LEAP & LEAN: Look-ahead Planning and Agile Navigation for LLM Agents

Nikhil Verma
LG Electronics, Toronto AI Lab
nikhil.verma@lge.com

Manasa Bharadwaj
LG Electronics, Toronto AI Lab
manasa.bharadwaj@lge.com

Abstract

Foundational models endowed with emergent abilities are increasingly deployed as autonomous agents to navigate intricate environments. Despite their capability to comprehend human intentions, even when paired with reasoning traces, they struggle to achieve robust autonomy. In this work, we introduce **LEAP & LEAN**, a novel paradigm designed to enhance the performance of Large Language Models (LLMs) as autonomous agents. LEAP employs look-ahead planning to refine action selection, while LEAN streamlines navigation through agile prompt construction, enabling more efficient task completion. Together, LEAP & LEAN address the explore-exploit dilemma, fostering optimal decision-making and improving task performance. We evaluate our framework across diverse, multi-faceted task-oriented domains (WebShop, ALFWorld, and TravelPlanner) using both proprietary and open-source LLM agents. Notably, without any fine-tuning, our framework outperforms agents trained via imitation learning, reinforcement learning, and reasoning-based approaches. Our findings underscore the importance of action and prompt curation to create robust and efficient fully autonomous LLM agents.

1 Introduction

The advent of foundational models has triggered a significant increase in their deployment as fully autonomous decision-making agents, driven by their remarkable emergent abilities (Wei et al., 2022a). Training large-scale models with extensive datasets improves language understanding (Hoffmann et al., 2022), but their ability to function independently across diverse environments is limited by their inadequate planning capabilities compared to humans (Liu et al., 2023; Yao et al., 2022a). More conventionally, using imitation or reinforcement learning (IL, RL) techniques rely on human demonstrations of action traces for training the models (Shridhar et al., 2020b; Fereidouni and Siddique, 2024).

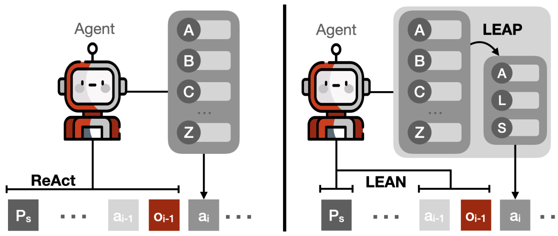

Figure 1: Reasoning strategies (such as ReAct (Yao et al., 2022b)) often use complete context to determine the next action, while our framework only uses curated context (LEAN) along with potential high-reward actions obtained via look-ahead planning (LEAP).

Self-generated verbal reasoning traces, such as thoughts, have proven effective in improving LLM performance across logical tasks like arithmetic and commonsense reasoning, using strategies such as chain-of-thought prompting (Wei et al., 2022b). Similarly using few-shot prompts (human demonstrations) with verbal reasoning have been leveraged in approaches like WebGPT (aka Act) (Nakano et al., 2021) and ReAct (Yao et al., 2022b) to navigate autonomous agent environments. Figure 1 illustrates how ReAct uses complete history of actions with reasoning trace, to determine the next plausible action from the action space. Further, Reflexion (Shinn et al., 2024) incorporates a memory component along with reason-to-act signals to track action history of the agent. Overall, human demonstrations help in utilizing the instruction-following capabilities of LLM for navigation while verbal reasoning serves as an implicit planning methodology that helps LLM select the most appropriate actions.

Although existing approaches effectively enable LLM agents to operate autonomously and make informed decisions, we observed a high rate of inefficient planning, particularly the inability to complete tasks within a predetermined step limit. Upon qualitative analysis, we categorized the inefficiencies as: (1) Unanticipated action suggestion, i.e.

Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 6: Industry Track), pages 896–933
July 28-30, 2025

