# Improving Knowledge Extraction from LLMs for Task Learning through Agent Analysis

**James R. Kirk, Robert E. Wray, Peter Lindes, John E. Laird**

Center for Integrated Cognition at IQMRI
Ann Arbor, MI 48105 USA
{james.kirk,robert.wray,peter.lindes,john.laird}@cic.iqmri.org

## Abstract

Large language models (LLMs) offer significant promise as a knowledge source for task learning. Prompt engineering has been shown to be effective for eliciting knowledge from an LLM, but alone it is insufficient for acquiring relevant, situationally grounded knowledge for an embodied agent learning novel tasks. We describe a cognitive-agent approach, STARS, that extends and complements prompt engineering, mitigating its limitations and thus enabling an agent to acquire new task knowledge matched to its native language capabilities, embodiment, environment, and user preferences. The STARS approach is to increase the response space of LLMs and deploy general strategies, embedded within the autonomous agent, to evaluate, repair, and select among candidate responses produced by the LLM. We describe the approach and experiments that show how an agent, by retrieving and evaluating a breadth of responses from the LLM, can achieve $77-94\%$ task completion in one-shot learning without user oversight. The approach achieves $100\%$ task completion when human oversight (such as an indication of preference) is provided. Further, the type of oversight largely shifts from explicit, natural language instruction to simple confirmation/discomfirmation of high-quality responses that have been vetted by the agent before presentation to a user.

## Introduction

Prompt engineering (Reynolds and McDonell 2021), along with in-context learning (OpenAI 2023), has been shown to be an effective strategy for extracting knowledge from a large language model (LLM). However, embodied agents learning task knowledge (e.g., goals and actions) face far more stringent requirements. LLM responses must be:

1. Interpretable by the agent's parsing capabilities. LLM responses must be understandable by the agent, meaning grammar and terminology are presented in a form that the agent can actually process.
2. Situated to the agent's environment. Objects, features, and relations referenced in an LLM response must be perceivable and identifiable in the environment for the agent to ground the response successfully.
3. Matched to agent's embodiment and affordances. An LLM, trained on a large corpus describing human activities, will (generally) generate responses conforming with human embodiment and affordances. Responses that do not consider an agent's often non-human embodiment (e.g., a single-armed robot) will often be infeasible for that agent to execute.
4. Aligned with individual human preferences and values. Users will have individual expectations about how tasks should be performed and what constitutes appropriate outcomes in the current situation. Task success requires identifying and conforming to these preferences.

Copyright © 2024, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

The first three requirements are necessary for an embodied agent to use an LLM response to act in its world. We define responses that meet these requirements as *viable*. The final requirement is necessary to achieve the task as a specific human user prefers. A response is *situationally relevant* if it is viable *and* matches the user's preferences.

To attempt to elicit viable responses from the LLM, we previously (Kirk et al. 2023) employed a template-based prompting approach (TBP; Olmo, Sreedharan, and Kambhampati 2021; Kirk et al. 2022; Reynolds and McDonell 2021). We developed prompt templates that included examples of desired task knowledge, instantiated them with context from the current task, and retrieved multiple responses (varying LLM temperature to generate different responses). Unfortunately, this TBP strategy produced responses that often violated one or more of the first three requirements. Human feedback could be used to overcome these limitations, but required substantial input to correct responses (as well as to align them with agent needs and user preferences), making TBP impractical for an embodied agent.

Motivated by these inadequacies, we present a novel strategy: Search Tree, Analyze and Repair, and Selection (STARS). Similar to "agentic" uses of LLMs (Sumers et al. 2023; Park et al. 2023; Richards 2023), we employ the LLM as a component within a larger system. Like self-consistency (Wang et al. 2023), STARS generates a large space of responses from the LLM (multiple responses to a query). In contrast with the voting in self-consistency, the agent analyzes and evaluates each response for potential issues (e.g., mismatched embodiment, unknown words, ungrounded references). It attempts to repair problematic responses via targeted re-prompting of the LLM. To select among candidates, the agent queries the LLM for a "preferred" response. The STARS agent can still solicit human feedback, but the pri-

mary purpose of oversight is to ensure that agent behavior (and learning) incorporates user preferences.

To evaluate STARS against TBP, we embed both methods within an existing embodied agent (Mohan and Laird 2014; Mininger 2021; Kirk and Laird 2016). This agent uses interactive task learning (ITL; Laird et al. 2017; Gluck and Laird 2019) to learn novel tasks via natural language instruction from a human user. Instead of querying a human for a goal description of the task (e.g., "the goal is that the can is in the recycling bin"), the new agents (using TBP or STARS) access the LLM for that goal.

We compare STARS to TBP and also evaluate the individual components of STARS (i.e., Search Tree, Analysis & Repair, Selection) in a simulated robotic environment. We assess both task completion rate and the amount of oversight needed to achieve 100% task completion. We hypothesize STARS will eliminate the need to solicit human feedback for unviable responses, resulting in a much higher task completion rate (without oversight) and reducing how much oversight is required when human input is available.

As we show below, over three different tasks, STARS achieves 77-94% task completion without oversight (in comparison to 35-66% with TBP). With oversight, STARS reduces the number of words needed from the user by 52-68% (compared to TBP). Further, providing oversight is much simpler for the user. The user no longer needs to evaluate the viability of responses nor provide (many) goal descriptions; now, the user largely indicates preference, simply confirming or disconfirming from the LLM responses that the agent has determined to be viable. Finally, because the original ITL agent learns long-term task and subtask knowledge in one shot, this new agent also demonstrates one-shot performance: it achieves 100% task completion when prompted to perform the same task in the future, without accessing the LLM or requiring further human input.

