MENSA: Leveraging Mental Simulation for In-Context Policy Improvement in LLM Agents

Chung-Che Chang
Department of Computer Science
National Taiwan University
Taipei, Taiwan
ccchang.rg@gmail.com

Erick Chandra
Department of Computer Science
National Taiwan University
Taipei, Taiwan
erickchandra.1@gmail.com

Jane Yung-jen Hsu
Department of Artificial Intelligence
Chang Gung University & National Taiwan University
Taoyuan, Taiwan
yjhsu@csie.ntu.edu.tw

Yen-Ling Kuo
Department of Computer Science
University of Virginia
Charlottesville, VA, United States
ylkuo@virginia.edu

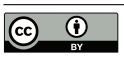
ABSTRACT

Large Language Model (LLM) powered agents have shown promise in sequential decision-making tasks in interactive environments. However, prior agent frameworks usually rely on advanced LLM capabilities such as planning or instruction following to carry out tasks successfully. Effectively improving the performance of an LLM agent without assuming these capabilities remains challenging. To address this issue, we propose **MEN**tal **S**imulation **A**gent (**MENSA**), a novel model-based approach that enhances LLM agents without fine-tuning. MENSA leverages the fundamental ability of any LLMs, text completion, to generate forecasts of action-state pairs (i.e., transitions) for future time steps. These forecasts are used to construct a set of relevant past experiences, which are provided to the LLM agent in context to improve its decision-making behavior. We evaluate MENSA in two challenging interactive environments, ScienceWorld and NetHack, and show that MENSA improves performance across various sizes of LLMs. Using large models (e.g., GPT-4o-mini), MENSA outperforms previous state-of-the-art methods by +15.8 points in ScienceWorld and by +40.0 points in NetHack. Even with smaller models like Phi-3-mini, MENSA achieves a gain of +11.9 points in ScienceWorld. Our results further suggest that MENSA is less affected by an LLM's limitations in instruction-following and planning compared to baselines. Project page and code are available at: https://roger0426.github.io/MENSA.

KEYWORDS

LLM Agents; In-context Learning; Mental Simulation; Sequential Decision Making; Model-based Policy Improvement

ACM Reference Format:
Chung-Che Chang, Erick Chandra, Jane Yung-jen Hsu, and Yen-Ling Kuo. 2026. MENSA: Leveraging Mental Simulation for In-Context Policy Improvement in LLM Agents. In *Proc. of the 25th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2026), Paphos, Cyprus, May 25 – 29, 2026*, IFAAMAS, 10 pages. https://doi.org/10.65109/BBRH3447



Proc. of the 25th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2026), C. Amato, L. Dennis, V. Mascardi, J. Thangarajah (eds.), May 25 – 29, 2026, Paphos, Cyprus. © 2026 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org). https://doi.org/10.65109/BBRH3447

1 INTRODUCTION

To successfully carry out tasks in complex environments, autonomous agents must engage in a sequence of decisions to select optimal actions. This usually requires an agent to learn to decompose tasks [11], identify relevant prior experience, or generate, evaluate, and refine plans [30]. Progress in large language models (LLMs) has demonstrated the potential to leverage in-context learning [2, 37, 38] to design several components in autonomous agents [35]. These LLM-based agents have been created in interactive domains such as programming [40], games [34], and robotics [3, 14].

Recent works in LLM agents have considered several aspects to optimize an agent's policy, e.g., the reflection on past executions [30] or the next subgoals [27]. As shown in Figure 1 (a,c), these optimization approaches are *model-free* — they rely on the feedback from direct interaction with the environment to update the in-context prompt to improve action generation. However, these methods usually require numerous interactions with the environment to learn a reasonable policy. Their performances are largely impacted by LLMs' planning and instruction following capabilities as these abilities are needed to derive information from the environment feedback.

Human decision-making, on the other hand, does not rely only on direct feedback. Before executing the actions in an environment, humans also imagine the necessary steps, anticipate their outcomes, and determine how to combine them for successful task execution [32]. This *mental simulation* process enables us to hypothesize the preconditions and effects of actions so we can select the actions that help satisfy the preconditions of subgoals. In policy learning, this is a *model-based* approach – as shown in Figure 1 (b,c), the agent uses a model to estimate the state-action transitions of the next few steps and uses this information to improve the policy in context. The ability to simulate future steps enables the agent to improve its policy more effectively from a few interactions.

In this paper, we present the Mental Simulation Agent (MENSA), an implementation of model-based LLM agents that leverage the text completion ability [25, 26] of LLM as a model to simulate the state-action transitions. At each time step, in addition to selecting the next action, MENSA prompts LLMs to generate action-state pairs (i.e., a forecast) for future time steps. The simulated actions are used to retrieve the relevant experience for the task. For example,

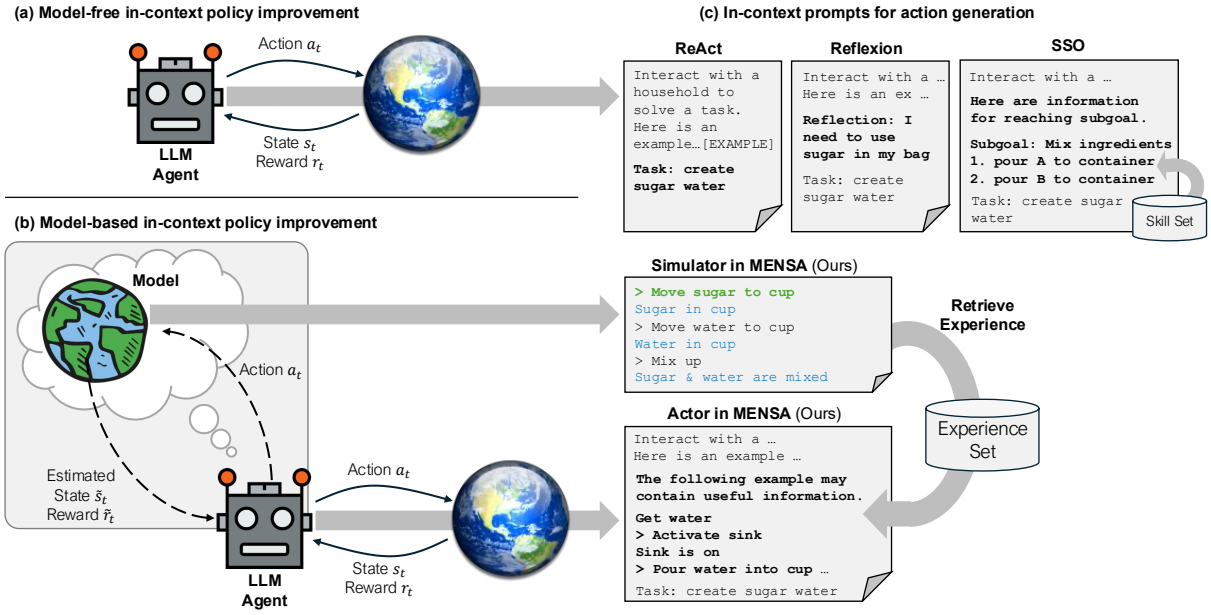


Figure 1: Comparison between (a) model-free and (b) model-based in-context policy improvement frameworks. (c) We show the different in-context prompt examples for the LLM agent in each framework. ReAct, Reflexion, and SSO represent the model-free method, which relies on direct interactions with the environment through task instruction, self-reflection, and skills based on the environment feedback (shown in bold texts in top prompts). Our proposed approach, MENSA, represents the model-based approach, which uses an LLM as a simulator to estimate the outcome of an action a_t (in green text). This estimated outcome includes the next few states (in blue texts) and action (prefixed with ">") transitions, and is used to retrieve the relevant experience for the actor prompt to improve action generation.