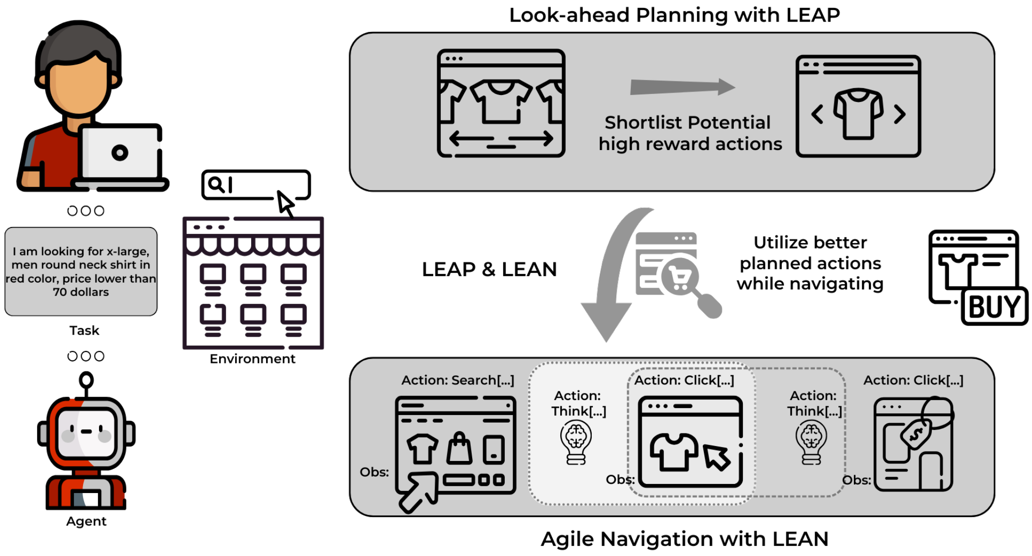

Figure 2: Workflow of the LEAP & LEAN paradigm: Systematically exploring the action space for optimized planning (LEAP) along with strategically limiting context in prompts for efficient navigation (LEAN), balancing exploration and exploitation (cf. Algorithm 1).

generating non-existing actions due to oversight of possible action space; (2) Contextual Stagnation, referring to repetitive action prediction due to failure to update context; (3) Proactive action planning which leads to pre-emptive decision-making based on generic knowledge of the LLM. This performance gap is especially pronounced when applied to LLMs, with fewer than 10 billion parameters, designed for efficiency (Liu et al., 2023). (cf. Appendix A)

We hypothesize that optimizing the performance of LLMs (of varying sizes) requires strategically designed prompts to guide actions, combined with a decoupled exploration of the full action space to facilitate well-informed decision-making. We propose LEAP & LEAN, a novel and modular paradigm designed to enhance the planning and navigation capabilities of LLMs, creating efficient and robust autonomous agents. On one hand, integrating LEAP allows the LLM to systematically explore the entire action space, thereby reducing unanticipated action suggestions. On the other hand, incorporating LEAN facilitates the strategic generation of lightweight prompts that contain only highly relevant trajectory history, preventing contextual stagnation. LEAP & LEAN can be employed at each phase of the trajectory.

These modular, plug-and-play components strike a balance between look-ahead exploration and targeted exploitation (shown in Figure 1). Figure 2 illustrates the workflow of our paradigm, demonstrated through a WebShop example of purchasing a men's round-neck shirt. To evaluate our framework, we employed diverse, multi-faceted task-oriented domains, including interactive decision-making for WebShop (Yao et al., 2022a), embodied reasoning using the ALFWorld (Shridhar et al., 2020b), and long-horizon, multi-day itinerary scheduling for TravelPlanner (Xie et al., 2024). We outperform state-of-the-art solutions and fine-tuned models, demonstrating a significant improvement over the base prompting frameworks. These results underscore the effectiveness of incorporating our modular LEAP & LEAN components into agentic frameworks, driving superior performance and adaptability.

The main contributions of this paper are:

1. We present the *LEAP* framework, which employs look-ahead planning for action-selection in dynamic environments.

2. We introduce *LEAN* prompting, which adaptively selects fine-grained segments of the current context to efficiently focus on the information required for the next action generation.

3. We empirically demonstrate that the modular integration of *LEAP & LEAN* significantly boosts the performance of LLM-based agents across diverse, multi-faceted, task-oriented domains.

2 Related Works

2.1 LLM-as-agents

As LLMs evolve in their ability to tackle real-world tasks, they are increasingly being deployed as autonomous agents to navigate complex environments. These agents leverage reasoning to decompose overarching goals into manageable subgoals, a strategy exemplified by systems like AutoGPT (Yang et al., 2023).These advancements underscore the importance of thoroughly evaluating the effectiveness of various LLMs when deployed as autonomous agents. Addressing this need, numerous benchmarks have been proposed including WebShop (Yao et al., 2022a), ALFWorld (Shridhar et al., 2020b), TravelPlanner (Xie et al., 2024) which have been chosen for this study due to the large action spaces and requirement of long-horizon planning.

2.2 Learning based approaches

Conventionally, imitation and reinforcement learning models have utilized human-generated trajectories to train agents to replicate human behavior in action selection while navigating environments (Yao et al., 2022a; Fereidouni and Siddique, 2024; Deng et al., 2024). RetLLM (Modarressi et al., 2023) used structured "triplet-natural language" pairs, while ToolLLM (Qin et al., 2023) and ToolFormer (Schick et al., 2023) use synthetic datasets to instill tool usage capacity. Our approach does not require any fine-tuning and instead relies upon the implicit knowledge of LLMs.