## Related Work

Core features of our approach are 1) online task learning (no pre-training for domain or task), 2) the exploitation of multiple sources of knowledge, 3) proactive evaluation of LLM responses, and 4) one-shot task learning. We review related work in terms of these solution features.

Inner Monologue (Huang et al. 2022) modifies its prompts based on feedback from the environment, agent, and user to elicit new responses when an action fails. Repair focuses on a single response at a time; STARS analyzes a set of responses to evaluate the result of using them, making evaluations and repairs before any response is selected and used. Logeswaran et al. (2022) plan sequences of subgoals from multiple LLM responses obtained from beam search (as in STARS) that does re-ranking based on feedback from the environment. SayCan (Ahn et al. 2022) uses an LLM and a trained set of low-level robot skills with short language descriptions for objects. The LLM is prompted multiple times for a high-level task to retrieve one low-level step at a time until a complete plan is found. To obtain knowledge of low-level tasks, SayCan is trained on over 68K teleoperated demonstrations and human-rated simulations. STARS encodes properties for object classes (e.g., whether an object can be "grabbed" by the robot) but requires no pre-training or prior exposure to the domain.

TidyBot (Wu et al. 2023) and TIDEE (Sarch et al. 2022) address robotic problems similar to one of our experimental tasks (tidying a kitchen). They also account for human preferences. TidyBot tries to elicit human preferences by having the LLM summarize a few answers given by a human. TIDEE attempts to learn preferences by using "commonsense priors" learned previously by performing tasks in a "training house." STARS does not depend on pre-training, but does elicit human preferences via NL dialogues.

PROGPROMPT (Singh et al. 2022) produces task plans by prompting an LLM with Python code that specifies the action primitives, objects, example tasks, and task name. The LLM returns a task plan in Python which includes assertions about states of the environment that are checked during execution, and recovery steps if an assertion fails. STARS retrieves NL descriptions of goals, rather than plans, and evaluates goals before they are used.

STARS attempts to verify LLM responses before attempting to achieve the goal indicated by a response. There are many approaches to verification of LLM knowledge, including 1) response sampling (Wang et al. 2023), 2) use of other sources of knowledge such as planning (Valmeekam et al. 2023) or an LLM (Kim, Baldi, and McAleer 2023), and 3) human feedback/annotation (TidyBot). Recursively Criticizes and Improves (RCI; Kim, Baldi, and McAleer 2023) verifies LLM output by prompting the LLM again to identify (potential) issues. Cobbe et al. (2021) train a verifier to rank responses, while self-consistency (Wang et al. 2023) uses voting to select an answer. Diao et al. (2023) combine all three of the above verification strategies by eliciting responses from an LLM, ranking them using an uncertainty metric (a source of knowledge other than the LLM), and then having humans annotate responses for further exploration.

While these efforts address similar challenges (or aspects of them), a unique aspect of STARS is the proactive analysis of many responses retrieved via prompting an LLM through embodied reasoning. The analysis enables the identification of known problems and targeted repairs. STARS also learns goal states for tasks, rather than action sequences to achieve tasks. The STARS agent learns task knowledge in one shot, during performance, without prior training. When confronted with the same or similar tasks in the future, the agent can efficiently execute the task without the use of the LLM (or STARS). Encoding persistent task knowledge contrasts with in-context learning (OpenAI 2023).

## Prior Baseline: Template-based Prompting

The agent employs template-based prompting (TBP) to elicit responses from the LLM. Templates enable the agent to construct prompts using context from the task and environment and introduce prompt examples matched to the agent's capabilities and embodiment. Figure 1 outlines the baseline template-based prompting approach for generating task-goal descriptions (i.e., it replaces the NL-dialogue for "Get goal description" in Figure 4). A prompt template is chosen and instantiated with relevant context, the LLM is queried (potentially soliciting multiple responses using varying temper-

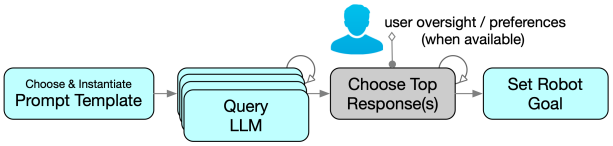


Figure 1: Baseline approach to elicitation of goal descriptions via template-based prompting (TBP).

atures), and response(s) are chosen for execution. In this baseline approach, choices are ranked by the mean log probabilities of tokens in each response. Oversight is used to select an LLM response or to give a goal description when all LLM-generated choices are unacceptable. The agent uses the chosen response to attempt to perform the task and, if successful, learns a policy to execute the task in the future (see Figure 4). Few-shot examples in the prompt bias the LLM toward responses that are viable and relevant, matching the agent's NLP capabilities, desired semantic content (e.g., simple goal statements), and embodiment limitations (Kirk et al. 2022). This baseline approach learns the task in one shot but requires substantial user oversight to overcome errors (Kirk et al. 2023).

## The STARS Approach

STARS extends the TBP baseline with three processes: retrieving a tree of LLM responses via beam search (ST: Search Tree), analyzing and repairing responses (AR: Analysis and Repair), and using the LLM to select a goal response from the candidates (S: Selection). After presenting each of these components of STARS, we describe the oversight strategy of soliciting user feedback.