in Figure 1 (c), the forecast in MENSA suggests that "move water to cup" is needed before mixing sugar and water, so it can select the action "Activate sink" that helps satisfy the precondition of the task. MENSA employs a multi-episodic refinement framework to improve its forecasts. In the beginning, the agent uses the knowledge in pretrained LLMs to generate forecasts. After each episode of task execution, the LLM will incorporate the environmental feedback to update an experience set, which is used to refine the agent's forecast to be more precise and task-specific.

We evaluate the effectiveness of MENSA on the interactive text-based environment ScienceWorld [36] and the grid-based game environment NetHack [15]. We show that MENSA outperforms the top-performing SSO [27] agent by +15.8 points on ScienceWorld and by +40.0 points on NetHack. Moreover, for agents that use weaker LLMs, MENSA can improve the performance by a large margin (e.g., +11.9 points for Phi-3-mini [1] in ScienceWorld), demonstrating that mental simulation can consistently improve performance across LLMs of different capacities.

In summary, this paper makes the following contributions:

- We propose MENSA, a novel model-based LLM agent that leverages the text completion ability in LLMs to simulate transitions for policy improvements.
- We implement MENSA in the ScienceWorld and NetHack benchmark to perform complex decision making tasks, e.g., science experiments.
- Our results show that MENSA significantly improves the performance of LLM agents across different sizes, reaching state-of-the-art performance in ScienceWorld and Nethack, while exhibiting lower sensitivity to LLM's planning and instruction following abilities.

2 RELATED WORK

Sequential Decision Making with LLMs. LLMs have been increasingly employed in sequential decision-making tasks. Prior studies [18, 20] demonstrated that fine-tuned LLMs can plan or decompose tasks effectively, yet model updates during interaction are often infeasible for large or closed-source systems. Recent work therefore explores policy refinement without gradient updates [12, 14, 30, 39]. Methods such as CLIN [24], SSO [27], and ExpeL [41] share the principle of learning across episodes, they distill useful knowledge from interaction histories and reuse it to guide future decisions. While CLIN organizes contextual knowledge into symbolic transition insights that describe action dependencies, and ExpeL extracts high-level insights and stores successful trajectories in a static recall pool, only SSO constructs, refines, and continually retrieves transferable skills during interaction, representing a more dynamic and adaptive design. MENSA extends this line of work by introducing model-based mental simulation to guide experience retrieval. Rather than recalling past skills by querying the agent as in SSO,

it predicts possible future states and retrieves experiences that are most relevant to these predicted situations.

LLMs as World Models. A world model can simulate the outcome of an action and the transition of the future timesteps after applying an action to the current state. Whether LLMs can be considered faithful world models remains a topic of ongoing debate [5]. However, some evidence suggests that the process of predicting the next token or word in an LLM is akin to learning the world [13]. Recent studies demonstrate the effectiveness of LLMs as zero-shot learners for time series prediction by treating numerical sequences as text [7, 21, 31]. Frameworks like RAP [9] also leverage LLMs as both world models and reasoning agents, employing Monte Carlo Tree Search to produce reasoning traces. While prior works often interpret LLMs as explicit world models that approximate environment dynamics, MENSA adopts a weaker perspective, treating LLM-generated forecasts as sources of task-relevant signals rather than faithful simulations of the environment.

Capability Evaluation in LLMs. LLMs have been shown to have a diverse set of capabilities that enable them to tackle complex tasks across a wide range of domains. Recent benchmarks have evaluated LLMs' capabilities in instruction-following [28, 42] and tool utilization [4, 10]. In the context of sequential decision-making, instruction-following and planning are particularly critical. Instruction-following ensures that the model accurately interprets and executes commands. Planning allows the model to anticipate future steps, evaluate potential outcomes, and make decisions that optimize the process. However, not all LLMs' capabilities are equal. In this paper, we use T-Eval [4] to evaluate an LLM's capabilities in instruction-following and planning to show how MENSA improves LLMs with different capabilities.

3 MODEL-BASED IN-CONTEXT POLICY IMPROVEMENT

Sequential decision-making, unlike other natural language processing (NLP) tasks, requires an agent to interact with an environment by making a series of actions over a time horizon H to achieve a specified goal. This process is often framed as a Markov Decision Process (MDP), which is defined by a 4-tuple $(\mathcal{S}, \mathcal{A}, T, R)$ where: $\mathcal{S}$ is the set of states, with $s_t \in \mathcal{S}$ representing the state at time t; $\mathcal{A}$ is the set of actions with $a_t \in \mathcal{A}$ representing the action taken by the agent; $T : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ is the transition function, defined as $T(s_t, a_t, s_{t+1}) = P(s_{t+1}|s_t, a_t)$. $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function, where $R(s_t, a_t)$ denotes the immediate reward received after taking action a_t in state s_t. The objective of the agent is to learn an optimal policy $\pi : \mathcal{S} \to \mathcal{A}$ that maximizes the expected cumulative reward, also known as the return G_t: $G_t = \sum_{k=0}^{\infty} \gamma^k R(s_{t+k}, a_{t+k})$ where $\gamma \in [0, 1)$ is the discount factor that weights the importance of future rewards. Formally, the agent aims to solve for: $\pi^* = \arg\max_\pi \mathbb{E}_\pi[G_t]$

In LLMs, considering the efficiency and effectiveness of fine-tuning an LLM, we focus on gradient-free in-context learning, which assumes that the policy π is parameterized by text inputs θ that are provided to the agent with the state s_t.

Model-free Policy Improvement. Existing LLM agents primarily employ model-free methods to improve their policies. After an agent takes actions in the environment, these methods update the in-context prompts to improve their policies based on the collected trajectories $\tau = (s_0, a_0, r_0, s_1, a_1, r_1, \cdots, s_H, a_H, r_H)$. For example, Reflexion [30] leverages self-reflection on agent performance to derive language-based insights for prompt optimization; SSO [27] collects valuable trajectory segments and converts them into skills to guide action generation. However, these approaches often assume LLMs have good planning and instruction following capabilities. For instance, Reflexion relies on LLMs to follow instructions to engage in self-reflection, whereas SSO requires LLMs to plan for the next subgoal. It is very challenging for weaker LLMs (usually smaller parameter sizes) to effectively follow those methods.