2.3 Reasoning and Planning Strategies

Recent advancements have significantly enhanced the planning capabilities of LLMs (Men et al., 2024) by leveraging reasoning traces, with methods such as Chain-of-Thought (Wei et al., 2022b) and numerous prompting techniques (Zhou et al., 2022; Wang et al., 2022; Zheng et al., 2023a,b), improving their thinking styles. Using LLMs as an autonomous agent, WebGPT (Nakano et al., 2021) used prompting with in-context example to improve upon task at hand. Further improving and utilising reasoning for planning, ReAct (Yao et al., 2022b) combines verbal reasoning and acting with language models. Reflexion (Shinn et al., 2024) and similar works (Zeng et al., 2024), building on the self-refine (Madaan et al., 2023) framework, exemplifies methods that allow LLMs to critique and iteratively refine their outputs, aiming to overcome limitations and improve solution quality. Despite the growth of complex prompting strategies for LLM agents (Wang et al., 2023; Song et al., 2023; Koh et al., 2024), we used ReAct as our base strategy due to the simplicity and robustness across numerous benchmarks.

3 LEAP & LEAN

Background: Consider a typical environment setup, where an agent interacts with an environment E to perform a task T with the description d_T. At each time step, the agent performs an action $a \in \mathcal{A}$ and receives a resulting observation $o \in \mathcal{O}$, such that $o \leftarrow E(a)$. Inspired by (Yao et al., 2022b), we augment agent's action space as $\hat{\mathcal{A}} = \mathcal{A} \cup L$ where L denotes the language space of the LLM agent $\mathcal{L}$. This enables the generation of a verbal reasoning trace, $\hat{a} \in L$, accompanied by neutral environmental feedback $\hat{o}$ (e.g., 'OK.'), effectively injecting thought information into the overall context $\mathcal{C}$, thereby allowing the agent to generate its next action in a more informed manner using $\mathcal{C}$. The overall context $\mathcal{C}$ refers to the concatenation of the task description d_T and the sequence of action-observation-reason at each time step $t \in Z_+$, represented as $\mathcal{C} = \{d_T, (a_t, o_t, \hat{a}_t, \hat{o}_t) \mid t \in Z_+\}$.

We propose LEAP & LEAN as an efficient framework of LLMs for agentic workflows. The overall methodology is formally outlined in Algorithm 1. It primarily consists of two stages of execution at each iteration. In the first stage, look-ahead planning is performed to explore possible future states and identify potentially high reward actions. In the second stage, one of these actions are executed in the environment using a strategically designed prompt structure, containing reasoning traces to guide the progress. Finally, the environment evaluates task success by calculating the success rate (r) if the task is accomplished within a predefined step limit S.

3.1 Stage I: Look-ahead Planning - LEAP

Initially, an LLM agent evaluates potential actions by examining the possible action space (i.e. $A_p \leftarrow \mathcal{L}(d_{\text{T}}, pairs)$) with pairs comprising of action and respective observation ($o \leftarrow E(a)$ and $pairs \leftarrow (a, o)$). Such look-ahead reduces the exploration space by matching the available environment information with task requirements, thereby choosing a limited set of the potential high-reward actions for goal completion. The idea of using

Algorithm 1 LEAP & LEAN Methodology

Input:
Task T with description d_{T}
LLM agent $\mathcal{L}$
Environment E producing observations ($\in \mathcal{O}$) upon receiving actions ($\in \mathcal{A}$)
Pre-determined step limit S
Output:
Task success rate r for task T

Set environment E for task T
$i := 0$
while $i \leq S$ **do**
  Let the possible action-space be A_i
  ▷ ***Stage I: Look-ahead Planning***
  Initialize potential actions, $A_p \leftarrow [\,]$
  Collect all action-observation $pairs$
  for each action a in A_i **do**
    $pairs \leftarrow (a, o)$ where $o \leftarrow E(a)$
  end for
  Agent selects potential high reward actions
  $A_p \leftarrow \mathcal{L}(d_{\text{T}}, pairs)$
  ▷ ***Stage II: Agile Navigation with Planning***
  Generate $reason$ to act while navigating
  $reason \leftarrow \mathcal{L}(d_{\text{T}}, A_p)$
  Use $reason$ to find optimal next action
  $a_{\text{next}} \leftarrow \mathcal{L}(d_{\text{T}}, A_p, reason)$
  if a_{next} corresponds to final state **then**
    Calculate r
    return r
  end if
  $i := i + 1$
end while
return 0

look-ahead planning for action exploration progressively unveils pertinent details, facilitating informed decision-making while minimizing the impact of irrelevant options. LEAP stage provides the subsequent LEAN stage with pertinent information about potential actions and consequences to reduce the exploration.