Figure 2 outlines the process of the STARS approach (blue boxes are re-purposed elements from TBP; green boxes are new components of STARS). With STARS, the agent retrieves goal descriptions from the LLM (the rest of the task-learning process is the same). STARS ensures that the goal descriptions it retrieves from the LLM are viable for the agent. Acquiring goal knowledge is crucial to learning novel tasks, enabling an agent with planning capabilities to perform the new task. Goal learning enables greater flexibility than learning a sequence of actions because goal-state knowledge can transfer to other situations that require different action sequences to achieve the same goal.

### Search Tree (ST)

In prior work with TBP (Figure 1), we increased the temperature parameter iteratively to retrieve multiple responses for the same prompt. This approach resulted in many duplicate responses and more responses that were not viable, deviating from targeted content and form. Similar to others (Logeswaran et al. 2022; Wang et al. 2023), here we enable the agent to use a beam-search strategy to generate a breadth of high-probability responses from a single prompt.

### Analyze and Repair (AR)

While many responses retrieved from the LLM are reasonable, they often fail to meet other requirements: being

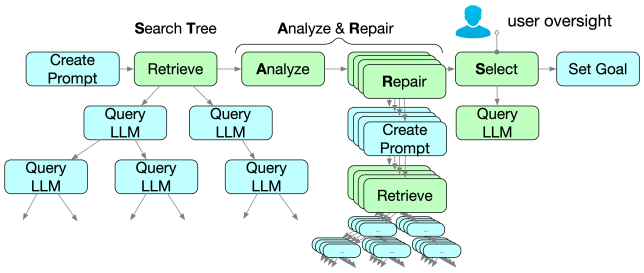


Figure 2: Summary of STARS approach.

matched to the agent's embodiment, language capabilities, and situation. An agent that attempts to use a mismatched response will fail. Analysis and Repair detects and categorizes mismatches, drawing on the cognitive agent's knowledge and capabilities to identify problems, and then attempts to repair responses with identifiable mismatches.

The overall process for Analysis and Repair is illustrated in Figure 3. The agent performs a mental simulation of *what would happen* if it attempted to use a response from the LLM, using the same knowledge of parsing and grounding it uses when performing the task. The analysis evaluates interpretability (orange: whether the agent can parse and interpret the language and terms), grounding (green: whether each referent in the response can be grounded to an object observable in the environment), and affordances (blue: whether the agent can achieve the actions on objects implied by clauses in the goal response). The "AR" process currently addresses these three sources of mismatch:

- **Language**: The agent parses the response with its native NLP capabilities and examines the output. The language processor indicates if a sentence can be interpreted and identifies unknown words.
- **Situation**: To detect grounding issues, the agent evaluates the results of its language comprehension process. When a sentence contains a referring expression to an object, such as a cabinet, the agent's language processing identifies grounding candidates observable by the agent. Failure to ground a referent indicates a mismatch with the current situation.
- **Embodiment and Affordance**: The agent detects embodiment and affordance mismatches using its knowledge of objects (semantic memory) and properties detected from perception (environment). E.g., when it processes a clause in a goal response such as "the dish rack is in the cabinet," it evaluates if the object to be moved ("dish rack") has the property "grabbable."

Repair is coupled to these diagnostic mismatches detected during analysis. For each type of diagnosis, the agent constructs a new prompt using a repair template for that category of mismatch. The agent instantiates the template by appending the non-viable response with an instruction indicating the specific mismatch that occurred, e.g., "No. Cannot see a cabinet." or "No. Rack is not grabbable."[1] ST then uses

[1]A Technical Appendix provides complete examples of prompts for repairs and selection: https://arxiv.org/abs/2306.06770.

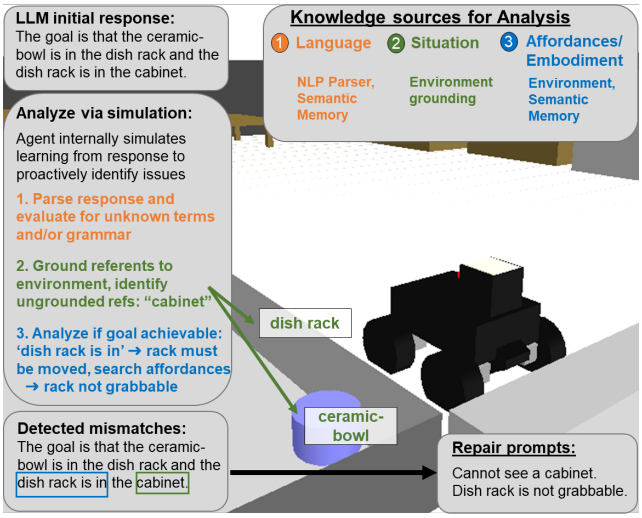


Figure 3: Agent analysis of mismatches via internal simulation

this repair prompt to generate a new tree of responses.

## Selection (S)

ST and AR are designed to generate viable candidate responses. However, the agent must select a single response to use. Rather than using mean log probability (as in TBP; Figure 1) or voting (as in self-consistency Wang et al. 2023), the new Selection strategy employs the LLM for choosing a response. The agent constructs a prompt with the candidates and asks which of a numbered list of candidate responses is the most reasonable goal given the task context. The prompt solicits a single integer response from the LLM, indicating which response is the best.

## User Oversight (O)

The correct goal for some tasks depends on human preferences (e.g., some users prefer storing cereal in the cupboard, others, the pantry). The original ITL agent solicited all task knowledge from a human, which naturally captured this preference knowledge. STARS reduces user interaction while still ensuring capture of preference. Having the human in the loop also ensures correct learning. The agent solicits user feedback by asking if a retrieved goal is correct (yes/no) before using it (below). Selection determines which option to present. If the first response is rejected, Selection is repeated with the rejected option removed. If all responses are rejected, the user must provide the correct goal description.