Model-based Policy Improvement. Model-based frameworks, on the other hand, do not just update prompts from collected trajectories. It has a model that simulates the outcome of an action $(\tilde{r}_t, \tilde{s}_{t+1})$ or the transition of the following timesteps $\tilde{\tau}_{t+1:H}$ without directly interacting with the real environment. The agent can update the in-context prompts to improve their policies based on the simulated outcomes and transitions in addition to the collected trajectories. While it is possible to create prompts for simulation, weaker LLMs usually do not follow instructions well. To address this limitation, we leverage the core capability of LLMs, *text completion*, to implement mental simulation. This is the most fundamental capability for LLMs as they are pretrained with next token prediction. This allows us to demonstrate MENSA across different sizes of LLMs.

4 MENTAL SIMULATION AGENT

Figure 2 shows the architecture of MENSA, which is composed of three key components: Actor, Executor, and Experience Learner. The Actor carries out mental simulations for a given task, allowing the agent to plan actions and anticipate outcomes, and use the simulated results (called *Forecast*) to improve its policy. The Executor turns the LLM-generated actions into admissible actions and executes them in the environment. The Experience Learner distills the executed trajectories into experiences at the end of each interaction episode.

4.1 Agent Architecture Overview

Actor. The actor constructs a structured prompt for in-context learning. The prompt includes a one-shot example, along with a set of selected relevant experience $\hat{E}$, a task description, and the current trajectory $\tau_{0:t-1} = (s_0, a_0, s_1, a_1, ..., s_{t-1}, a_{t-1})$. We adopt the prompt structure used in ReAct [39], where an action is prefixed with a angle bracket symbol '>', and the subsequent line without the symbol denotes the feedback from the environment. This format not only enhances the clarity of the input for weaker LLMs but also is fully compatible with stronger LLMs, ensuring consistent and effective processing across different model sizes. The criteria and methodology for retrieving relevant experiences as the in-context example are discussed in detail in Section 4.3. The LLM response is then parsed into a raw action and forecast, where the action is executed in the environment and the forecast is used to update the agent's policy.

Executor. Some environments, e.g., ScienceWorld used in our experiments, have strict requirements for actions. Similar to prior

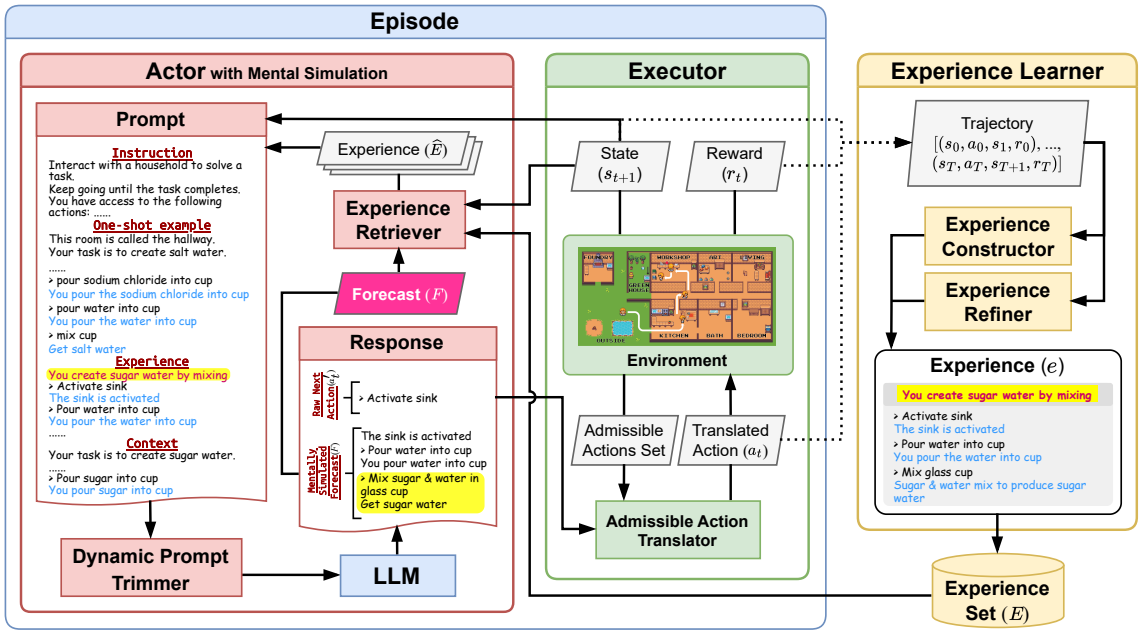


Figure 2: Overview of the MENSA Architecture

works, we design the executor to translate the raw actions into admissible ones to ensure executability. We use SentenceBERT [29] to map raw actions to the most semantically similar admissible actions. We employ a similarity threshold to prevent misalignments. If the similarity score is low, the raw action is executed as is to preserve the agent's original intent.

Experience Learner. After each interaction episode with the environment, the experience learner constructs and refines an experience set to enable effective reuse of relevant past experiences for policy improvement. Each experience consists of a subgoal and the corresponding subtrajectory that leads to the fulfillment of the subgoal, yielding a structured representation of past interactions. We adapt the skill construction and refinement mechanism from SSO [27] to build the experience set. In the construction phase, we first extract potential subtrajectories and then use an LLM to summarize them into reusable experiences. To extract potential subtrajectories, we first score the subtrajectories by the following heuristic function: $exp\_score = w_1 \cdot \text{similarity} + w_2 \cdot \text{reward} + w_3 \cdot \text{length}$ where the weights are set to 1, 0.1, and 0.01, to balance the importance of trajectory similarity, performance, and length, following values reported in previous studies, SSO. The scored subtrajectories are subsequently sampled via beam search. After this, the refinement phase filters out less impactful experiences by tracking the cumulative discounted reward associated with each experience, computed based on past trajectories. Only experiences that demonstrate meaningful contributions to policy improvement are retained for future use.

4.2 Actor with Mental Simulation

LLMs, trained on large-scale textual data, can capture patterns and dependencies within sequences of events. We leverage their text completion capability to generate plausible intermediate steps toward the goal. Unlike methods like ReAct, which focus on generating the next action, we add "Keep going until the task completes." to the in-context prompt to enable LLMs to keep generating subsequent actions and their outcomes. This simple prompt change adds minimal cognitive load to LLMs. No matter what size the LLM is, it can output the continuing trajectory. The output of the LLM is parsed into two parts: (1) the next action, which is to be executed in the environment, and (2) the forecast, which is a sequence of simulated actions and the observation from the environment. In a fully observable setting, we can derive the simulated state by directly incorporating the observation change. This process is equivalent to sampling the future trajectories with horizon h from an LLM:

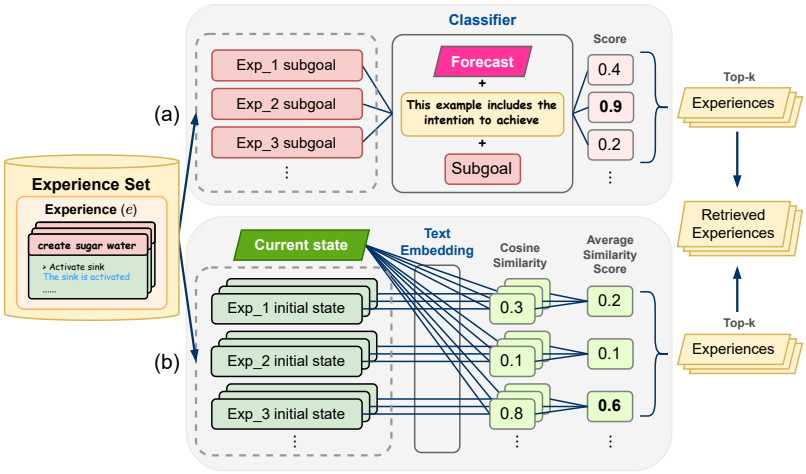
$$a_{t+1}, s_{t+1}, \cdots, a_{t+h}, s_{t+h} \sim P_{LLM}(\tau_{t+1:t+h} | \tau_{0:t})$$

Here, we employ greedy sampling to get future trajectories. The simulated trajectory is then used to retrieve the most relevant past experiences to guide policy improvement and future simulations.

