capture the intentions found useful by humans in executing generative tasks. To test this approach, comparing prompts where the intention is evident will determine its value in creating satisfying outputs from LLMs. Strategies from co-work, such as asking someone to repeat back their understanding of the task instructions given, may prove similarly useful with LLMs. An empirical agenda can determine not just the factual or writing quality of AI systems but also their value to human users. This proposed approach to HCI with LLMs aims to support the user as they must think more deeply and fully during interactions with systems in order to integrate their processing abilities with the strengths offered by systems. This work, like the development of LLMs, is at its beginning stages.

7 RELATED WORK

7.1 Documented Challenges of LLMs

A core challenge of using LLMs is their *explainability* [102], i.e., how can we explain why a model behaves in a certain manner? Compared to the domain of traditional machine learning, LLM explainability is a different challenge [198], as these models are pre-trained to do a variety of complex reasoning tasks [188] and absorb patterns from data automatically [110]. Regarding inputs to LLMs (i.e., prompts), the largest issue is that it is not always clear how a prompt strategy affects model output [97, 142]. Even for popular methods like chain-of-thought – which asks a model to explain itself – there is no evidence to suggest whether models reason towards the answer through the steps they provide, based on their pre-training data, or through other heuristics [143]. Overall, explainability, or the lack thereof, is a significant contributor to the gaps involved in LLM interaction, as users struggle to build a mental model [15, 170, 174].

Another challenge with LLMs is concerned with the *usability* of their outputs. For instance, such models can hallucinate, where the text generated seems structurally correct but is actually nonsensical or incoherent [13, 72, 135]. In addition, LLMs do not always produce factually correct output, and it may be difficult to verify whether the output is correct or not [73, 104]. Depending on the context and domain, such as in the realms of medical or military applications [123], this inaccuracy can be severely detrimental [83, 94, 140]. Furthermore, while a prevalent technique for addressing this shortcoming is to provide sources, it can be hard to implement in practice and may also not always be correct [16, 96, 134]. Combined with the issue of explainability, it can be difficult to correct and steer LLMs to responses that are more usable.

Lastly, a final challenge concerns the issue of *bias* in LLMs. Much empirical research has shown that a plethora of biases are encoded in these models, including racial bias, gender bias, and bias around political leanings, to name but a few [45, 115, 155, 171]. There is also an abundance of work in the realm of jailbreaking LLMs to generate toxic outputs and leak private information in both their training data and conversation history [25, 98, 175]. There are many

factors contributing to this propagation of bias [46], including the corpora these models are trained on, how the data is labeled and annotated, and the architectures that power these LLMs under the hood [13, 23, 114]. Overall, while existing models reject most harmful prompts (i.e., ChatGPT responds with "I'm sorry, but I can't assist with that request."), socially situated contexts can still produce potentially offensive output from an LLM [153].

7.2 Prompt Engineering

Prompt engineering encompasses the set of techniques used to converse with LLMs. These methods assist with setting rules, structuring output, and overall guiding the model in the direction in which a user intends [180]. While there has been much research devoted to uncovering these emergent properties, these techniques are often simple tricks that are intended to mimic the process of human reasoning.

One of the most effective prompting strategies is providing examples of expected input and output, also known as few-shot prompting [22]. The inspiration comes from human cognition, as people can learn new concepts from a small set of examples while also applying these concepts to new inputs [88, 89]. The effects of few-shot prompting are more pronounced when models are of a certain scale, and there are numerous factors that can aid or inhibit the helpfulness of such prompts [76]. These include the semantic similarity of the training examples to the test examples, the choice of prompt format, and even the order of the examples in the prompt [95, 199].

Another predominant technique used in prompting is chain-of-thought, or providing a series of reasoning steps to show the model how to get to the final answer [84, 178]. These prompts are helpful to learning because, for humans, explanations break down why a certain answer is correct and not just what the final answer is [87, 90, 99, 131]. Coupled with few-shot prompting, there are numerous factors that make for useful chain-of-thought prompts, such as the amount of complexity within the prompts (measured by the number of steps), whether the provided examples are relevant to future queries, and the correct ordering of the reasoning steps [26, 51, 176]. There are also a plethora of variations that build upon chain-of-thought, including sampling multiple responses given the same prompt [163, 177] (Self-Constency Sampling); repeatedly prompting a language model to ask follow-up questions [130] (Self-Ask); and both decomposing a problem into numerous steps and sampling numerous responses at each step [192] (Tree of Thoughts).

7.3 Designing Human-AI Interfaces

The rise of artificial intelligence (AI) has led to a surge of interest in the development of human-AI systems. The creation of these novel systems has brought new challenges. For instance, since these models are sometimes perceived as non-deterministic "black-boxes," users can have a hard time discerning how these interfaces produce their outputs [11, 62]. Likewise, given the massive amount of data these models are trained on, there are new concerns around the bias and privacy of such systems [17, 85, 193]. Perhaps most critical is the change in the relationship between humans and interfaces: while traditional NLI systems served more as assistants to end users [44, 181], human-AI systems act more as collaborators due to their human-like cognitive abilities [19, 100]. As a result of this role change, there are many propositions for human-AI interaction guidelines [4, 111, 184], including conveying the consequences of user actions, providing diverse options from models, and holding the system accountable for errors. Further prior research has looked at changing design practices to accommodate the new challenges of designing user experiences for machine learning capabilities [165, 167, 190, 191].

Specific to natural language interfaces, prior work has looked at designing natural language interfaces and LLM-powered chatbots [189, 196]. These interfaces are most prevalent in the domains of data visualization [52, 161] and querying [1, 127]. There are several issues involved in the development of NLI systems, including the ambiguity of natural language, communicating to users what the system can do, and evaluating the utility of these interfaces in accomplishing their end goal [154]. To this end, researchers within academia and industry have put forth ideas for creating effective NLI systems, including the use of autocomplete to show users how to phrase their queries to the system [8, 195], the design of conversational interfaces to engage users in a back-and-forth dialogue about their intentions [64, 151, 152], and the development of multimodal features to give users more control over their input to the interface [77, 160].