3.2 Stage II: Agile Navigating with Planning - LEAN

LEAN is specifically designed to enhance the performance of LLMs of varying sizes (especially smaller LLMs), which often struggle to process the full action space and in-context examples efficiently, leading to hallucinated actions when faced with excessive context. To address this, LEAN employs a selective prompting strategy that utilizes only the most meaningful segments from $\mathcal{C}$ at each decision point, rather than relying on the complete context. During this stage, a reasoning trace ($reason$) and the next action (a_{next}) are generated, with actions selected from a pool of high-potential candidates (A_{p}) identified in the earlier LEAP stage. LEAN's segment selection strategy is applied to both reasoning trace generation and action generation. Relevant segments can be derived using approaches such as heuristics or retrieval; in this work, we adopt heuristics due to their simplicity and low computational overhead. Segment curation is applied to both in-context examples and the current task context, providing a carefully curated subset of examples alongside highly relevant subsections of task progress during each action generation phase. This dual simplification of the prompt enhances its clarity, making it easier for instruction-following LLMs to comprehend and respond effectively.

Overall, LEAP explores the full action-space to identify potential high reward actions while LEAN constructs clear concise prompts for efficient navigation. Their integration effectively decouples the tasks of planning and navigation, preventing the LLM from being overwhelmed by excessive exploration and overthinking, thereby enhancing goal achievement efficiency.

4 Experimental Details and Results

We conducted experiments in complex decision-making environments characterized by an expansive action space to evaluate the effectiveness of the LEAP and LEAN paradigms. The dynamic environments we considered are WebShop (Yao et al., 2022a), ALFWorld (Shridhar et al., 2020b) and TravelPlanner (Xie et al., 2024). All the environments feature large action spaces to explore while traversing and offering sparse rewards, with no partial rewards during exploration; agents receive rewards only upon task completion, necessitating effective reasoning to navigate and explore over long horizon.

4.1 Experimental Setup

We primarily evaluated our framework using Gemma-2-9B and the Gemini model. Additionally, our extended evaluation covered six efficient open-source LLM agents (ranging from 2.7B to 9B parameters) and two large API-based LLMs, including Gemini and GPT-3.5, ensuring a diverse

Model	#Size	Form	Version	Creator
Phi-2 (Javaheripi et al., 2023)	2.7B	open	v2.0-instruct	Microsoft
Qwen-4B (Team, 2024)	4B	open	v1.5-chat	Alibaba
Vicuna-7B (Zheng et al., 2024)	7B	open	v1.5-chat	Lmsys
Qwen-7B (Team, 2024)	7B	open	v1.5-chat	Alibaba
Llama-3.1-8B (Dubey et al., 2024)	8B	open	v3.1-instruct	Meta
Gemma-2-9B (Team et al., 2024)	9B	open	v2.0-instruct	Google
GPT-3.5 (OpenAI, 2022)	N/A	API	turbo-0125	OpenAI
Gemini (Reid et al., 2024)	N/A	API	v1.5-flash	Google

Table 1: Models utilized for the assessment of LEAP & LEAN in autonomous system environment.

representation of model families across all experiments. Their key properties are summarized in Table 1. To ensure the reproducibility and consistency of LLM-generated outputs across all experimental settings, the following hyperparameters were meticulously maintained: a deterministic temperature value of 0, a nucleus sampling probability of $top\_p = 0.7$, a token sampling limit of $top\_k = 50$, and a repetition penalty set to 1. They ensure controlled exploration within the model's probabilistic output space while preserving fidelity to the input context. For our comparative analysis, we selected the ReAct framework as the baseline due to its well-established effectiveness and widespread application across various reasoning and planning benchmark studies. In contrast, the Reflexion framework was excluded from our evaluation, as it demonstrated challenges with local minima and failed to show significant improvements, even when utilizing GPT-4 in the WebShop and TravelPlanner environments (Shinn et al., 2024).

4.2 Interactive decision-making: WebShop

It is a synthetic online shopping environment with 1.18 million Amazon items and over 12,000 user instructions for purchasing. An example instruction is: "*i would like a extra round 53mm brush for hair styling, and price lower than 40.00 dollars*". Agents must understand human-provided textual instructions to select products matching specific criteria. For each task, the user enters a text query into search bar, and the system displays the top 50 matching search results, defining the initial action space. Performance is measured by Task Score, reflecting the alignment between the purchased product and the goal, and Success Rate, indicating the percentage of perfect matches.

For baseline comparison, we examined: 1) Rule-based system that selects the first item appearing in the search results; 2) Learning-based models trained with human demonstrations using imitation and reinforcement learning techniques; and 3) ReAct strategy, which utilizes reasoning traces generated by the LLM-as-agent to navigate, plan, and update item selection. For search page planning, we leveraged the titles and prices of up to 50 products displayed on the search results page, narrowing down potential matches to the top 5 candidates. For product page planning, we utilized detailed product descriptions, attributes, options, and pricing information to identify the most suitable match for the user's requirements. The navigation process evaluates the shortlisted options and recommends the next action. This was further supported by providing relevant in-context example chunks to guide decision-making effectively. For LEAP & LEAN we used a step-size limit of 30-steps. (Refer to Appendix B for details on the environment and evaluation metrics, and to Appendices E and F for the prompts utilized.)