4.3 Policy Improvements with Forecast

The forecast simulated by LLMs is relevant to the task and indicates how to carry out the task. However, they may not fully correct or fully match the items or descriptions available for the environment. We cannot directly use the forecast in the actor's in-context prompt. Instead, we use this forecast to retrieve the relevant subtrajectories (called experiences) that the agent has experienced in the environment before. The retrieved experiences are included in the actor's in-context prompt to guide the agent.

Experience Retrieval. We consider that experiences are related to the current state in two ways. (1) *Task-based*: The experiences have a similar objective to the current subgoal. (2) *State-based*:



Algorithm 1: Dynamic Prompt Trimming

Require: Prompt p; Trajectories $H, \hat{H}$; Experience set $\hat{E}$; Thresholds $\tau_{\text{tok}}, \tau_H, \tau^{\hat{H}}$. ▷ $i^C_{start/end}$: *start/end index of component C*
Ensure: Trimmed prompt p_{tok} s.t. $\text{LEN}(p_{\text{tok}}) \leq \tau_{\text{tok}}$.

while $\text{len}(p_{\text{tok}}) > \tau_{\text{tok}}$ **do**
  if $\text{len}(H) > \tau^H$ **then**
    $i^H_{\text{start}} \leftarrow i^H_{\text{start}} + 1$ ▷ *Trim current trajectory*
    if $i^{\hat{H}}_{\text{end}} < i^{\hat{H}}_{\text{original\_end}}$ **then**
      $\{i^{\hat{H}}_{\text{start}}, i^{\hat{H}}_{\text{end}}\} \leftarrow \{i^{\hat{H}}_{\text{start}}, i^{\hat{H}}_{\text{end}}\} + 1$ ▷ *Window sliding*
    end if
  else if $\text{LEN}(\hat{H}) > \tau^{\hat{H}}$ **then**
    $i^{\hat{H}}_{\text{end}} \leftarrow i^{\hat{H}}_{\text{end}} - 1$ ▷ *Trim example's trajectory*
  else if $\text{LEN}(\hat{E}) > 0$ **then**
    $\hat{E} \leftarrow \hat{E} \setminus \hat{E}_{-1}$ ▷ *Trim experience*
  else
    $\{\tau^H, \tau^{\hat{H}}\} \leftarrow \{\tau^H, \tau^{\hat{H}}\} - 1$ ▷ *Decrease the thresholds*
  end if
end while

Figure 3: Experience Retriever in MENSA: (a) Task-based (b) State-based

The experiences have a similar initial state to the current state. In the task-based retrieval, we employ a semantic classifier, Bart MultiNLI [17], to classify whether the forecast F aligns with the subgoal of an experience. The experiences are ranked based on the predicted probabilities produced by the classifier. We select the top-k as the task-based experiences. In the state-based retrieval, we compute the cosine similarity between the text embeddings of the initial states of the subtrajectories in experiences and the current state to identify the top-k state-based experiences. This ensures that agent can seamlessly transition using experiences retrieved from the current state. We concatenate both types into a single sequence. However, incorporating these experiences expands the context window, making LLMs susceptible to primacy and recency biases [8, 22]. To mitigate these biases, we reverse the sequence of the concatenated experiences before adding the retrieved experience to the prompt. Figure 3 shows the details of this retrieval process.

Dynamic Prompt Trimmer. LLMs have fixed context window sizes. As we add more related experiences to the context, they may struggle with long in-context learning. Several prior works have shown that LLM performance declines as input length increases, especially in smaller models [19, 23]. Existing methods typically trim trajectories naively. This may lead to a loss of crucial historical context. To address this challenge, as detailed in Algorithm 1., we dynamically trim the prompt by retaining the most relevant input segments while discarding less critical information, ensuring alignment with the ongoing context.

5 EXPERIMENTS AND RESULTS

5.1 Environments

We evaluate MENSA and baseline methods on two text-based interactive environments: ScienceWorld and NetHack, which allow agents to interact and perform complex tasks. For both environments, we follow the original benchmark designs and prior work regarding episode definitions and evaluation protocols to ensure comparability with existing research. An episode terminates upon task completion, failure, or reaching the maximum step limit, which is set to 1.5 times the length of the gold trajectory. The "`think`" action is excluded from step counting.

ScienceWorld. This environment evaluates agents' scientific reasoning through elementary science experiments conducted in a text-based environment. Following prior works, e.g., CLIN, SSO, and SwiftSage[20], we select 18 task classes, with five variants randomly sampled per class to ensure evaluation diversity. These variants differ in critical objects, starting locations, and environmental configurations. We adopt the same one-shot example setting as SwiftSage for both baseline methods and MENSA. Agent performance is measured using a subtask-based reward function that assigns scores on a scale from 0 to 100.

NetHack. It is a procedurally generated environment that emphasizes spatial understanding and low-level navigation. Unlike ScienceWorld, it primarily consists of primitive movement actions (e.g., `move north`, `move south`), with only a limited number of higher-level commands (e.g., `apply`, `open`), posing additional challenges for LLM agents. We adopt the *Crossing Lava* task, where the agent must navigate the map to acquire a key, unlock a door, use an item, and safely cross the lava to reach the goal. The environment introduces stochasticity through randomized starting positions and key entity placements. Agents are evaluated using a stage-based reward system that assigns scores from 0 to 100 based on partial goal completion.

Evaluation Settings. We consider two evaluation settings: *adaptation* and *transfer*. The adaptation setting evaluates an agent's ability to continuously learn within the same task. All LLM agents begin in their initial state without prior training. Each test variant is evaluated over five episodes, with learning occurring between consecutive episode. The transfer setting accesses an agent's ability to generalize learned experiences to new tasks. We evaluate transfer performance only on ScienceWorld, as it provides a larger number of task variants, allowing us to construct disjoint training

and test sets. In this setting, the agent is trained on 15 trajectories drawn from 5 distinct task variants, with 3 trajectories per variant. This setup enables the agent to learn generalized strategies across diverse scenarios. During evaluation, the agent is tested on novel variants of the same task that were not encountered during training, assessing its ability to generalize to unseen variants.