Finally, recent work in HCI has been focused on generative AI, in particular, LLMs [6, 57, 124] and diffusion models [133, 139]. Within this space, significant effort has been concentrated on developing effective prompt strategies (see Section 7.2), helping users craft and author these prompts [47, 186, 197], and also developing preliminary guidelines for generative models [27, 179]. Focusing specifically on LLMs, though, what is missing in the current literature is a better understanding of *why human-LLM interaction is different from human-AI interaction* – and more broadly, *how human-LLM systems are different from traditional natural language interfaces* – and what types of strategies designers can employ to address the gaps that result from these new computational opportunities.

8 CONCLUSION

Our work applies a cognitive framework to characterize the dynamics of prompt-based interfaces such as ChatGPT, highlighting the complexities of interacting with LLMs. While LLMs are capable of interpreting a vast range of queries, their very flexibility can pose challenges for users attempting to convey precise intentions. We identify and characterize a new kind of interaction gulf called the "gulf of envisioning," which captures the challenge users face in successfully formulating their intentions to elicit the desired response from an LLM. This gulf is further identified by the capability gap – what intentions can the LLM perform, the instruction gap –how to say what is needed to the LLM, and the intentionality gap – what to expect and how to evaluate the generated output, all of which describe varying facets of human-LLM misalignments. By arguing that for LLM interfaces, "intentions are actions," we provide design recommendations to support the process of envisioning with generative AI models.

ACKNOWLEDGMENTS

We thank the reviewers for their feedback on the paper. Subramonyam and Agrawala are supported through the AI Research

Institutes program by the National Science Foundation and the Institute of Education Sciences, U.S. Department of Education through Award #2229873 - National AI Institute for Exceptional Education. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation, the Institute of Education Sciences, or the U.S. Department of Education.

REFERENCES

[1] Katrin Affolter, Kurt Stockinger, and Abraham Bernstein. 2019. A comparative survey of recent natural language interfaces for databases. *The VLDB Journal* 28 (2019), 793–819.

[2] Maneesh Agrawala. 2023. https://magrawala.substack.com/p/unpredictable-black-boxes-are-terrible

[3] Maneesh Agrawala, Doantam Phan, Julie Heiser, John Haymaker, Jeff Klingner, Pat Hanrahan, and Barbara Tversky. 2003. Designing effective step-by-step assembly instructions. *ACM Transactions on Graphics (TOG)* 22, 3 (2003), 828–837.

[4] Saleema Amershi, Dan Weld, Mihaela Vorvoreanu, Adam Fourney, Besmira Nushi, Penny Collisson, Jina Suh, Shamsi Iqbal, Paul N Bennett, Kori Inkpen, et al. 2019. Guidelines for human-AI interaction. In *Proceedings of the 2019 chi conference on human factors in computing systems*. 1–13.

[5] Tyler Angert, Miroslav Ivan Suzara, Jenny Han, Christopher Lawrence Pondoc, and Hariharan Subramonyam. 2023. Spellburst: A Node-based Interface for Exploratory Creative Coding with Natural Language Prompts. *arXiv preprint arXiv:2308.03921* (2023).

[6] Anthropic. 2023. Claude. https://claude.ai/

[7] Anysphere. 2023. Cursor. https://www.cursor.so/

[8] Franscesca Bacci, Federico Maria Cau, and Lucio Davide Spano. 2020. Inspecting data using natural language queries. In *Computational Science and Its Applications–ICCSA 2020: 20th International Conference, Cagliari, Italy, July 1–4, 2020, Proceedings, Part VI 20*. Springer, 771–782.

[9] Mikhail Mikhaĭ Bakhtin. [n. d.]. *The dialogic imagination: Four essays*.

[10] Jeanne Bamberger and Donald A Schön. 1983. Learning as reflective conversation with materials: Notes from work in progress. *Art Education* 36, 2 (1983), 68–73.

[11] Yavar Bathaee. 2017. The artificial intelligence black box and the failure of intent and causation. *Harv. JL & Tech.* 31 (2017), 889.

[12] Piraye Bayman and Richard E Mayer. 1984. Instructional manipulation of users' mental models for electronic calculators. *International Journal of Man-Machine Studies* 20, 2 (1984), 189–199.

[13] Emily M. Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. 2021. On the Dangers of Stochastic Parrots: Can Language Models Be Too Big?. In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency* (Virtual Event, Canada) *(FAccT '21)*. Association for Computing Machinery, New York, NY, USA, 610–623. https://doi.org/10.1145/3442188.3445922

[14] Olav W Bertelsen and Susanne Bødker. 2003. Activity theory. *HCI models, theories, and frameworks: Toward a multidisciplinary science* (2003), 291–324.

[15] Umang Bhatt, Javier Antorán, Yunfeng Zhang, Q. Vera Liao, Prasanna Sattigeri, Riccardo Fogliato, Gabrielle Gauthier Melançon, Ranganath Krishnan, Jason Stanley, Omesh Tickoo, Lama Nachman, Rumi Chunara, Madhulika Srikumar, Adrian Weller, and Alice Xiang. 2021. Uncertainty as a Form of Transparency: Measuring, Communicating, and Using Uncertainty. arXiv:2011.07586 [cs.CY]

[16] Bernd Bohnet, Vinh Tran, Pat Verga, Roee Aharoni, Daniel Andor, Livio Baldini Soares, Massimiliano Ciaramita, Jacob Eisenstein, Kuzman Ganchev, Jonathan Herzig, Kai Hui, Tom Kwiatkowski, Ji Ma, Jianmo Ni, Tal Schuster, Lierni Sestorain Saralegui, William Weston Cohen, Michael Collins, Dipanjan Das, Don Metzler, Slav Petrov, and Kellie Webster. 2022. Attributed Question Answering: Evaluation and Modeling for Attributed Large Language Models. https://arxiv.org/abs/2212.08037

[17] Tolga Bolukbasi, Kai-Wei Chang, James Y Zou, Venkatesh Saligrama, and Adam T Kalai. 2016. Man is to computer programmer as woman is to homemaker? debiasing word embeddings. *Advances in neural information processing systems* 29 (2016).

[18] Matthew M Botvinick. 2008. Hierarchical models of behavior and prefrontal function. *Trends in cognitive sciences* 12, 5 (2008), 201–208.

[19] Summer L Brandt, Joel Lachter, Ricky Russell, and Robert Jay Shively. 2018. A human-autonomy teaming approach for a flight-following task. In *Advances in Neuroergonomics and Cognitive Engineering: Proceedings of the AHFE 2017 International Conference on Neuroergonomics and Cognitive Engineering, July 17–21, 2017, The Westin Bonaventure Hotel, Los Angeles, California, USA 8*. Springer, 12–22.