Results: To evaluate the effectiveness of our proposed methodologies, we first conducted experiments with the LEAP and LEAN components independently. For LEAP, we utilized the top-5 products as actions on search page for exploration and identified the product most relevant to the user query. For LEAN, we focused on product selection using reasoning and incorporated relevant in-context example chunks into prompt construction. Table 2 summarizes the performance of LLM-as-agent in the WebShop environment with ReAct, LEAP, and LEAN strategies, highlighting significant improvements in planning and navigation capabilities achieved by leveraging our methodologies.

Compared to the performance of ReAct, which achieved a Task Score of 13.1% and a Success Rate of 4.0% with the Gemma-2-9B agent, the LEAP method significantly improved these metrics to 63.1% and 27.4%, respectively. Additionally, LEAN alone achieved scores of 45.0% and 25.8% for the two metrics. We also evaluated an integrated approach that combined LEAP's high-potential item selection with LEAN's navigation flow, yielding stronger performance than either strategy individually. This combined methodology achieved the highest overall performance, with a Success Rate of 27.6%, surpassing LEAP alone (27.4%) and LEAN alone (25.8%) in the WebShop environment. A similar trend was observed with the Gemini model.

WebShop environment		
	Task Score	**Success Rate**
Rule-based	44.8	9.2
Learning-based baseline models (Yao et al., 2022a)		
IL	60.4	28.0
IL + RL	62.4	28.7
Open-source LLM - Gemma-2-9B		
ReAct	13.1	4.0
LEAP	**63.1**	27.4
LEAN	45.0	25.8
LEAP & LEAN	50.8	**27.6**
API-based LLM - Gemini		
ReAct	35.4	21.8
LEAP	**70.4**	42.8
LEAN	53.6	35.0
LEAP & LEAN	62.6	**44.0**
Human Expert	**82.1**	**59.6**

Table 2: Task Score and Success Rate (%) of utilizing LLM-as-agents with LEAP and LEAN strategies on WebShop.

4.3 Embodied Reasoning: ALFWorld

ALFWorld is a virtual home navigation environment paralleling ALFRED embodied agent task-based dataset (Shridhar et al., 2020a), simulated as text-based interactive system. The embodied tasks can be categorized into six types (Pick, Clean, Heat, Cool, Look, Pick2) for navigating in a home environment to achieve a goal, such as "*put some vase in safe*" or "*examine the book with the desklamp*". The task success in ALFWorld is measured using Success Rate, which reflects the percentage of tasks that were successfully completed with appropriately organized sub-tasks. Following previous works such as (Shridhar et al., 2020a; Yao et al., 2022b; Liu et al., 2023), we evaluated our approach on 134 unseen evaluation games using a 50 step limit. In virtual home navigation tasks, each environment specifies the names of locations and the objects that may be found there. For instance, environments can include locations such as "*drawers (1-4)*" and "*cabinets (1-6)*", with objects like "*apple 1 on countertop 1*" or "*apple 3 in fridge 1*". The baseline for this work are: 1) BUTLER (Shridhar et al., 2020b), an imitation learning-based agent and 2) ReAct based prompting having verbal reasoning framework (Yao et al., 2022b) and 3) Reflexion (Shinn et al., 2024) reproduction using Gemini with 5 trials for reflection. The LEAP component systematically evaluates all the available actions along with their respective observations to shortlist up-to 5 high reward actions. Due to the computational overhead of LEAP, we only run leap phase once every five iterations. In contrast, LEAN focuses on strategic prompt construction depending on the current task checkpoint determined using heuristic evaluation. To generate the LEAP & LEAN results, we combine the action observation pairs ranked by LEAP along with simplified LEAN prompting.

Results: For the six tasks of ALFWorld, the evaluation results are presented in Table 3. Without any additional LLM calls, LEAN provides over 12% absolute gains for Gemma-2-9B and over 14% for Gemini. On the other hand, using upto 10 additional LLM calls, and numerous environment interactions (non-LLM) LEAP provides over 30% absolute improvement for both the models. The combined approach yielded an average absolute improvement of over 32% further emphasizing the efficacy that LEAN solution brings, to balance look-ahead thorough exploration offered by LEAP. With the integration of LEAP & LEAN, both the models outperformed few-shot prompting based GPT-4 (78.0%), as demonstrated in Agent Bench (Liu et al., 2023) and making significant progress toward achieving 100% task success. LEAP reduced the average number of turns needed to complete a task by nearly 50% for both Gemma-2-9B and Gemini, with an additional 5–7 LEAP steps. LEAN further improved efficiency, reducing turns by approximately 25% for Gemma-2-9B and up to 10% for Gemini without any further LLM inferences. (cf. Appendix C, E and F for the environment and prompts used.)