5.2 Baselines

We compare MENSA with state-of-the-art gradient-free methods that employ few-shot in-context learning. The following methods are selected for evaluation:

- **ReAct** [39] synergizes between reasoning and action traces, helping agents effectively solve complex tasks by generating additional '`think`' steps for reasoning and the intended next actions.
- **Reflexion** [30] instructs the ReAct-based LLM to generate verbal reflections between trials, emphasizing the analysis of previous unsuccessful attempts. This process aims to facilitate learning from past mistakes to improve performance in subsequent trials.
- **SSO** [27] adopts comprehensive instructions to gather skill set from past trajectories over long-term episodes. This method accumulates and refines of skills over episodes.

5.3 Implementation Details

To highlight the improvements across different types of LLMs, we evaluated the agent frameworks with a diverse set of LLMs ranging from high-performance models to more compact, limited ones. For the stronger model, we chose a proprietary model, GPT-4o-mini, accessible through the OpenAI API. For the median and weaker models, we employ open-sourced models with parameter sizes ranging between 2B and 9B, as shown in Table 1. We use the base LLMs in all LLM agents except for Phi-3-small, Phi-3-mini, and all SSO experiments, which use instruction-tuned models as no base model is available or the approach is incompatible with the base models' capabilities (i.e., requires instruction following capability). To manage the API of open-source models, we use vLLM [16] in all experiments. All random seeds were set to 42.

We use SentenceBERT *paraphrase-MiniLM-L6-v2* model [29] to extract the text embeddings for experience construction, state-based experience retrieval, and text similarity measurements. We use *BART-large-mnli* model [17] as the semantic classifier used in the target-based experience retrieval.

5.4 Results and Analysis

Table 1 and Table 2 show the comparison between MENSA and baseline methods. Notably, unlike previous works that often restrict their validation to a single, contemporary SOTA LLM, we present a systematic evaluation across multiple LLM, allowing us to understand how different methods generalize under models with diverse capacities. Across all model configurations, MENSA consistently achieves higher cumulative rewards, demonstrating its generalization ability and robustness. We apply a Wilcoxon signed-rank test over 89 scores across the 18 task classes. Statistical significance markers are indicated in the '$\Delta_{\text{top1-top2}}$' column.

Table 1: Performance of different methods across LLMs in ScienceWorld's adaptation and transfer settings. The performance difference $\Delta_{top1-top2}$ presents the difference in average scores between the best and second-best methods.

	LLM	ReAct	Reflexion	SSO	MENSA	$\Delta_{\text{top1-top2}}$
Adaptation	*Closed-Source Large*					
	GPT-4o-mini	22.4*	25.9*	54.5*	**70.3***	+15.8‡
	Open-Source Small					
	Gemma-2-9B	29.9	31.2	30.7*	**42.0**	+10.8†
	Llama-3-8B	26.7	31.9	41.6*	**45.0**	+3.4
	Phi-3-small (7B)	18.1*	23.6*	26.4*	**58.8***	+32.4‡
	Mistral-7B	24.5	26.4	13.1*	**44.3**	+17.9‡
	Open-Source Mini					
	Phi-3-mini (4B)	6.5*	7.2*	20.7*	**32.6***	+11.9‡
	Gemma-2-2B	17.9	20.6	10.5*	**29.9**	+9.3‡
Transfer	*Open-Source Small*					
	Gemma-2-9B	–	–	15.9*	**35.8**	+19.9‡
	Llama-3-8B	–	–	20.8*	**39.4**	+18.6‡

*: Using instruction-tuned model.
†: Indicates statistical significance at the $p < 0.05$ level.
‡: Indicates strong statistical significance at the $p < 0.01$ level.

Table 2: Performance of SSO and MENSA in NetHack's adaptation setting.

LLM	ReAct	Reflexion	SSO	MENSA
GPT-4o-mini	12.0*	20.0*	10.0*	**50.0***
Llama-3-8B	6.0	8.0	0.0*	**32.0**

*: Using instruction-tuned model.

Adaptation. In the adaptation setting, MENSA outperforms the previous state-of-the-art (SSO), achieving a maximum improvement of +32.4 points with Phi-3-small. Additionally, MENSA demonstrates gains of +11.9 and +15.8 points on Phi-3-mini and GPT-4o-mini, respectively. Similarly, in NetHack, MENSA consistently achieves substantial improvements over SSO for +40.0 and +32.0 points across both large and small models, providing evidence that its effectiveness generalizes beyond a single scenario.

Transfer. In the transfer setting, we perform experiments using two representative models, Llama-3-8B [6] and Gemma-2-9B [33]. MENSA surpasses previous SOTA by an improvement of +18.6 using Llama-3-8B, and +19.9 using Gemma-2-9B. This shows that our model-based policy learning can acquire not only in-domain knowledge but also transferable experience.

Performance Evolution Over Episodes. We analyze the agent's performance across episodes. In this setup, the agent continuously adapts and learns from interactions with the environment. Reflexion, SSO, and MENSA accumulate reflections, skills, and experiences, respectively. Figure 4 compares the episode-wise performance of MENSA and the baselines over episodes under different LLM backbones. MENSA demonstrates a generally upward growth trend across episodes. In contrast, the baselines exhibit only marginal improvements or even performance degradation as the number of episodes increases. These results highlight MENSA's ability

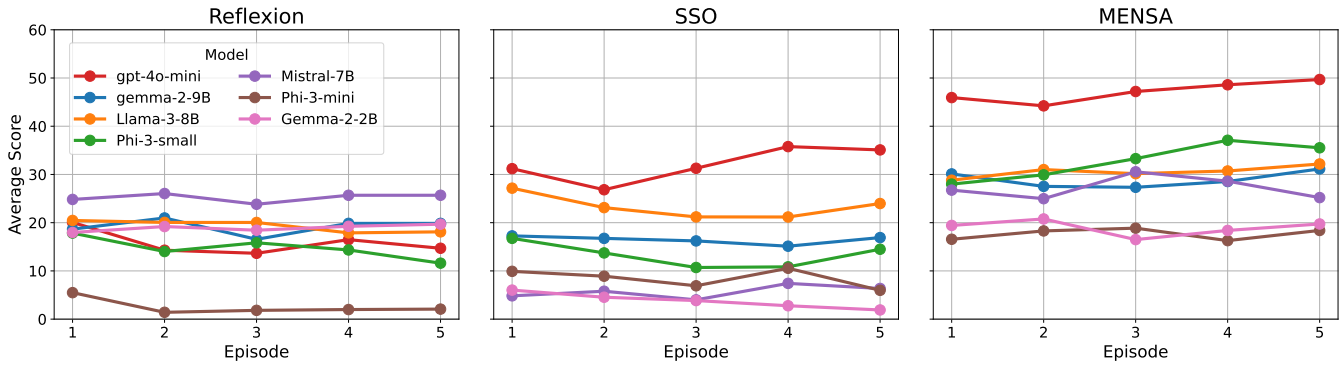


Figure 4: Episode-wise performance trends of MENSA and baselines with different LLMs in ScienceWorld's adaptation settings, with performance computed as the mean score over all tasks. MENSA shows a generally upward trajectory across episodes, while baselines exhibit marginal gains or declining trends, indicating the advantage of experience accumulation in MENSA.

to effectively leverage accumulated experience compared to the baseline methods.