[20] Holly P Branigan, Martin J Pickering, Jamie Pearson, and Janet F McLean. 2010. Linguistic alignment between people and computers. *Journal of pragmatics* 42, 9 (2010), 2355–2368.

[21] Ann L Brown. 2017. Metacognitive development and reading. In *Theoretical issues in reading comprehension*. Routledge, 453–482.

[22] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners. arXiv:2005.14165 [cs.CL]

[23] Aylin Caliskan, Joanna J. Bryson, and Arvind Narayanan. 2017. Semantics derived automatically from language corpora contain human-like biases. *Science* 356, 6334 (2017), 183–186. https://doi.org/10.1126/science.aal4230 arXiv:https://www.science.org/doi/pdf/10.1126/science.aal4230

[24] John M Carroll and Judith Reitman Olson. 1988. Mental models in human-computer interaction. *Handbook of human-computer interaction* (1988), 45–65.

[25] Stephen Casper, Jason Lin, Joe Kwon, Gatlen Culp, and Dylan Hadfield-Menell. 2023. Explore, Establish, Exploit: Red Teaming Language Models from Scratch. *arXiv preprint arXiv:2306.09442* (2023).

[26] Jiuhai Chen, Lichang Chen, Heng Huang, and Tianyi Zhou. 2023. When do you need Chain-of-Thought Prompting for ChatGPT? arXiv:2304.03262 [cs.AI]

[27] Xiang'Anthony' Chen, Jeff Burke, Ruofei Du, Matthew K Hong, Jennifer Jacobs, Philippe Laban, Dingzeyu Li, Nanyun Peng, Karl DD Willis, Chien-Sheng Wu, et al. 2023. Next Steps for Human-Centered Generative AI: A Technical Perspective. *arXiv preprint arXiv:2306.15774* (2023).

[28] Richard E Clark, David F Feldon, Jeroen JG Van Merrienboer, Kenneth A Yates, and Sean Early. 2008. Cognitive task analysis. In *Handbook of research on educational communications and technology*. Routledge, 577–593.

[29] Russell Cropanzano and Marie S Mitchell. 2005. Social exchange theory: An interdisciplinary review. *Journal of management* 31, 6 (2005), 874–900.

[30] Nigel Cross. 2001. Design cognition: Results from protocol and other empirical studies of design activity. *Design knowing and learning: Cognition in design education* (2001), 79–103.

[31] Mihaly Csikszentmihalyi and Jacob W Getzels. 1971. Discovery-oriented behavior and the originality of creative products: A study with artists. *Journal of personality and social psychology* 19, 1 (1971), 47.

[32] Mihaly Csikszentmihalyi and Jacob W Getzels. 1988. Creativity and problem finding in art. (1988).

[33] Can Cui, Yunsheng Ma, Xu Cao, Wenqian Ye, and Ziran Wang. 2023. Drive as you speak: Enabling human-like interaction with large language models in autonomous vehicles. *arXiv preprint arXiv:2309.10228* (2023).

[34] Clarisse Sieckenius De Souza. 2005. *The semiotic engineering of human-computer interaction*. MIT press.

[35] Jean Decety and Julie Grèzes. 2006. The power of simulation: Imagining one's own and other's behavior. *Brain research* 1079, 1 (2006), 4–14.

[36] Victor Dibia. 2023. LIDA: A Tool for Automatic Generation of Grammar-Agnostic Visualizations and Infographics using Large Language Models. *arXiv preprint arXiv:2303.02927* (2023).

[37] Danica Dillion, Niket Tandon, Yuling Gu, and Kurt Gray. 2023. Can AI language models replace human participants? *Trends in Cognitive Sciences* (2023).

[38] Judit Dombi, Tetyana Sydorenko, and Veronika Timpe-Laughlin. 2022. Common ground, cooperation, and recipient design in human-computer interactions. *Journal of Pragmatics* 193 (2022), 4–20.

[39] Kees Dorst. 2011. The core of 'design thinking'and its application. *Design studies* 32, 6 (2011), 521–532.

[40] Kees Dorst and Nigel Cross. 2001. Creativity in the design process: co-evolution of problem–solution. *Design studies* 22, 5 (2001), 425–437.

[41] Karl Duncker. 1945. On problem-solving.(Psychological Monographs, No. 270.). (1945).

[42] David W Ecker. 1963. The artistic process as qualitative problem solving. *The Journal of Aesthetics and Art Criticism* 21, 3 (1963), 283–290.

[43] Alex Endert, Remco Chang, Chris North, and Michelle Zhou. 2015. Semantic interaction: Coupling cognition and computation through usable interactive analytics. *IEEE Computer Graphics and Applications* 35, 4 (2015), 94–99.

[44] Umer Farooq and Jonathan Grudin. 2016. Human-computer integration. *interactions* 23, 6 (2016), 26–32.

[45] Shangbin Feng, Chan Young Park, Yuhan Liu, and Yulia Tsvetkov. 2023. From Pretraining Data to Language Models to Downstream Tasks: Tracking the Trails of Political Biases Leading to Unfair NLP Models. *arXiv preprint arXiv:2305.08283* (2023).

[46] Emilio Ferrara. 2023. Should chatgpt be biased? challenges and risks of bias in large language models. *arXiv preprint arXiv:2304.03738* (2023).

[47] Alexander J. Fiannaca, Chinmay Kulkarni, Carrie J Cai, and Michael Terry. 2023. Programming without a Programming Language: Challenges and Opportunities for Designing Developer Tools for Prompt Programming. In *Extended Abstracts*

of the 2023 CHI Conference on Human Factors in Computing Systems (Hamburg, Germany) *(CHI EA '23)*. Association for Computing Machinery, New York, NY, USA, Article 235, 7 pages. https://doi.org/10.1145/3544549.3585737

[48] Raymond Fok and Daniel S Weld. 2023. In Search of Verifiability: Explanations Rarely Enable Complementary Performance in AI-Advised Decision Making. *arXiv preprint arXiv:2305.07722* (2023).

[49] Asbjørn Følstad and Petter Bae Brandtzæg. 2017. Chatbots and the new world of HCI. *interactions* 24, 4 (2017), 38–42.

[50] Chris Frith and Uta Frith. 2005. Theory of mind. *Current biology* 15, 17 (2005), R644–R645.