**Agent**: For a mug in the dish rack is the goal that the mug is in the cupboard and the cupboard is closed?

**User**: Yes.

# Experiment Design

In order to evaluate STARS, we first describe the embodied agent that incorporates STARS, an experimental design, and measures. In the next section, we present results for online

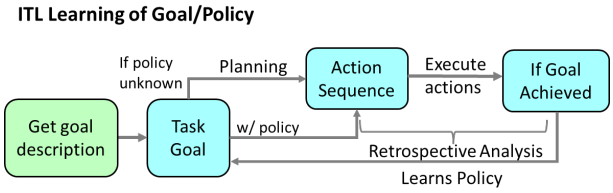


Figure 4: ITL process for learning goals and policy.

learning of three different tasks: tidying the kitchen, storing groceries, and organizing an office. We evaluate how well STARS addresses the above requirements and also examine the relative impact of components of STARS. STARS learns descriptions of goal states, while systems such as SayCan, InnerMonologue, and TidyBot learn action sequences. We do not directly compare performance for these tasks against these systems because of their different learning targets.

**Agent:** We embed STARS in an existing embodied ITL agent, replacing the human interaction that provided natural language descriptions of goals for tasks and subtasks.[2] The original agent learns a variety of diverse tasks (from puzzles to mobile patrol tasks) in many different physical (Fetch robot, mobile robot, and tabletop arm) and simulated (AI2Thor, April simulator) robotic domains (Mohan et al. 2012; Mininger 2021; Kirk and Laird 2019).

Figure 4 depicts the ITL process for learning goals. The ITL agent can also learn new concepts, new actions (when planning knowledge is insufficient), and lower-level skills via instruction (not shown here). We focus on the goal-learning pipeline here because STARS exploits an LLM to learn goal descriptions (replacing the green box) without changing other aspects of the pipeline. The ITL learning process depended on substantial user input to provide interpretable and accurate descriptions of goals. When a policy for achieving a goal is unknown, internal planning finds a sequence of actions that achieves the goal. A side effect of successful planning is that the agent learns long-term policy knowledge in one shot via the agent architecture's procedural learning mechanism. When the task arises in the future, that learned knowledge guides agent decision-making without planning or human interaction.

**Setting**: A simulated office and kitchen with a mobile robot created in the APRIL MAGIC simulator. The robot can move around the room, approach objects, and has a single arm that can grasp and manipulate all objects relevant to the task to be learned. For the "tidy kitchen" task (the largest task), the kitchen is populated with 35 objects that commonly exist in a kitchen (plates, condiments, utensils, etc.). Objects are initially distributed on a table, counter, and in the dish rack. For the "store groceries" task, 15 objects are contained in bags on the kitchen floor that must be stored (into the fridge, cupboard, or pantry). For the "organize office" task, 12 objects are distributed on a desk that must be cleared (into the drawer, bookshelf, trash, recycling bin, or

[2]Code for the ITL agent with STARS, simulator, and data analysis are available at https://github.com/Center-for-Integrated-Cognition/STARS.

| *Condition* | *Description* |
|---|---|
| TBP | **T**emplate-**B**ased **P**rompting (Baseline) |
| TBP+O | TBP with human **O**versight |
| ST | Beam **S**earch **T**ree |
| STS | Beam search with LLM **S**election |
| STAR | Beam search with **A**nalysis (check viability) and **R**epair |
| STARS | Search-tree, A&R, LLM Selection |
| STARS+O | STARS with human oversight. |
| Trial #2 | Task performance on second presentation after learning with STARS+O. |

Table 1: Definition of experimental conditions.

filing cabinet). The three tasks contain 58 unique objects for which the agent needs to learn a goal.

**Simulation**: Although prior work with the ITL agent has used a physical robot, this experiment is done in simulation, which is sufficient for investigating the grounding of concepts and interpreting and learning from the descriptions provided by STARS.

**Learning Task**: For each experiment, the user presents the task (e.g., "tidy kitchen") and primary subtasks (e.g., clearing, storing, and unloading all the objects from the table, counter, and dish rack). For all tasks, task success is measured by the fraction of objects moved to a location consistent with user preferences. Also, when another object is manipulated to achieve a task (e.g., opening a refrigerator door to put away ketchup), it must be in its desired state for the task-success evaluation (e.g., the door must be closed). For the "tidy kitchen" task, four object types have multiple instances that must be treated differently based on their positions (e.g., a mug on the table must be put in the dishwasher or sink, but a mug in the dish rack must be put in the cupboard). Using the approach in Figure 2 (or a STARS variant as below), the agent acquires goal descriptions for each perceived object. It then uses the processing described in Figure 4 to learn the goal and action policy, enabling it to correctly process that object in the future without the LLM, planning, or oversight.

**Experimental conditions**: Experimental conditions are enumerated in Table 1. The TBP conditions are baselines for assessing the impact of the components of STARS. For all conditions, the LLM used is GPT-3 (for TBP, Search Tree, and Repair) and GPT-4 (for Selection).[3] In all conditions, a user provides the initial task. In the Oversight conditions, the user reviews up to 5 responses. In non-oversight conditions, the choice of the goal is based on the highest mean log probability of candidates (ST and STAR) or the Selection strategy (STS and STARS).