Quality of Forecasts. We evaluate forecast quality using (i) the similarity to ground-truth short-horizon trajectories (number of forecast steps = 3), and (ii) the failure rate of forecast generation, where invalid or empty outputs are regarded as failures. As a point of comparison, we also analyze the subgoals generated by SSO. As shown in Figure 5, MENSA achieves higher similarity and success rate than SSO, indicating that MENSA can reliably generate forecasts and the generated forecasts are more relevant to the executable plans.

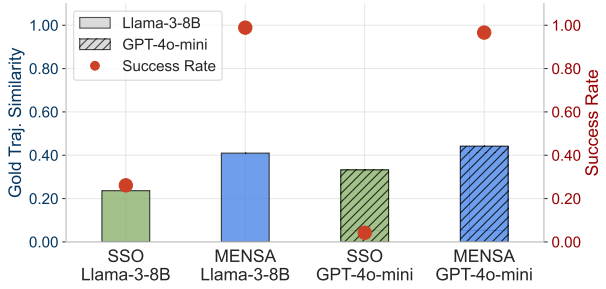


Figure 5: The generation success rate and quality of the forecasts(MENSA)/subgoals(SSO) on GPT-4o-mini and Llama-3-8B. The red dots indicate the success rate of generation. The bar charts show the cosine similarity between the generated forecasts/subgoals and ground-truth subtrajectories.

5.4.1 Impact of Context Length Restriction. We further investigate the effect of different context lengths on MENSA's performance. As shown in Table 3, reducing the context window from 4k/8k to 2k incurs only a small performance loss of about 2-3 points, suggesting robustness to tight context limits. The results underscore that the dynamic prompt trimmer effectively preserves key contextual information even when the available context is substantially reduced.

Table 3: MENSA performance with different context-length restrictions in ScienceWorld.

LLM	8k	4k	2k
Gemma-2-9B	42.0	39.3	38.92
Llama-3-8B	45.0	46.3	43.53

5.4.2 Impact of Forecast Steps. To understand how the number of forecast steps affects the agent, we plot the performance score and the token cost at different numbers of forecast steps in Figure 6.

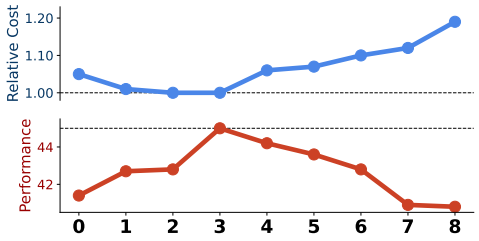


Figure 6: Performance and cost with different forecast step count. The cost calculation is based on the sum of input tokens and three times the output tokens, reflecting a similar pricing structure to the APIs.

Experience Retrieval. In the experience retrieval process, the forecast is used to match the entries in the experience set. To determine the optimal number of forecast steps, we test a range of forecast step counts from 0 to 8. When the forecast step count is 0, that means that no experience will be retrieved through the forecast . We found that a forecast step count of 3 achieves the best performance, after which the effectiveness declines. The results indicate that the length of the forecast significantly affects the effectiveness of the experience.

Token Cost. We also analyzed the relationship between operating costs and forecast steps as shown in Figure 6. We found that

Table 5: Performance of different experience ordering in target-based retrieval.

Ordering	Score
Reversed	**45.0**
Non-Reversed	42.4
Shuffled	42.8

performance improves and costs decrease when the number of forecast steps is ≤ 3. This suggests that forecasting not only enhances the agent's success rate but also reduces operational costs, as the agent can complete tasks with less effort (i.e., tokens). However, increasing the number of forecast steps beyond 3 negatively impacts both performance and token costs. We identify two primary failure cases associated with this increase. First, LLMs struggle to generate relevant actions and outcomes for later forecast steps due to limitations in their capabilities. Second, even when forecasts are accurate, LLMs may fail to select optimal next actions because excessive information from future steps can overwhelm them.

Table 4: *Instruct* and *Plan* abilities evaluated on T-Eval. *Instruct* evaluates response accuracy under given instructions, while *Plan* measures the quality of the LLM's predicted action sequences.

LLM	Instruct	Plan
GPT-4o-mini	100.0	84.2
Gemma-2-9B	99.8	65.0
Llama-3-8B	100.0	60.5
Phi-3-small	87.2	64.8
Mistral-7B	99.2	56.1
Phi-3-mini	95.1	61.6
Gemma-2-2B	99.9	44.4

5.4.3 Impact of LLM Capability. To understand how LLMs' abilities in instruction-following and planning affect the effectiveness of different methods, we first evaluated the performance of various LLMs on instruction-following and planning capabilities defined in T-Eval [4]. We report the ability score of different LLMs in Table 4. The following analysis examines the impact of MENSA and baseline methods in adaptation settings when using LLMs with varying capabilities. Our results in Table 1 show a noticeable performance decline for ReAct and Reflexion when using Phi-3-mini, which has weaker abilities. However, when instruction-following or planning ability is strong, the decline is less pronounced. In contrast, SSO's performance is significantly affected when planning ability falls below 60%, while weaker instruction-following ability has minimal impact. This suggests that SSO is heavily dependent on planning, as it relies on inferring subgoals for skill retrieval. However, MENSA shows better resilience, maintaining stable performance regardless of the ability strength. This indicates that MENSA's mental simulation mechanism effectively leverages basic capabilities for simulation, allowing smaller LLMs to perform better in sequential decision-making tasks.

5.4.4 Ablation Study. We perform a series of ablations with ScienceWorld in the adaptation setting to assess the influence of different factors on experience retrieval in MENSA. Specifically, we examine the effects of experience sequencing in prompts and retrieval types (state-based vs. target-based). For consistency, we use a representative small LLM alongside the well-established Llama-3-8B model throughout these experiments.

Experience Ordering. This ablation study compares different ordering strategies, including reversed, non-reversed, and shuffled sequences, when presented to the LLM. A sequence is deemed reversed if it is arranged in ascending order of relevance, as determined by the similarity score in relation to the current target or task. As shown in Table 5, our results indicate that reversed ordering achieves top performance at 45.0, surpassing the shuffled configuration by +2.2 points. Consequently, reversed ordering is utilized in all experiments.

Table 6: Performance of different retrieval types in reversed experience order setting.

Retrieval Type	Score
State-Based Only	41.4
Target-Based Only	43.9
State-Based First	42.9
Target-Based First	**45.0**

Experience Retrieval Approaches. We evaluated several strategies: state-based similarity, target-based similarity, and a combination of both, with varying orderings. Specifically, we examined whether placing state-based experiences before or after target-based experiences would yield better results. Building on our earlier findings on the ordering of experience, we reversed the final sequence. As shown in Table 6, the best performance (45.0) was achieved when target-based experiences were prioritized. This result underscores the value of including both state-based and target-based experiences, with a notable advantage when target-based experiences are more valued.

6 CONCLUSION

This paper introduces MENSA, an LLM agent that implements model-based in-context policy improvement. MENSA leverages text completion, the core capability of every LLM, to perform mental simulation from the current action. The simulated forecast is used to retrieve relevant experiences to improve policy. We show that MENSA outperforms the state-of-the-art methods in the interactive environments, ScienceWorld and NetHack. Our analysis in T-Eval shows that MENSA can effectively improve the performance of the agent powered by weaker LLMs because of the instruction-following ability.