[51] Yao Fu, Hao Peng, Ashish Sabharwal, Peter Clark, and Tushar Khot. 2023. Complexity-Based Prompting for Multi-Step Reasoning. arXiv:2210.00720 [cs.CL]

[52] Tong Gao, Mira Dontcheva, Eytan Adar, Zhicheng Liu, and Karrie G. Karahalios. 2015. DataTone: Managing Ambiguity in Natural Language Interfaces for Data Visualization. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology* (Charlotte, NC, USA) *(UIST '15)*. Association for Computing Machinery, New York, NY, USA, 489–500. https://doi.org/10.1145/2807442.2807478

[53] James J Gibson. 1977. The theory of affordances. *Hilldale, USA* 1, 2 (1977), 67–82.

[54] Github. 2023. Github Copilot. https://github.com/features/copilot

[55] Peter M Gollwitzer and Gabriele Oettingen. 2020. Implementation intentions. In *Encyclopedia of behavioral medicine*. Springer, 1159–1164.

[56] Charles Goodwin and John Heritage. 1990. Conversation analysis. *Annual review of anthropology* 19, 1 (1990), 283–307.

[57] Google. 2023. Bard. https://bard.google.com/

[58] Herbert P Grice. 1975. Logic and conversation. In *Speech acts*. Brill, 41–58.

[59] Joy Paul Guilford. 1956. The structure of intellect. *Psychological bulletin* 53, 4 (1956), 267.

[60] Andrew B Hargadon and Beth A Bechky. 2006. When collections of creatives become creative collectives: A field study of problem solving at work. *Organization science* 17, 4 (2006), 484–500.

[61] John R Hayes. 2013. A new framework for understanding cognition and affect in writing. In *The science of writing*. Routledge, 1–27.

[62] Robert R Hoffman, Gary Klein, and Shane T Mueller. 2018. Explaining explanation for "explainable AI". In *Proceedings of the human factors and ergonomics society annual meeting*, Vol. 62. SAGE Publications Sage CA: Los Angeles, CA, 197–201.

[63] Ari Holtzman, Peter West, Vered Shwartz, Yejin Choi, and Luke Zettlemoyer. 2021. Surface form competition: Why the highest probability answer isn't always right. *arXiv preprint arXiv:2104.08315* (2021).

[64] Enamul Hoque, Vidya Setlur, Melanie Tory, and Isaac Dykeman. 2018. Applying Pragmatics Principles for Interaction with Visual Analytics. *IEEE Transactions on Visualization and Computer Graphics* 24, 1 (2018), 309–318. https://doi.org/10.1109/TVCG.2017.2744684

[65] Kasper Hornbæk and Antti Oulasvirta. 2017. What is interaction?. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. 5040–5052.

[66] Thomas J Howard, Stephen J Culley, and Elies Dekoninck. 2008. Describing the creative design process by the integration of engineering design and cognitive psychology literature. *Design studies* 29, 2 (2008), 160–180.

[67] Edwin Hutchins. 1987. *Metaphors for interface design*. Institute for Cognitive Science, University of California, San Diego.

[68] Edwin L Hutchins, James D Hollan, and Donald A Norman. 1985. Direct manipulation interfaces. *Human–computer interaction* 1, 4 (1985), 311–338.

[69] Saki Imai. 2022. Is GitHub Copilot a Substitute for Human Pair-Programming? An Empirical Study. In *Proceedings of the ACM/IEEE 44th International Conference on Software Engineering: Companion Proceedings* (Pittsburgh, Pennsylvania) *(ICSE '22)*. Association for Computing Machinery, New York, NY, USA, 319–321. https://doi.org/10.1145/3510454.3522684

[70] Michael Jackson. 1995. *Software Requirements & Specifications: a lexicon of practice, principles and prejudices*. ACM Press/Addison-Wesley Publishing Co.

[71] David G Jansson and Steven M Smith. 1991. Design fixation. *Design studies* 12, 1 (1991), 3–11.

[72] Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. 2023. Survey of Hallucination in Natural Language Generation. *ACM Comput. Surv.* 55, 12, Article 248 (mar 2023), 38 pages. https://doi.org/10.1145/3571730

[73] Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. 2023. Survey of hallucination in natural language generation. *Comput. Surveys* 55, 12 (2023), 1–38.

[74] Peiling Jiang, Jude Rayan, Steven P Dow, and Haijun Xia. 2023. Graphologue: Exploring Large Language Model Responses with Interactive Diagrams. *arXiv preprint arXiv:2305.11473* (2023).

[75] Jean Kaddour, Joshua Harris, Maximilian Mozes, Herbie Bradley, Roberta Raileanu, and Robert McHardy. 2023. Challenges and Applications of Large Language Models. *arXiv preprint arXiv:2307.10169* (2023).

[76] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling Laws for Neural Language Models. arXiv:2001.08361 [cs.LG]

[77] Jan-Frederik Kassel and Michael Rohs. 2018. Valletto: A multimodal interface for ubiquitous visual analytics. In *Extended Abstracts of the 2018 CHI Conference on Human Factors in Computing Systems*. 1–6.

[78] Majeed Kazemitabaar, Justin Chow, Carl Ka To Ma, Barbara J Ericson, David Weintrop, and Tovi Grossman. 2023. Studying the effect of AI Code Generators on Supporting Novice Learners in Introductory Programming. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*. 1–23.

[79] Istvan Kecskes. 2010. The paradox of communication: Socio-cognitive approach to pragmatics. *Pragmatics and Society* 1, 1 (2010), 50–73.

[80] Yoonsu Kim, Jueon Lee, Seoyoung Kim, Jaehyuk Park, and Juho Kim. 2023. Understanding Users' Dissatisfaction with ChatGPT Responses: Types, Resolving Tactics, and the Effect of Knowledge Level. *arXiv preprint arXiv:2311.07434* (2023).

[81] Amy J Ko, Brad A Myers, and Htet Htet Aung. 2004. Six learning barriers in end-user programming systems. In *2004 IEEE Symposium on Visual Languages-Human Centric Computing*. IEEE, 199–206.

[82] Jan Kocoń, Igor Cichecki, Oliwier Kaszyca, Mateusz Kochanek, Dominika Szydło, Joanna Baran, Julita Bielaniewicz, Marcin Gruza, Arkadiusz Janz, Kamil Kanclerz, et al. 2023. ChatGPT: Jack of all trades, master of none. *Information Fusion* (2023), 101861.