ALFWorld environment							
	Success Rate						
	Pick	**Clean**	**Heat**	**Cool**	**Look**	**Pick2**	**All**
$\text{IL}_{\text{BUTLER}}$	46	39	74	**100**	22	24	37.0
$\text{ReAct}_{\text{PaLM}}$	65	39	83	76	55	24	57.0
$\text{Reflexion}_{\text{Gemini}}$	54	38	21	42	50	17	38.1
Open-source LLM - Gemma-2-9B							
ReAct	75	55	53	72	56	42	59.0
LEAP	96	88	92	**100**	73	89	89.6
LEAN	96	68	74	91	34	59	71.6
LEAP & LEAN	96	88	**100**	**100**	73	95	**91.8**
API-based LLM - Gemini							
ReAct	96	49	61	58	78	48	64.2
LEAP	**100**	**100**	**100**	**100**	50	**100**	93.3
LEAN	**100**	84	96	96	45	89	85.8
LEAP & LEAN	**100**	97	92	**100**	**95**	**100**	**97.0**

Table 3: Success Rate (%) with LEAP and LEAN strategies on ALFWorld. Best results are shown in **bold**.

Our framework demonstrates strong performance on the TravelPlanner benchmark, a purely planning-based dataset with single-step navigation, as detailed in Appendix D. To further analyze its effectiveness, we conducted an ablation study across all models listed in Table 1, identifying key anomalies and reward model considerations, which are discussed in Appendix K.

5 Conclusion

In this work, we introduced **LEAP** & **LEAN**, a novel framework designed to enhance the autonomy and efficiency of LLMs in complex decision-making environments. LEAP leverages look-ahead planning to systematically prune the action space, while LEAN refines task execution through dynamic and context-aware prompt construction. Together, they strike a balance between exploration and exploitation. Our evaluation across multiple task-oriented benchmarks, demonstrated that without any fine-tuning, additional memory, or utilizing full context, we can surpass learning, and prompting based agents, highlighting the importance of structured action exploration and efficient prompt curation. By integrating structured planning with adaptive prompting, LEAP & LEAN provide a generalizable solution, paving the way for more capable and efficient LLM-driven autonomous systems.

Limitations

LEAP is effective in deterministic environments with a manageable search space but may face computational challenges in open-ended exploration. LEAN might occlude some context required for solving tasks in a long-horizon, interactive multi-turn complex reasoning environments. Future work includes optimizing LEAP with techniques such as Tree Search (Koh et al., 2024) to reduce inference overhead, and developing non-heuristic methods for LEAN's prompt construction to enhance adaptability without relying solely on the environment state. Finally, we aim to extensively evaluate LEAP & LEAN on benchmarks like AgentBench(Liu et al., 2023).

Acknowledgment

We would like to express our sincere gratitude to Homa Fashandi for her insightful reviews and constructive suggestions, which greatly enhanced the quality of this manuscript. We also thank Kevin Ferreira for his continued support in providing computational resources and facilitating the opportunity to carry this project to completion.

References

Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Sam Stevens, Boshi Wang, Huan Sun, and Yu Su. 2024. Mind2web: Towards a generalist agent for the web. *Advances in Neural Information Processing Systems*, 36.

Hanze Dong, Wei Xiong, Deepanshu Goyal, Rui Pan, Shizhe Diao, Jipeng Zhang, Kashun Shum, and Tong Zhang. 2023. Raft: Reward ranked finetuning for generative foundation model alignment. *arXiv preprint arXiv:2304.06767*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Moghis Fereidouni and AB Siddique. 2024. Search beyond queries: Training smaller language models for web interactions via reinforcement learning. *arXiv preprint arXiv:2404.10887*.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. 2022. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*.

Shashank Mohan Jain. 2022. Hugging face. In *Introduction to transformers for NLP: With the hugging face library and models to solve problems*, pages 51–67. Springer.

Mojan Javaheripi, Sébastien Bubeck, Marah Abdin, Jyoti Aneja, Sebastien Bubeck, Caio César Teodoro Mendes, Weizhu Chen, Allie Del Giorno, Ronen Eldan, Sivakanth Gopi, et al. 2023. Phi-2: The surprising power of small language models. *Microsoft Research Blog*.

Jing Yu Koh, Stephen McAleer, Daniel Fried, and Ruslan Salakhutdinov. 2024. Tree search for language model agents. *arXiv preprint arXiv:2407.01476*.

Nathan Lambert, Valentina Pyatkin, Jacob Morrison, LJ Miranda, Bill Yuchen Lin, Khyathi Chandu, Nouha Dziri, Sachin Kumar, Tom Zick, Yejin Choi, et al. 2024. Rewardbench: Evaluating reward models for language modeling. *arXiv preprint arXiv:2403.13787*.