**Measures:** We assess conditions in three dimensions: performance, response quality, and cost. For performance, task completion rate (number of goal assertions achieved / total number of goal assertions) is the primary measure. For response quality, we evaluate how well responses align with requirements for situational relevance and viability, as well as reasonableness. User effort is the largest factor impacting cost, but cannot be measured directly. To estimate effort, we use the number of interactions and words as well as the percentage of accepted goals. LLM costs are evaluated via tokens presented (prompts) and generated (responses).

[3]GPT-4 does not currently expose logprobs, making it inapt for beam search. Selection does not use beam search and GPT-4 demonstrated better, more consistent results.

| *Condition* | Comp. (%) | Goals retvd | Total tokens | # instrct | # words |
|---|---|---|---|---|---|
| **Tidy kitchen** | | | | | |
| TBP | 52.5 | 93 | 41407 | 14 | 76 |
| TBP+O | 100.0 | 89 | 42469 | 92 | 403 |
| ST | 50.0 | 243 | 56874 | 14 | 76 |
| STS | 40.0 | 247 | 66458 | 14 | 76 |
| STAR | 77.5 | 353 | 126086 | 14 | 76 |
| STARS | 77.5 | 368 | 139871 | 14 | 76 |
| STARS+O | 100.0 | 361 | 138096 | 65 | 127 |
| Trial #2 | 100.0 | 0 | 0 | 1 | 2 |
| **Store groceries** | | | | | |
| TBP | 66.7 | 39 | 17078 | 6 | 28 |
| TBP+O | 100.0 | 37 | 18689 | 29 | 92 |
| ST | 66.7 | 96 | 21518 | 6 | 28 |
| STS | 66.7 | 99 | 25690 | 6 | 28 |
| STAR | 77.8 | 170 | 57709 | 6 | 28 |
| STARS | 94.4 | 171 | 61808 | 6 | 28 |
| STARS+O | 100.0 | 177 | 64501 | 22 | 44 |
| Trial #2 | 100.0 | 0 | 0 | 1 | 2 |
| **Organize office** | | | | | |
| TBP | 35.7 | 34 | 12992 | 6 | 28 |
| TBP+O | 100.0 | 35 | 11662 | 41 | 184 |
| ST | 21.4 | 95 | 21082 | 6 | 28 |
| STS | 21.4 | 97 | 24717 | 6 | 28 |
| STAR | 64.3 | 204 | 75509 | 6 | 28 |
| STARS | 92.9 | 201 | 76056 | 6 | 28 |
| STARS+O | 100.0 | 206 | 77722 | 22 | 60 |
| Trial #2 | 100.0 | 0 | 0 | 1 | 2 |

Table 2: Summary of outcomes by condition for three tasks.

## Experimental Results

The discussion of experimental results is organized around the three measures introduced above. Table 2 summarizes performance (task completion) and costs (tokens; oversight) for each condition for the three tasks. The Trial #2 condition shows task performance after successful learning from STARS+O when given a second direction to perform the task; all tasks are completed successfully without further interaction beyond receiving the task (e.g., "tidy kitchen").[4]

For each task we ran the STARS condition 10 times. Table 3 shows the mean values and standard deviation for task completion for each task. Due to the lack of variation between runs (attributable to the LLM and STARS) as well as experimental costs (GPT budget and the time to conduct each condition for all task experiments) we report results

[4]A video demonstration of STARS with a few objects is available at http://tinyurl.com/STARS-AAAI24.

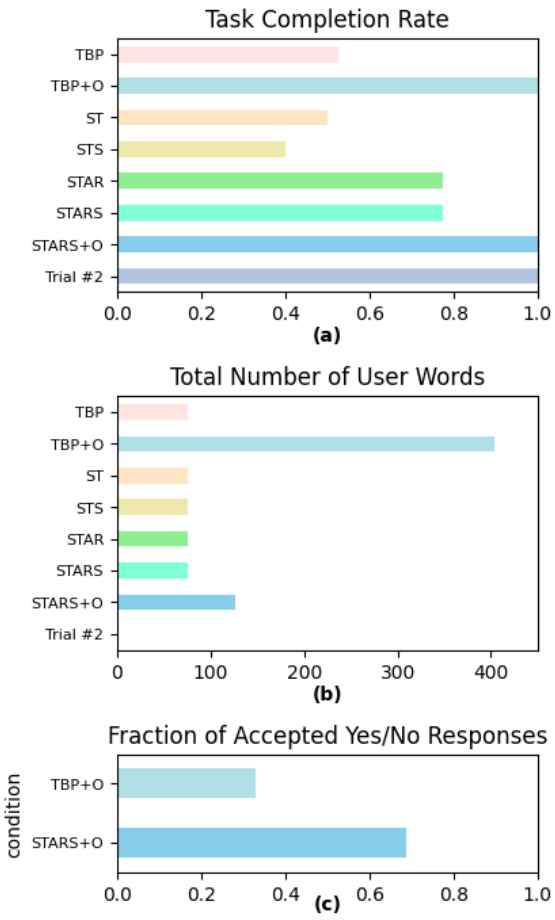


Figure 5: Performance and user cost measures for experimental conditions for the "tidy kitchen" task.

from one run for each condition (Table 2). The overall variance for STARS is small and has a marginal effect on key outcomes (See Section D in the Technical Appendix for further exploration of variability in outcomes).