[83] Shunsuke Koga. 2023. Exploring the Pitfalls of Large Language Models: Inconsistency and Inaccuracy in Answering Pathology Board Examination-Style Questions. *medRxiv* (2023), 2023–08.

[84] Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2023. Large Language Models are Zero-Shot Reasoners. arXiv:2205.11916 [cs.CL]

[85] Kristin M Kostick-Quenet, I Glenn Cohen, Sara Gerke, Bernard Lo, James Antaki, Faezah Movahedi, Hasna Njah, Lauren Schoen, Jerry E Estep, and JS Blumenthal-Barby. 2022. Mitigating racial bias in machine learning. *Journal of Law, Medicine & Ethics* 50, 1 (2022), 92–100.

[86] Andrew Kuznetsov, Joseph Chee Chang, Nathan Hahn, Napol Rachatasumrit, Bradley Breneisen, Julina Coupland, and Aniket Kittur. 2022. Fuse: In-Situ Sensemaking Support in the Browser. In *Proceedings of the 35th Annual ACM Symposium on User Interface Software and Technology*. 1–15.

[87] Woo kyoung Ahn, William F. Brewer, and Raymond J. Mooney. 1992. Schema acquisition from a single example. *Journal of Experimental Psychology: Learning, Memory, and Cognition* 18, 2 (mar 1992), 391–412. https://doi.org/10.1037/0278-7393.18.2.391

[88] Brenden M. Lake, Tal Linzen, and Marco Baroni. 2019. Human few-shot learning of compositional instructions. arXiv:1901.04587 [cs.CL]

[89] Brenden M. Lake, Tomer D. Ullman, Joshua B. Tenenbaum, and Samuel J. Gershman. 2016. Building Machines That Learn and Think Like People. arXiv:1604.00289 [cs.AI]

[90] Andrew K. Lampinen, Ishita Dasgupta, Stephanie C. Y. Chan, Kory Matthewson, Michael Henry Tessler, Antonia Creswell, James L. McClelland, Jane X. Wang, and Felix Hill. 2022. Can language models learn from explanations in context? arXiv:2204.02329 [cs.CL]

[91] Gierad P Laput, Mira Dontcheva, Gregg Wilensky, Walter Chang, Aseem Agarwala, Jason Linder, and Eytan Adar. 2013. Pixeltone: A multimodal interface for image editing. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 2185–2194.

[92] Karl Spencer Lashley et al. 1951. *The problem of serial order in behavior*. Vol. 21. Bobbs-Merrill Oxford.

[93] Keelin Leahy, Shanna R Daly, Seda McKilligan, and Colleen M Seifert. 2020. Design fixation from initial examples: Provided versus self-generated ideas. *Journal of Mechanical Design* 142, 10 (2020), 101402.

[94] Peter Lee, Sebastien Bubeck, and Joseph Petro. 2023. Benefits, limits, and risks of GPT-4 as an AI chatbot for medicine. *New England Journal of Medicine* 388, 13 (2023), 1233–1239.

[95] Jiachang Liu, Dinghan Shen, Yizhe Zhang, Bill Dolan, Lawrence Carin, and Weizhu Chen. 2021. What Makes Good In-Context Examples for GPT-3? arXiv:2101.06804 [cs.CL]

[96] Nelson F Liu, Tianyi Zhang, and Percy Liang. 2023. Evaluating verifiability in generative search engines. *arXiv preprint arXiv:2304.09848* (2023).

[97] Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021. Pre-train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing. arXiv:2107.13586 [cs.CL]

[98] Yi Liu, Gelei Deng, Zhengzi Xu, Yuekang Li, Yaowen Zheng, Ying Zhang, Lida Zhao, Tianwei Zhang, and Yang Liu. 2023. Jailbreaking chatgpt via prompt engineering: An empirical study. *arXiv preprint arXiv:2305.13860* (2023).

[99] Tania Lombrozo and Susan Carey. 2006. Functional explanation and the function of explanation. *Cognition* 99, 2 (2006), 167–204. https://doi.org/10.1016/j.cognition.2004.12.009

[100] Joseph B Lyons, Sean Mahoney, Kevin T Wynne, and Mark A Roebke. 2018. Viewing machines as teammates: A qualitative study. In *2018 AAAI Spring*