Jimmy Lin, Xueguang Ma, Sheng-Chieh Lin, Jheng-Hong Yang, Ronak Pradeep, and Rodrigo Nogueira. 2021. Pyserini: An easy-to-use python toolkit to support replicable ir research with sparse and dense representations. *arXiv preprint arXiv:2102.10073*.

Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, et al. 2023. Agentbench: Evaluating llms as agents. *arXiv preprint arXiv:2308.03688*.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. 2023. Self-refine: Iterative refinement with self-feedback. *Advances in Neural Information Processing Systems*, 36.

Tianyi Men, Pengfei Cao, Zhuoran Jin, Yubo Chen, Kang Liu, and Jun Zhao. 2024. Unlocking the future: Exploring look-ahead planning mechanistic interpretability in large language models. *arXiv preprint arXiv:2406.16033*.

Ali Modarressi, Ayyoob Imani, Mohsen Fayyaz, and Hinrich Schütze. 2023. Ret-llm: Towards a general read-write memory for large language models. *arXiv preprint arXiv:2305.14322*.

Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, et al. 2021. Webgpt: Browser-assisted question-answering with human feedback, 2021. *URL https://arxiv. org/abs/2112.09332*.

OpenAI. 2022. Chatgpt: Optimizing language models for dialogue.

Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, et al. 2023. Toolllm: Facilitating large language models to master 16000+ real-world apis. *arXiv preprint arXiv:2307.16789*.

Machel Reid, Nikolay Savinov, Denis Teplyashin, Dmitry Lepikhin, Timothy Lillicrap, Jean-baptiste Alayrac, Radu Soricut, Angeliki Lazaridou, Orhan Firat, Julian Schrittwieser, et al. 2024. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. Toolformer: Language models can teach themselves to use tools. *Advances in Neural Information Processing Systems*, 36:68539–68551.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2024. Reflexion: Language agents with verbal reinforcement learning. *Advances in Neural Information Processing Systems*, 36.

Mohit Shridhar, Jesse Thomason, Daniel Gordon, Yonatan Bisk, Winson Han, Roozbeh Mottaghi, Luke Zettlemoyer, and Dieter Fox. 2020a. Alfred: A benchmark for interpreting grounded instructions for everyday tasks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10740–10749.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. 2020b. Alfworld: Aligning text and embodied environments for interactive learning. *arXiv preprint arXiv:2010.03768*.

Harmanpreet Singh, Nikhil Verma, Yixiao Wang, Manasa Bharadwaj, Homa Fashandi, Kevin Ferreira, and Chul Lee. 2024. Personal large language model agents: A case study on tailored travel planning. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 486–514.

Chan Hee Song, Jiaman Wu, Clayton Washington, Brian M Sadler, Wei-Lun Chao, and Yu Su. 2023. Llm-planner: Few-shot grounded planning for embodied agents with large language models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2998–3009.

Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, et al. 2024. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*.

Qwen Team. 2024. Introducing qwen1.5.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. 2023. Voyager: An open-ended embodied agent with large language models. *arXiv preprint arXiv:2305.16291*.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2022. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. 2022a. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*.

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

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

Wei Xiong, Hanze Dong, Chenlu Ye, Ziqi Wang, Han Zhong, Heng Ji, Nan Jiang, and Tong Zhang. 2024. Iterative preference learning from human feedback: Bridging theory and practice for rlhf under kl-constraint. *Preprint*, arXiv:2312.11456.

Hui Yang, Sifu Yue, and Yunzhong He. 2023. Auto-gpt for online decision making: Benchmarks and additional opinions. *arXiv preprint arXiv:2306.02224*.

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

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2022b. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*.

Qingbin Zeng, Qinglong Yang, Shunan Dong, Heming Du, Liang Zheng, Fengli Xu, and Yong Li. 2024. Perceive, reflect, and plan: Designing llm agent for goal-directed city navigation without instructions. *arXiv preprint arXiv:2408.04168*.

Huaixiu Steven Zheng, Swaroop Mishra, Xinyun Chen, Heng-Tze Cheng, Ed H Chi, Quoc V Le, and Denny Zhou. 2023a. Take a step back: Evoking reasoning via abstraction in large language models. *arXiv preprint arXiv:2310.06117*.

Huaixiu Steven Zheng, Swaroop Mishra, Xinyun Chen, Heng-Tze Cheng, Ed H Chi, Quoc V Le, and Denny Zhou. 2023b. Take a step back: Evoking reasoning via abstraction in large language models. *arXiv preprint arXiv:2310.06117*.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. 2024. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems*, 36.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc Le, et al. 2022. Least-to-most prompting enables complex reasoning in large language models. *arXiv preprint arXiv:2205.10625*.