**Performance:** Table 2 shows the task completion rates for all experimental conditions for the the three tasks. Figure 5(a) graphically compares task completion rates for the largest task: "tidy kitchen." The baseline condition, TBP, achieves the experiment-defined targets (e.g., "mug in the dishwasher") only 52.5% (tidy kitchen), 66.7% (store groceries), and 35.7% (organize office) of the time. Adding Oversight to the baseline condition (TBP+O) results in 100% task completion but vastly increases the number of required words (5b). Because many responses from the LLM are not viable and situationally relevant, the user must provide goal descriptions, resulting in many more words of instruction. Without oversight, STARS delivers a large gain in task completion, increasing to 77.5% (tidy), 94.4% (store), and 92.9% (organize). Analysis and Repair (AR) prevents the agent from using unviable responses and increases the number of viable responses via repair. Search Tree (ST) alone results in no improvement but is a prerequisite for AR.

| Task: | Kitchen | Groceries | Office |
|---|---|---|---|
| Mean | 77.5 | 93.89 | 92.14 |
| Std Dev. | 2.04 | 1.76 | 2.26 |

Table 3: Variation in task completion rate for three tasks (STARS condition only).

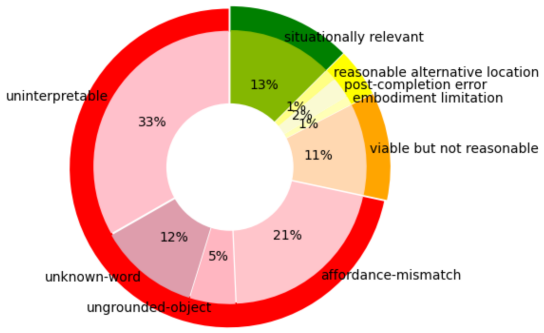


Figure 6: Categorization of responses retrieved from the LLM (STARS condition from "tidy kitchen" task).

The task completion for "tidy kitchen" (77.5%) is significantly lower than for the other tasks using STARS. For the "store groceries" and "organize office" tasks, the addition of Selection (S) improved task completion, but did not for "tidy kitchen." From detailed analysis, we determined that the agent lacks context specific to the tidy task. For instance, the agent (in this instantiation) lacks the ability to discriminate between a "clean" and "dirty" mug. In the "tidy kitchen" experiment, dishware on the table is assumed to be dirty (in terms of defining the target outcomes in the design), but the agent lacks this context. When such context is provided to the LLM (a variation we label STARS*),[5] Selection achieves 92.5% task completion for "tidy kitchen" (without user oversight), comparable to the STARS task completion results for the other two tasks. In the future, we will enable the user to provide this context directly.

With oversight, STARS task completion rises to 100% for all tasks with much-reduced user input compared to TBP. This gain comes from shifting user input from providing goal descriptions (often needed in TBP) to confirming LLM-generated goal descriptions with yes/no responses (STARS+O). In addition, as highlighted in Figure 5(c), the greater precision of STARS in generating acceptable goal descriptions results in the user accepting a larger fraction of the goals in the oversight condition. The fraction of accepted goals increases from 33% to 69% (tidy kitchen), 62% to 94% (store groceries), and 18% to 73% (organize office).

**Quality of Responses:** Figure 6 shows the percentage of different classifications of the responses retrieved from the LLM for STARS for tidying the kitchen.[6] Responses are

[5]Context provided to GPT-4 as a System prompt: "Assume that dishware on the table or counter are dirty. Assume that bottles and cans are empty. Non-perishable food belongs in the pantry."

[6]Chart is representative of all conditions except TBP and Oversight; see appendix for each condition for all tasks.


Figure 7: Fraction of responses used by the robot that are reasonable/sit. relevant for the "tidy kitchen" task.

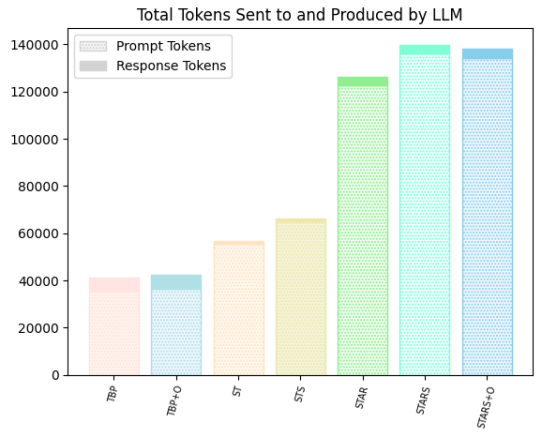


Figure 8: LLM tokens sent (hatched) and received (solid).

classified as unviable (red), viable but not reasonable (orange), reasonable (yellow), or situationally relevant (green). Further categorization identifies the type of mismatch for unviable responses (unknown word, ungrounded object, uninterpretable, affordance mismatch) and reasonable ones (reasonable alternative location, post-completion error, embodiment limitation). "Post-completion error" indicates a reasonable failure to close a door in situations where an object might not have a door. "Embodiment limitation" captures when the robot places an object in a location that would otherwise be reasonable if its sensing were not limited.

Over 70% of responses are not viable, leading to failure if the robot executed them; only 13% are situationally relevant, meeting all four requirements. For storing groceries 58% were not viable and 14% were situationally relevant, and for organizing the office 85% were not viable and only 5% were situationally relevant. Thus, analysis of responses appears essential for reliable use of an LLM by an embodied agent to prevent the use of unviable goal descriptions. In the baseline (TBP) for tidying the kitchen, the agent retrieves at least one situationally relevant responses for only 15 of the 35 objects, while STARS results in 100% of the objects having at least one situationally relevant response.[7]

Figure 7 shows the quality of response by evaluating how frequently the robot receives a viable and (at least) reasonable response (situationally relevant for some user but not necessarily this one). For "tidy kitchen," STARS (and STAR) results in 100% of the used responses being at least reasonable. This indicates that STARS' 77.5% task completion is close to the best it can achieve without oversight (or additional context). Human input is necessary to differentiate situationally relevant goals from reasonable ones.