Symposium Series.
[101] Norman RF Maier. 1931. Reasoning in humans. II. The solution of a problem and its appearance in consciousness. *Journal of comparative Psychology* 12, 2 (1931), 181.
[102] Sherin Mary Mathews. 2019. Explainable artificial intelligence applications in NLP, biomedical, and malware classification: A literature review. In *Intelligent Computing: Proceedings of the 2019 Computing Conference, Volume 2*. Springer, 1269–1292.
[103] Richard E Mayer. 1981. The psychology of how novices learn computer programming. *ACM Computing Surveys (CSUR)* 13, 1 (1981), 121–141.
[104] Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan McDonald. 2020. On Faithfulness and Factuality in Abstractive Summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Online, 1906–1919. https://doi.org/10.18653/v1/2020.acl-main.173
[105] Lauren McCarthy. 2023. p5.js. https://p5js.org/
[106] Microsoft. 2023. Visual Studio Code. https://code.visualstudio.com/
[107] George A Miller. 1956. The magical number seven, plus or minus two: Some limits on our capacity for processing information. *Psychological review* 63, 2 (1956), 81.
[108] George A Miller, Galanter Eugene, and Karl H Pribram. 2017. Plans and the Structure of Behaviour. In *Systems Research for Behavioral Science*. Routledge, 369–382.
[109] George A Miller, Eugene Galanter, and Karl H Pribram. 1960. *Plans and the structure of behavior. New York, NY: Henry Holt and Co.* Inc.
[110] Suvir Mirchandani, Fei Xia, Pete Florence, Brian Ichter, Danny Driess, Montserrat Gonzalez Arenas, Kanishka Rao, Dorsa Sadigh, and Andy Zeng. 2023. Large Language Models as General Pattern Machines. arXiv:2307.04721 [cs.AI]
[111] Sina Mohseni, Niloofar Zarei, and Eric D Ragan. 2021. A multidisciplinary survey and framework for design and evaluation of explainable AI systems. *ACM Transactions on Interactive Intelligent Systems (TiiS)* 11, 3-4 (2021), 1–45.
[112] Neville Moray. 1987. Intelligent aids, mental models, and the theory of machines. *International journal of man-machine studies* 27, 5-6 (1987), 619–629.
[113] Meredith Ringel Morris, Jascha Sohl-dickstein, Noah Fiedel, Tris Warkentin, Allan Dafoe, Aleksandra Faust, Clement Farabet, and Shane Legg. 2023. Levels of AGI: Operationalizing Progress on the Path to AGI. *arXiv preprint arXiv:2311.02462* (2023).
[114] Robert Munro, Steven Bethard, Victor Kuperman, Vicky Tzuyin Lai, Robin Melnick, Christopher Potts, Tyler Schnoebelen, and Harry Tily. 2010. Crowdsourcing and language studies: the new generation of linguistic data. In *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*. Association for Computational Linguistics, Los Angeles, 122–130. https://aclanthology.org/W10-0719
[115] Moin Nadeem, Anna Bethke, and Siva Reddy. 2020. StereoSet: Measuring stereotypical bias in pretrained language models. *arXiv preprint arXiv:2004.09456* (2020).
[116] Dennis C Neale and John M Carroll. 1997. The role of metaphors in user interface design. In *Handbook of human-computer interaction*. Elsevier, 441–462.
[117] Allen Newell and Herbert A Simon. 1961. Computer Simulation of Human Thinking: A theory of problem solving expressed as a computer program permits simulation of thinking processes. *Science* 134, 3495 (1961), 2011–2017.
[118] Allen Newell and Herbert A Simon. 2007. Computer science as empirical inquiry: Symbols and search. In *ACM Turing award lectures*. 1975.
[119] Allen Newell, Herbert Alexander Simon, et al. 1972. *Human problem solving*. Vol. 104. Prentice-hall Englewood Cliffs, NJ.
[120] Donald A Norman. 1986. Cognitive engineering. *User centered system design* 31, 61 (1986), 2.
[121] Donald A Norman. 2014. Some observations on mental models. In *Mental models*. Psychology Press, 15–22.
[122] Notion. 2023. Notion. https://www.notion.so
[123] David Oniani, Jordan Hilsman, Yifan Peng, Ronald K Poropatich, COL Pamplin, LTC Legault, Yanshan Wang, et al. 2023. From Military to Healthcare: Adopting and Expanding Ethical Principles for Generative Artificial Intelligence. *arXiv preprint arXiv:2308.02448* (2023).
[124] OpenAI. 2023. ChatGPT. https://chat.openai.com/chat
[125] OpenAI. 2023. GPT-4 Technical Report. arXiv:2303.08774 [cs.CL]
[126] Antti Oulasvirta, Jussi PP Jokinen, and Andrew Howes. 2022. Computational rationality as a theory of interaction. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*. 1–14.
[127] Fatma Özcan, Abdul Quamar, Jaydeep Sen, Chuan Lei, and Vasilis Efthymiou. 2020. State of the art and open challenges in natural language interfaces to data. In *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*. 2629–2636.
[128] Andrea L. Patalano and Colleen M. Seifert. 1997. Opportunistic Planning: Being Reminded of Pending Goals. *Cognitive Psychology* 34, 1 (1997), 1–36. https://doi.org/10.1006/cogp.1997.0655
[129] Roy D Pea. 1982. What is planning development the development of? *New Directions for Child and Adolescent Development* 1982, 18 (1982), 5–27.
[130] Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A. Smith, and Mike Lewis. 2023. Measuring and Narrowing the Compositionality Gap in Language Models. arXiv:2210.03350 [cs.CL]
[131] Ben Prystawski, Paul Thibodeau, Christopher Potts, and Noah D. Goodman. 2023. Psychologically-informed chain-of-thought prompts for metaphor understanding in large language models. arXiv:2209.08141 [cs.CL]
[132] A Terry Purcell and John S Gero. 1996. Design and other types of fixation. *Design studies* 17, 4 (1996), 363–383.
[133] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. 2022. Hierarchical text-conditional image generation with clip latents, 2022. *URL https://arxiv. org/abs/2204.06125* 7 (2022).
[134] Hannah Rashkin, Vitaly Nikolaev, Matthew Lamm, Lora Aroyo, Michael Collins, Dipanjan Das, Slav Petrov, Gaurav Singh Tomar, Iulia Turc, and David Reitter. 2023. Measuring attribution in natural language generation models. *Computational Linguistics* (2023), 1–66.
[135] Vipula Rawte, Amit Sheth, and Amitava Das. 2023. A Survey of Hallucination in Large Foundation Models. arXiv:2309.05922 [cs.AI]
[136] Byron Reeves and Clifford Nass. 1996. The media equation: How people treat computers, television, and new media like real people. *Cambridge, UK* 10, 10 (1996).
[137] Replit. 2023. Replit. https://replit.com/
[138] John Restrepo and Henri Christiaans. 2004. Problem structuring and information access in design. *Journal of Design Research* 4, 2 (2004), 218–236.
[139] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 2022. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 10684–10695.
[140] Malik Sallam. 2023. ChatGPT utility in healthcare education, research, and practice: systematic review on the promising perspectives and valid concerns. In *Healthcare*, Vol. 11. MDPI, 887.
[141] Arnold Sameroff. 2009. *The transactional model*. American Psychological Association.
[142] Victor Sanh, Albert Webson, Colin Raffel, Stephen H. Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, Manan Dey, M Saiful Bari, Canwen Xu, Urmish Thakker, Shanya Sharma Sharma, Eliza Szczechla, Taewoon Kim, Gunjan Chhablani, Nihal Nayak, Debajyoti Datta, Jonathan Chang, Mike Tian-Jian Jiang, Han Wang, Matteo Manica, Sheng Shen, Zheng Xin Yong, Harshit Pandey, Rachel Bawden, Thomas Wang, Trishala Neeraj, Jos Rozen, Abheesht Sharma, Andrea Santilli, Thibault Fevry, Jason Alan Fries, Ryan Teehan, Tali Bers, Stella Biderman, Leo Gao, Thomas Wolf, and Alexander M. Rush. 2022. Multitask Prompted Training Enables Zero-Shot Task Generalization. arXiv:2110.08207 [cs.LG]
[143] Abulhair Saparov and He He. 2022. Language models are greedy reasoners: A systematic formal analysis of chain-of-thought. *arXiv preprint arXiv:2210.01240* (2022).
[144] Roger C Schank and Robert P Abelson. 2013. *Scripts, plans, goals, and understanding: An inquiry into human knowledge structures*. Psychology Press.
[145] D Schon. 1983. Becoming a reflective practitioner. *How professionals think in action. London: Temple Smith* (1983).
[146] Gregory Schraw and Rayne Sperling Dennison. 1994. Assessing metacognitive awareness. *Contemporary educational psychology* 19, 4 (1994), 460–475.
[147] John R Searle. 1983. *Intentionality: An essay in the philosophy of mind*. Cambridge university press.
[148] John R Searle, Ferenc Kiefer, Manfred Bierwisch, et al. 1980. *Speech act theory and pragmatics*. Vol. 10. Springer.
[149] Colleen M Seifert, David E Meyer, Natalie Davidson, Andrea L Patalano, and Ilan Yaniv. 1994. Demystification of cognitive insight: Opportunistic assimilation and the prepared-mind hypothesis. (1994).
[150] Colleen M Seifert and Andrea L Patalano. 2001. Opportunism in memory: Preparing for chance encounters. *Current Directions in Psychological Science* 10, 6 (2001), 198–201.
[151] Vidya Setlur, Sarah E Battersby, Melanie Tory, Rich Gossweiler, and Angel X Chang. 2016. Eviza: A natural language interface for visual analysis. In *Proceedings of the 29th annual symposium on user interface software and technology*. 365–377.
[152] Vidya Setlur and Melanie Tory. 2022. How do you converse with an analytical chatbot? revisiting gricean maxims for designing analytical conversational behavior. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*. 1–17.
[153] Omar Shaikh, Hongxin Zhang, William Held, Michael Bernstein, and Diyi Yang. 2023. On Second Thought, Let's Not Think Step by Step! Bias and Toxicity in Zero-Shot Reasoning. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Toronto, Canada, 4454–4470. https://doi.org/10.18653/v1/2023.acl-long.244