One limitation of MENSA is that it employs LLMs as implicit, language-grounded world models for simulating action outcomes. The policy improvement depends on the fidelity of such mental simulations. Moreover, our evaluation focuses on text-based interactive environments. Extending MENSA to embodied settings would require bridging language-level state abstractions with grounded perception and action grounding, potentially through improved state transition reasoning and the integration of perceptual models.

REFERENCES

[1] Marah Abdin, Jyoti Aneja, Hany Awadalla, Ahmed Awadallah, Ammar Ahmad Awan, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Jianmin Bao, Harkirat Behl, et al. 2024. Phi-3 Technical Report: A Highly Capable Language Model Locally on Your Phone. arXiv:2404.14219 [cs.CL] https://arxiv.org/abs/2404.14219

[2] Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Michal Podstawski, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Hubert Niewiadomski, Piotr Nyczyk, and Torsten Hoefler. 2024. Graph of Thoughts: Solving Elaborate Problems with Large Language Models. *Proceedings of the AAAI Conference on Artificial Intelligence* 38, 16 (Mar. 2024), 17682–17690. https://doi.org/10.1609/aaai.v38i16.29720

[3] Konstantinos Bousmalis, Giulia Vezzani, Dushyant Rao, Coline Devin, Alex X Lee, Maria Bauzá, Todor Davchev, Yuxiang Zhou, Agrim Gupta, Akhil Raju, et al. 2023. Robocat: A self-improving generalist agent for robotic manipulation. *arXiv preprint arXiv:2306.11706* (2023).

[4] Zehui Chen, Weihua Du, Wenwei Zhang, Kuikun Liu, Jiangning Liu, Miao Zheng, Jingming Zhuo, Songyang Zhang, Dahua Lin, Kai Chen, and Feng Zhao. 2024. T-Eval: Evaluating the Tool Utilization Capability of Large Language Models Step by Step. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Lun-Wei Ku, Andre Martins, and Vivek Srikumar (Eds.). Association for Computational Linguistics, Bangkok, Thailand, 9510–9529. https://aclanthology.org/2024.acl-long.515

[5] Jingtao Ding, Yunke Zhang, Yu Shang, Yuheng Zhang, Zefang Zong, Jie Feng, Yuan Yuan, Hongyuan Su, Nian Li, Nicholas Sukiennik, Fengli Xu, and Yong Li. 2024. Understanding World or Predicting Future? A Comprehensive Survey of World Models. arXiv:2411.14499 [cs.CL] https://arxiv.org/abs/2411.14499

[6] Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. 2024. The Llama 3 Herd of Models. arXiv:2407.21783 [cs.AI] https://arxiv.org/abs/2407.21783

[7] Nate Gruver, Marc Finzi, Shikai Qiu, and Andrew G Wilson. 2023. Large language models are zero-shot time series forecasters. *Advances in Neural Information Processing Systems* 36 (2023), 19622–19635.

[8] Xiaobo Guo and Soroush Vosoughi. 2024. Serial Position Effects of Large Language Models. arXiv:2406.15981 [cs.CL] https://arxiv.org/abs/2406.15981

[9] Shibo Hao, Yi Gu, Haodi Ma, Joshua Hong, Zhen Wang, Daisy Wang, and Zhiting Hu. 2023. Reasoning with Language Model is Planning with World Model. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, Houda Bouamor, Juan Pino, and Kalika Bali (Eds.). Association for Computational Linguistics, Singapore, 8154–8173. https://doi.org/10.18653/v1/2023.emnlp-main.507

[10] Shijue Huang, Wanjun Zhong, Jianqiao Lu, Qi Zhu, Jiahui Gao, Weiwen Liu, Yutai Hou, Xingshan Zeng, Yasheng Wang, Lifeng Shang, Xin Jiang, Ruifeng Xu, and Qun Liu. 2024. Planning, Creation, Usage: Benchmarking LLMs for Comprehensive Tool Utilization in Real-World Complex Scenarios. In *Findings of the Association for Computational Linguistics: ACL 2024*, Lun-Wei Ku, Andre Martins, and Vivek Srikumar (Eds.). Association for Computational Linguistics, Bangkok, Thailand, 4363–4400. https://doi.org/10.18653/v1/2024.findings-acl.259

[11] Wenlong Huang, Pieter Abbeel, Deepak Pathak, and Igor Mordatch. 2022. Language Models as Zero-Shot Planners: Extracting Actionable Knowledge for Embodied Agents. In *Proceedings of the 39th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 162)*, Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (Eds.). PMLR, 9118–9147. https://proceedings.mlr.press/v162/huang22a.html

[12] Wenlong Huang, Fei Xia, Ted Xiao, Harris Chan, Jacky Liang, Pete Florence, Andy Zeng, Jonathan Tompson, Igor Mordatch, Yevgen Chebotar, et al. 2023. Inner Monologue: Embodied Reasoning through Planning with Language Models. In *Proceedings of The 6th Conference on Robot Learning (Proceedings of Machine Learning Research, Vol. 205)*, Karen Liu, Dana Kulic, and Jeff Ichnowski (Eds.). PMLR, 1769–1782. https://proceedings.mlr.press/v205/huang23c.html

[13] Minyoung Huh, Brian Cheung, Tongzhou Wang, and Phillip Isola. 2024. The Platonic Representation Hypothesis. arXiv:2405.07987 [cs.LG] https://arxiv.org/abs/2405.07987

[14] Brian Ichter, Anthony Brohan, Yevgen Chebotar, Chelsea Finn, Karol Hausman, Alexander Herzog, Daniel Ho, Julian Ibarz, Alex Irpan, Eric Jang, et al. 2023. Do As I Can, Not As I Say: Grounding Language in Robotic Affordances. In *Proceedings of The 6th Conference on Robot Learning (Proceedings of Machine Learning Research, Vol. 205)*, Karen Liu, Dana Kulic, and Jeff Ichnowski (Eds.). PMLR, 287–318. https://proceedings.mlr.press/v205/ichter23a.html

[15] Heinrich Küttler, Nantas Nardelli, Alexander H. Miller, Roberta Raileanu, Marco Selvatici, Edward Grefenstette, and Tim Rocktäschel. 2020. The NetHack Learning Environment. In *Proceedings of the Conference on Neural Information Processing Systems (NeurIPS)*.

[16] Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient Memory Management for Large Language Model Serving with PagedAttention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*.

[17] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault (Eds.). Association for Computational Linguistics, Online, 7871–7880. https://doi.org/10.18653/v1/2020.acl-main.703

[18] Shuang Li, Xavier Puig, Chris Paxton, Yilun Du, Clinton Wang, Linxi Fan, Tao Chen, De-An Huang, Ekin Akyürek, Anima Anandkumar, et al. 2022. Pre-trained language models for interactive decision-making. *Advances in Neural Information Processing Systems* 35 (2022), 31199–31212.