**Cost:** Table 2 shows that oversight, in the form of instructions and words, is reduced by STARS (from 403 words to 127 for tidy kitchen, 92 to 44 words for store groceries, and 184 to 60 words for organize office). While the magnitude of the reduction is modest, the user now confirms a goal with a single word in comparison to supplying a complete goal description. STARS+O also increases the precision of presented responses (Figure 5c); 69% (kitchen), 94% (grocery), and 73% (office) of responses are accepted. Figure 8 summarizes LLM tokens used for prompting and generation for "tidy kitchen." For this task and the others, token cost increases substantially in Search Tree (ST) and Analysis and Repair (AR), because of the recursive beam search.

[7]See appendix for graphical analysis of all conditions and tasks.

## Conclusion

Using LLMs as the sole source of knowledge for an embodied agent is challenging due to the specific requirements that arise in operationalizing that knowledge. STARS enables an agent to more effectively exploit an LLM, ensuring that the responses are viable (interpretable and grounded in the situation and agent capabilities). STARS shifts the role of the LLM from being the sole knowledge source to one source within a more comprehensive task-learning process (Kirk et al. 2023). It both addresses LLM limitations and takes advantage of the knowledge, reasoning, and online learning capabilities of cognitive agents.

While STARS provides significant improvements, further exploration and development are warranted. In particular, Selection does not provide a consistent improvement over the mean log prob choice strategy for "tidy kitchen" due to a lack of context. For future work, we will explore improvements to Selection, especially via the use of additional context that the agent can obtain from the user and (for some contexts) the LLM as briefly outlined here (STARS*).

Finally, STARS also helps highlight the necessity of human oversight in the use of LLMs for agent task learning. Minimally, oversight ensures that an agent that uses an LLM is not led astray by the LLM, which can produce unsafe, biased, and unethical responses (Weidinger et al. 2021). Further, a human user will often be the only source of *certain* knowledge of what goals and outcomes are appropriate for the task (Requirement 4). STARS, by ensuring that all candidates presented to the user are viable, simplifies and streamlines human oversight, reserving it for knowledge only a human can provide. This streamlining not only reduces the tedium of interaction (as suggested by the experimental results), it also potentially allows users to better focus on alignment with their needs, goals, and values.

## Ethical Statement

This work uses large language models which can present ethical and social risks (Weidinger et al. 2021) such as discrimination, exclusion, and toxicity or malicious uses. We consider each of these risks.

Because large language models are generative, depending on their corpus and training, they can produce language that reflects cultural biases, offensive stereotypes, derogatory usages, etc. In this work, where LLM queries are focused solely on producing goal descriptions for a particular task environment, we have not seen *any* responses from GPT-3 that include such language.

In terms of exclusion, the specific tasks we have chosen do reflect (and mirror) cultural specificity to US/Western settings, in that the items in the kitchen and office (and the labels used to describe them) are both specific to the English language and typical of the objects one would find in a kitchen in a home or in an office environment. One of the long-term potential outcomes of this work (and ITL more generally) is that the agent is taught by human users in a particular setting, allowing the human to customize agent behavior to their specific setting (including its cultural context). Investigating whether this potential can be realized in the subject of future work.

Malicious use is also a potential risk, in that this research aims to enable human users to instruct agents to do their bidding. Users could theoretically instruct agents to cause direct harm to others, violate laws, etc. At this point in our research, this risk is minimal because implementations are confined to controlled, laboratory experiments. We are actively investigating in other work how an ITL agent can be both instructed while also following and conforming to both codified rules (like laws) and social norms to further mitigate the potential for malicious use.

## Acknowledgements

This work was supported by the Office of Naval Research, contract N00014-21-1-2369. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Department of Defense or Office of Naval Research. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

## References

Ahn, M.; Brohan, A.; Brown, N.; Chebotar, Y.; Cortes, O.; David, B.; Finn, C.; Gopalakrishnan, K.; Hausman, K.; Herzog, A.; et al. 2022. Do As I Can, Not As I Say: Grounding Language in Robotic Affordances. In *6th Annual Conference on Robot Learning*.

Cobbe, K.; Kosaraju, V.; Bavarian, M.; Chen, M.; Jun, H.; Kaiser, L.; Plappert, M.; Tworek, J.; Hilton, J.; Nakano, R.; Hesse, C.; and Schulman, J. 2021. Training Verifiers to Solve Math Word Problems. ArXiv:2110.14168 [cs], arXiv:2110.14168.

Diao, S.; Wang, P.; Lin, Y.; and Zhang, T. 2023. Active Prompting with Chain-of-Thought for Large Language Models. ArXiv:2302.12246 [cs].

Gluck, K.; and Laird, J., eds. 2019. *Interactive Task Learning: Agents, Robots, and Humans Acquiring New Tasks through Natural Interactions*, volume 26 of *Strüngmann Forum Reports*. Cambridge, MA: MIT Press.

Huang, W.; Xia, F.; Xiao, T.; Chan, H.; Liang, J.; Florence, P.; Zeng, A.; Tompson, J.; Mordatch, I.; Chebotar, Y.; et al. 2022. Inner Monologue: Embodied Reasoning through Planning with Language Models. In *6th Annual Conference on Robot Learning*.