[154] Leixian Shen, Enya Shen, Yuyu Luo, Xiaocong Yang, Xuming Hu, Xiongshuai Zhang, Zhiwei Tai, and Jianmin Wang. 2022. Towards natural language interfaces for data visualization: A survey. *IEEE transactions on visualization and computer graphics* (2022).

[155] Emily Sheng, Kai-Wei Chang, Premkumar Natarajan, and Nanyun Peng. 2019. The Woman Worked as a Babysitter: On Biases in Language Generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Association for Computational Linguistics, Hong Kong, China, 3407–3412. https://doi.org/10.18653/v1/D19-1339

[156] Ben Shneiderman. 1982. The future of interactive systems and the emergence of direct manipulation. *Behaviour & Information Technology* 1, 3 (1982), 237–256.

[157] Herbert A Simon. 1956. Rational choice and the structure of the environment. *Psychological review* 63, 2 (1956), 129.

[158] Herbert Alexander Simon. 1997. *Models of bounded rationality: Empirically grounded economic reason*. Vol. 3. MIT press.

[159] Herbert A Simon and Allen Newell. 1971. Human problem solving: The state of the theory in 1970. *American psychologist* 26, 2 (1971), 145.

[160] Arjun Srinivasan, Bongshin Lee, and John Stasko. 2020. Interweaving multimodal interaction with flexible unit visualizations for data exploration. *IEEE Transactions on Visualization and Computer Graphics* 27, 8 (2020), 3519–3533.

[161] Arjun Srinivasan and John Stasko. 2020. How to Ask What to Say?: Strategies for Evaluating Natural Language Interfaces for Data Visualization. *IEEE Computer Graphics and Applications* 40, 4 (2020), 96–103. https://doi.org/10.1109/MCG.2020.2986902

[162] Nancy Staggers and Anthony F. Norcio. 1993. Mental models: concepts for human-computer interaction research. *International Journal of Man-machine studies* 38, 4 (1993), 587–605.

[163] Keith E. Stanovich and Richard F. West. 2000. Individual differences in reasoning: Implications for the rationality debate? *Behavioral and Brain Sciences* 23, 5 (oct 2000), 645–665. https://doi.org/10.1017/s0140525x00003435

[164] Matthew Stone. 2005. Communicative intentions and conversational processes in humanhuman and human-computer dialogue. *Approaches to studying world-situated language use* (2005), 39–70.

[165] Hariharan Subramonyam, Jane Im, Colleen Seifert, and Eytan Adar. 2022. Solving separation-of-concerns problems in collaborative design of human-AI systems through leaky abstractions. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*. 1–21.

[166] Hariharan Subramonyam, Wilmot Li, Eytan Adar, and Mira Dontcheva. 2018. Taketoons: Script-driven performance animation. In *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology*. 663–674.

[167] Hariharan Subramonyam, Colleen Seifert, and Eytan Adar. 2021. Towards a process model for co-creating AI experiences. In *Designing Interactive Systems Conference 2021*. 1529–1543.

[168] Hariharan Subramonyam, Colleen Seifert, Priti Shah, and Eytan Adar. 2020. Texsketch: Active diagramming through pen-and-ink annotations. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. 1–13.

[169] Sangho Suh, Bryan Min, Srishti Palani, and Haijun Xia. 2023. Sensecape: Enabling Multilevel Exploration and Sensemaking with Large Language Models. *arXiv preprint arXiv:2305.11483* (2023).

[170] Jiao Sun, Q. Vera Liao, Michael Muller, Mayank Agarwal, Stephanie Houde, Kartik Talamadupula, and Justin D. Weisz. 2022. Investigating Explainability of Generative AI for Code through Scenario-based Design. arXiv:2202.04903 [cs.HC]

[171] Alaina N Talboy and Elizabeth Fuller. 2023. Challenging the appearance of machine intelligence: Cognitive bias in LLMs. *arXiv preprint arXiv:2304.01358* (2023).

[172] Jenifer Tidwell. 2010. *Designing interfaces: Patterns for effective interaction design*. " O'Reilly Media, Inc.".

[173] Tomer Ullman. 2023. Large language models fail on trivial alterations to theory-of-mind tasks. *arXiv preprint arXiv:2302.08399* (2023).