[19] Tianle Li, Ge Zhang, Quy Duc Do, Xiang Yue, and Wenhu Chen. 2024. Long-context LLMs Struggle with Long In-context Learning. arXiv:2404.02060 [cs.CL]

[20] Bill Yuchen Lin, Yicheng Fu, Karina Yang, Faeze Brahman, Shiyu Huang, Chandra Bhagavatula, Prithviraj Ammanabrolu, Yejin Choi, and Xiang Ren. 2023. SwiftSage: A Generative Agent with Fast and Slow Thinking for Complex Interactive Tasks. In *Advances in Neural Information Processing Systems*, A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (Eds.), Vol. 36. Curran Associates, Inc., 23813–23825. https://proceedings.neurips.cc/paper_files/paper/2023/file/4b0eea69deea512c9e2c469187643dc2-Paper-Conference.pdf

[21] Haoxin Liu, Zhiyuan Zhao, Jindong Wang, Harshavardhan Kamarthi, and B. Aditya Prakash. 2024. LSTPrompt: Large Language Models as Zero-Shot Time Series Forecasters by Long-Short-Term Prompting. In *Findings of the Association for Computational Linguistics: ACL 2024*, Lun-Wei Ku, Andre Martins, and Vivek Srikumar (Eds.). Association for Computational Linguistics, Bangkok, Thailand, 7832–7840. https://doi.org/10.18653/v1/2024.findings-acl.466

[22] Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2024. Lost in the Middle: How Language Models Use Long Contexts. *Transactions of the Association for Computational Linguistics* 12 (2024), 157–173. https://doi.org/10.1162/tacl_a_00638

[23] Taiming Lu, Muhan Gao, Kuai Yu, Adam Byerly, and Daniel Khashabi. 2024. Insights into LLM Long-Context Failures: When Transformers Know but Don't Tell. arXiv:2406.14673 [cs.CL] https://arxiv.org/abs/2406.14673

[24] Bodhisattwa Prasad Majumder, Bhavana Dalvi Mishra, Peter Jansen, Oyvind Tafjord, Niket Tandon, Li Zhang, Chris Callison-Burch, and Peter Clark. 2023. CLIN: A Continually Learning Language Agent for Rapid Task Adaptation and Generalization. arXiv:2310.10134 [cs.CL] https://arxiv.org/abs/2310.10134

[25] Shervin Minaee, Tomas Mikolov, Narjes Nikzad, Meysam Chenaghlu, Richard Socher, Xavier Amatriain, and Jianfeng Gao. 2025. Large Language Models: A Survey. arXiv:2402.06196 [cs.CL] https://arxiv.org/abs/2402.06196

[26] Humza Naveed, Asad Ullah Khan, Shi Qiu, Muhammad Saqib, Saeed Anwar, Muhammad Usman, Naveed Akhtar, Nick Barnes, and Ajmal Mian. 2024. A Comprehensive Overview of Large Language Models. arXiv:2307.06435 [cs.CL] https://arxiv.org/abs/2307.06435

[27] Kolby Nottingham, Bodhisattwa Prasad Majumder, Bhavana Dalvi Mishra, Sameer Singh, Peter Clark, and Roy Fox. 2024. Skill Set Optimization: Reinforcing Language Model Behavior via Transferable Skills. In *Proceedings of the 41st International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 235)*, Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (Eds.). PMLR, 38409–38425. https://proceedings.mlr.press/v235/nottingham24a.html

[28] Yiwei Qin, Kaiqiang Song, Yebowen Hu, Wenlin Yao, Sangwoo Cho, Xiaoyang Wang, Xuansheng Wu, Fei Liu, Pengfei Liu, and Dong Yu. 2024. InFoBench: Evaluating Instruction Following Ability in Large Language Models. In *Findings of the Association for Computational Linguistics ACL 2024*, Lun-Wei Ku, Andre Martins, and Vivek Srikumar (Eds.). Association for Computational Linguistics, Bangkok, Thailand and virtual meeting, 13025–13048. https://aclanthology.org/2024.findings-acl.772

[29] Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics. http://arxiv.org/abs/1908.10084

[30] Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2023. Reflexion: language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems*, A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (Eds.), Vol. 36. Curran Associates, Inc., 8634–8652. https://proceedings.neurips.cc/paper_files/paper/2023/file/1b44b878bb782e6954cd888628510e90-Paper-Conference.pdf

[31] Hua Tang, Chong Zhang, Mingyu Jin, Qinkai Yu, Zhenting Wang, Xiaobo Jin, Yongfeng Zhang, and Mengnan Du. 2024. Time Series Forecasting with LLMs: Understanding and Enhancing Model Capabilities. arXiv:2402.10835 [cs.CL] https://arxiv.org/abs/2402.10835

[32] Shelley E. Taylor and Sherry K. Schneider. 1989. Coping and the Simulation of Events. *Social Cognition* 7, 2 (1989), 174–194. https://doi.org/10.1521/soco.1989.7.2.174 arXiv:https://doi.org/10.1521/soco.1989.7.2.174

[33] Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, LÃĭonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre RamÃĭ, et al. 2024. Gemma 2: Improving Open Language Models at a Practical Size. arXiv:2408.00118 [cs.CL] https://arxiv.org/abs/2408.00118

[34] Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. 2024. Voyager: An Open-Ended Embodied Agent with Large Language Models. *Transactions on Machine Learning Research* (2024). https://openreview.net/forum?id=ehfRiF0R3a

[35] Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, Wayne Xin Zhao, Zhewei Wei, and Jirong Wen. 2024. A survey on large language model based autonomous agents. *Frontiers of Computer Science* 18, 6 (March 2024). https://doi.org/10.1007/s11704-024-40231-1

[36] Ruoyao Wang, Peter Jansen, Marc-Alexandre Côté, and Prithviraj Ammanabrolu. 2022. ScienceWorld: Is your Agent Smarter than a 5th Grader?. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang (Eds.). Association for Computational Linguistics, Abu Dhabi, United Arab Emirates, 11279–11298. https://doi.org/10.18653/v1/2022.emnlp-main.775

[37] 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

[38] Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik R Narasimhan. 2023. Tree of Thoughts: Deliberate Problem Solving with Large Language Models. In *Thirty-seventh Conference on Neural Information Processing Systems*. https://openreview.net/forum?id=5Xc1ecxO1h

[39] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. 2023. ReAct: Synergizing Reasoning and Acting in Language Models. In *The Eleventh International Conference on Learning Representations*. https://openreview.net/forum?id=WE_vluYUL-X

[40] Kechi Zhang, Jia Li, Ge Li, Xianjie Shi, and Zhi Jin. 2024. CodeAgent: Enhancing Code Generation with Tool-Integrated Agent Systems for Real-World Repo-level Coding Challenges. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Lun-Wei Ku, Andre Martins, and Vivek Srikumar (Eds.). Association for Computational Linguistics, Bangkok, Thailand, 13643–13658. https://doi.org/10.18653/v1/2024.acl-long.737

[41] Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. 2024. ExpeL: LLM Agents Are Experiential Learners. *Proceedings of the AAAI Conference on Artificial Intelligence* 38, 17 (Mar. 2024), 19632–19642. https://doi.org/10.1609/aaai.v38i17.29936

[42] Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. 2023. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911* (2023).