Kim, G.; Baldi, P.; and McAleer, S. 2023. Language Models can Solve Computer Tasks. ArXiv:2303.17491 [cs].

Kirk, J. R.; and Laird, J. E. 2016. Learning General and Efficient Representations of Novel Games Through Interactive Instruction. In *Proceedings of the Advances in Cognitive Systems Conference*. ISBN 0021-9967.

Kirk, J. R.; and Laird, J. E. 2019. Learning Hierarchical Symbolic Representations to Support Interactive Task Learning and Knowledge Transfer. In *Proceedings of IJCAI 2019*, 6095–6102. International Joint Conferences on Artificial Intelligence.

Kirk, J. R.; Wray, R. E.; Lindes, P.; and Laird, J. E. 2022. Improving Language Model Prompting in Support of Semi-autonomous Task Learning. In *Proceedings of the Advances in Cognitive Systems (ACS) Conference*.

Kirk, J. R.; Wray, R. E.; Lindes, P.; and Laird, J. E. 2023. Integrating Diverse Knowledge Sources for Online One-shot Learning of Novel Tasks. ArXiv:2208.09554 [cs], arXiv:2208.09554.

Laird, J. E.; Gluck, K.; Anderson, J. R.; Forbus, K.; Jenkins, O.; Lebiere, C.; Salvucci, D.; Scheutz, M.; Thomaz, A.; Trafton, G.; Wray, R. E.; Mohan, S.; and Kirk, J. R. 2017. Interactive Task Learning. *IEEE Int. Sys.*, 32(4): 6–21.

Logeswaran, L.; Fu, Y.; Lee, M.; and Lee, H. 2022. Few-shot Subgoal Planning with Language Models. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL)*, 5493–5506. Association for Computational Linguistics.

Mininger, A. 2021. *Expanding Task Diversity in Explanation-Based Interactive Task Learning*. Ph.D. Thesis, University of Michigan, Ann Arbor.

Mohan, S.; and Laird, J. E. 2014. Learning Goal-Oriented Hierarchical Tasks from Situated Interactive Instruction. In *Proceedings of the 28$^{th}$ AAAI Conference on Artificial Intelligence*, volume 2, 113–130. AAAI Press.

Mohan, S.; Mininger, A.; Kirk, J.; and Laird, J. E. 2012. Acquiring Grounded Representation of Words with Situated Interactive Instruction. *Advances in Cognitive Systems*, 2: 113–130.

Olmo, A.; Sreedharan, S.; and Kambhampati, S. 2021. GPT3-to-plan: Extracting plans from text using GPT-3. In *Proceedings of ICAPS FinPlan*. ArXiv: 2106.07131 [cs].

OpenAI. 2023. GPT-4 Technical Report. arXiv:2303.08774.

Park, J. S.; O'Brien, J.; Cai, C. J.; Morris, M. R.; Liang, P.; and Bernstein, M. S. 2023. Generative Agents: Interactive Simulacra of Human Behavior. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*, 1–22.

Reynolds, L.; and McDonell, K. 2021. Prompt Programming for Large Language Models: Beyond the Few-Shot Paradigm. In *Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems*, CHI EA '21. New York, NY, USA: ACM. ISBN 9781450380959.

Richards, T. B. 2023. Auto-GPT: An Autonomous GPT-4 Experiment.

Sarch, G.; Fang, Z.; Harley, A. W.; Schydlo, P.; Tarr, M. J.; Gupta, S.; and Fragkiadaki, K. 2022. TIDEE: Tidying Up Novel Rooms using Visuo-Semantic Commonsense Priors. In *Computer Vision–ECCV 2022*, 480–496. Springer.

Singh, I.; Blukis, V.; Mousavian, A.; Goyal, A.; Xu, D.; Tremblay, J.; Fox, D.; Thomason, J.; and Garg, A. 2022. ProgPrompt: Generating Situated Robot Task Plans using Large Language Models. arXiv:2209.11302.

Sumers, T. R.; Yao, S.; Narasimhan, K.; and Griffiths, T. L. 2023. Cognitive Architectures for Language Agents. ArXiv:2309.02427 [cs].

Valmeekam, K.; Sreedharan, S.; Marquez, M.; Olmo, A.; and Kambhampati, S. 2023. On the Planning Abilities of Large Language Models (A Critical Investigation with a Proposed Benchmark). arXiv:2302.06706.

Wang, X.; Wei, J.; Schuurmans, D.; Le, Q.; Chi, E.; Narang, S.; Chowdhery, A.; and Zhou, D. 2023. Self-Consistency Improves Chain of Thought Reasoning in Language Models. In *The Eleventh International Conference on Learning Representations (ICLR 2023)*.

Weidinger, L.; Mellor, J.; Rauh, M.; Griffin, C.; Uesato, J.; Huang, P.-S.; Cheng, M.; Glaese, M.; Balle, B.; Kasirzadeh, A.; Kenton, Z.; Brown, S.; Hawkins, W.; Stepleton, T.; Biles, C.; Birhane, A.; Haas, J.; Rimell, L.; Hendricks, L. A.; Isaac, W.; Legassick, S.; Irving, G.; and Gabriel, I. 2021. Ethical and social risks of harm from Language Models. ArXiv:2112.04359 [cs].

Wu, J.; Antonova, R.; Kan, A.; Lepert, M.; Zeng, A.; Song, S.; Bohg, J.; Rusinkiewicz, S.; and Funkhouser, T. 2023. TidyBot: Personalized Robot Assistance with Large Language Models. ArXiv:2305.05658 [cs], arXiv:2305.05658.