[174] Helena Vasconcelos, Gagan Bansal, Adam Fourney, Q. Vera Liao, and Jennifer Wortman Vaughan. 2023. Generation Probabilities Are Not Enough: Exploring the Effectiveness of Uncertainty Highlighting in AI-Powered Code Completions. arXiv:2302.07248 [cs.HC]

[175] Boxin Wang, Weixin Chen, Hengzhi Pei, Chulin Xie, Mintong Kang, Chenhui Zhang, Chejian Xu, Zidi Xiong, Ritik Dutta, Rylan Schaeffer, et al. 2023. DecodingTrust: A Comprehensive Assessment of Trustworthiness in GPT Models. *arXiv preprint arXiv:2306.11698* (2023).

[176] Boshi Wang, Sewon Min, Xiang Deng, Jiaming Shen, You Wu, Luke Zettlemoyer, and Huan Sun. 2023. Towards Understanding Chain-of-Thought Prompting: An Empirical Study of What Matters. arXiv:2212.10001 [cs.CL]

[177] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023. Self-Consistency Improves Chain of Thought Reasoning in Language Models. arXiv:2203.11171 [cs.CL]

[178] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed Chi, Quoc V Le, and Denny Zhou. 2022. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. In *Advances in Neural Information Processing Systems*, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (Eds.), Vol. 35. Curran Associates, Inc., 24824–24837. https://proceedings.neurips.cc/paper_files/paper/2022/file/9d5609613524ecf4f15af0f7b31abca4-Paper-Conference.pdf

[179] Justin D Weisz, Michael Muller, Jessica He, and Stephanie Houde. 2023. Toward general design principles for generative AI applications. *arXiv preprint arXiv:2301.05578* (2023).

[180] Jules White, Quchen Fu, Sam Hays, Michael Sandborn, Carlos Olea, Henry Gilbert, Ashraf Elnashar, Jesse Spencer-Smith, and Douglas C. Schmidt. 2023. A Prompt Pattern Catalog to Enhance Prompt Engineering with ChatGPT. arXiv:2302.11382 [cs.SE]

[181] Christopher D Wickens, Justin G Hollands, Simon Banbury, and Raja Parasuraman. 2015. *Engineering psychology and human performance*. Psychology Press.

[182] Merlin C Wittrock. 1989. Generative processes of comprehension. *Educational psychologist* 24, 4 (1989), 345–376.

[183] Larry E Wood. 1997. *User interface design: Bridging the gap from user requirements to design*. CRC Press.

[184] Austin P Wright, Zijie J Wang, Haekyu Park, Grace Guo, Fabian Sperrle, Mennatallah El-Assady, Alex Endert, Daniel Keim, and Duen Horng Chau. 2020. A comparative analysis of industry human-AI interaction guidelines. *arXiv preprint arXiv:2010.11761* (2020).

[185] Tongshuang Wu, Ellen Jiang, Aaron Donsbach, Jeff Gray, Alejandra Molina, Michael Terry, and Carrie J Cai. 2022. Promptchainer: Chaining large language model prompts through visual programming. In *CHI Conference on Human Factors in Computing Systems Extended Abstracts*. 1–10.

[186] Tongshuang Wu, Michael Terry, and Carrie Jun Cai. 2022. Ai chains: Transparent and controllable human-ai interaction by chaining large language model prompts. In *Proceedings of the 2022 CHI conference on human factors in computing systems*. 1–22.

[187] Canwen Xu, Julian McAuley, and Penghan Wang. 2023. Mirror: A Natural Language Interface for Data Querying, Summarization, and Visualization. In *Companion Proceedings of the ACM Web Conference 2023*. 49–52.

[188] Jingfeng Yang, Hongye Jin, Ruixiang Tang, Xiaotian Han, Qizhang Feng, Haoming Jiang, Bing Yin, and Xia Hu. 2023. Harnessing the Power of LLMs in Practice: A Survey on ChatGPT and Beyond. arXiv:2304.13712 [cs.CL]

[189] Qian Yang, Justin Cranshaw, Saleema Amershi, Shamsi T Iqbal, and Jaime Teevan. 2019. Sketching nlp: A case study of exploring the right things to design with language intelligence. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. 1–12.

[190] Qian Yang, Alex Scuito, John Zimmerman, Jodi Forlizzi, and Aaron Steinfeld. 2018. Investigating how experienced UX designers effectively work with machine learning. In *Proceedings of the 2018 designing interactive systems conference*. 585–596.

[191] Qian Yang, Aaron Steinfeld, Carolyn Rosé, and John Zimmerman. 2020. Re-examining whether, why, and how human-AI interaction is uniquely difficult to design. In *Proceedings of the 2020 chi conference on human factors in computing systems*. 1–13.

[192] Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. Tree of Thoughts: Deliberate Problem Solving with Large Language Models. arXiv:2305.10601 [cs.CL]

[193] Samuel Yeom, Irene Giacomelli, Matt Fredrikson, and Somesh Jha. 2018. Privacy risk in machine learning: Analyzing the connection to overfitting. In *2018 IEEE 31st computer security foundations symposium (CSF)*. IEEE, 268–282.

[194] Richard M Young. 2014. Surrogates and mappings: Two kinds of conceptual models for interactive devices. In *Mental models*. Psychology Press, 43–60.

[195] Bowen Yu and Cláudio T Silva. 2019. FlowSense: A natural language interface for visual data exploration within a dataflow system. *IEEE transactions on visualization and computer graphics* 26, 1 (2019), 1–11.

[196] JD Zamfirescu-Pereira, Heather Wei, Amy Xiao, Kitty Gu, Grace Jung, Matthew G Lee, Bjoern Hartmann, and Qian Yang. 2023. Herding AI Cats: Lessons from Designing a Chatbot by Prompting GPT-3. (2023).

[197] JD Zamfirescu-Pereira, Richmond Y Wong, Bjoern Hartmann, and Qian Yang. 2023. Why Johnny can't prompt: how non-AI experts try (and fail) to design LLM prompts. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*. 1–21.

[198] Haiyan Zhao, Hanjie Chen, Fan Yang, Ninghao Liu, Huiqi Deng, Hengyi Cai, Shuaiqiang Wang, Dawei Yin, and Mengnan Du. 2023. Explainability for Large Language Models: A Survey. *arXiv preprint arXiv:2309.01029* (2023).

[199] Tony Z. Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. 2021. Calibrate Before Use: Improving Few-Shot Performance of Language Models. arXiv:2102.09690 [cs.CL]