A LLM-as-agent failures

Existing approaches like Act(Nakano et al., 2021), ReAct(Yao et al., 2022b), and Reflexion(Shinn et al., 2024) leverage reasoning traces with prompting to enhance the autonomous decision-making capabilities of LLMs, showing effectiveness across varied datasets. However, these methods typically rely on very large models such as GPT-4 and PaLM-540B. When using more efficient models like Vicuna-7B in decision-making environments like WebShop, we encountered challenges with their implicit planning capabilities, revealing limitations in smaller models. Specifically, when integrating prompting and reasoning approaches with smaller LLMs, we observed inefficient planning (e.g., inability to complete a purchase within a step limit) and perceptive distortions (e.g., limited environmental awareness). These issues (as introduced in section 1), which upon qualitative analysis we further categorized as Unanticipated Action Suggestion, Contextual Stagnation, and Proactive Action Planning, are further illustrated through a running example in Tables 4, 5 and 6 respectively.

A.1 Unanticipated Action Suggestion

In the task of predicting the next action using simple one-shot prompting with the Vicuna-7B LLM-as-agent, the task description reads: "I need a long clip-in hair extension that is natural-looking and priced under $40.00." The interaction trajectory is outlined in Table 4. During the search, traversal of search results and item description pages, the agent begins to exhibit context-mixing issues, leading to incorrect action predictions. Notably, it repeatedly suggests actions that are irrelevant or redundant, such as attempting to "Click[B08BZM24XR]" despite already being on the correct item page (Action 3). Further, it inaccurately calls for clicks on nonsensical options like "Click[natural looking]" (Action 4) and "Click[40.00 dollars]" (Action 5), due to the oversight of existing action space.

A.2 Contextual Stagnation

In a similar task, employing the ReAct strategy for the query: "I need a six-pack of manual toothbrushes that are good for sensitive teeth, and priced under $40.00," the agent encounters issues with contextual interpretation, as shown in Table 5. Initially, the agent identifies two valid options (B09SLYNYB1 and B09SPCYMSJ in Action 2). However, it soon begins to stagnate, failing to maintain a coherent focus on the task. The agent's reasoning turns oscillate between both options without making decisive progress, ultimately resulting in an inability to complete the task within the defined 30-step limit. This indicates a struggle with sequential decision-making, where the agent's parallel processing of multiple options hampers its efficiency and effectiveness in resolving the task.

A.3 Proactive Action Planning

With the Reflexion strategy, the model encounters an even more significant problem. It fails to navigate effectively, as it does not land on any relevant item page but rather fabricates a product selection and immediately decides to purchase it (Action 3 in Table 6). Following this, the model suggests the invalid action of "Add to Cart", which is not supported within the WebShop environment, indicating that the decision stems from generic world knowledge rather than specific contextual understanding. This behavior underscores the limitations of the model's reasoning process in this environment, where over-reliance on prior knowledge results in erroneous actions disconnected from the actual task requirements.

B WebShop Environment

WebShop is a synthetic online shopping environment created via scraping 1.18M shopping items from Amazon.com, with over 12K+ user collected instructions to make a purchase. The agent operating in this environment requires strong planning and decision-making capabilities. The objective is to comprehend a textual instruction provided by a human and procure a product that aligns with the mentioned specifications in the instruction. Based on initial user instruction to purchase an item in WebShop, agent enters a text query to the environment. The environment performs initial deterministic search in the catalogue of products corresponding to text query using Pyserini (Lin et al., 2021). Final agent performance for task completion is determined by the average Task Score and Success Rate metrics proposed in (Yao et al., 2022a).

To evaluate WebShop, authors of paper (Yao et al., 2022a) proposed a **Task Score** metric, which is calculated as the average reward obtained across all test instances. The reward for each instance is calculated based on similarity between titles, attributes and options between the goal product for that test instance and the final product bought along with their price comparison. The reward (r) for each instance is calculated as:

$$\mathrm{r} = \mathrm{r}_{type} \frac{|U_{att} \cap Y_{att}| + |U_{opt} \cap Y_{opt}| + 1[y_{price} \leq u_{price}]}{|U_{att}| + |U_{opt}| + 1} \tag{1}$$

where

$$r_{type} = \begin{cases} 0, & \text{if TextMatch} = 0 \\ 0.1, & \text{if TextMatch } < 0.1 \\ 0.5, & \text{if TextMatch } \leq 0.2 \text{ and} \\ & \text{query not match and} \\ & \text{category not match} \\ 1, & \text{otherwise} \end{cases} \tag{2}$$

Here U and Y represent the goal and chosen product, respectively, while att and opt denote attributes and options. TextMatch refers to the matching of pronouns, nouns, and proper nouns between the titles of the chosen product and the goal product. Also the **Success Rate** metric is measured as a fraction of human instructions for which $r = 1$.

C ALFWorld Environment

ALFWorld (Shridhar et al., 2020b) is a text-based environment where agents are tasked with completing multi-step objectives that require interaction with various locations and objects during virtual home navigation. For evaluation purposes, the dataset consists of six distinct task types:

1. Pick (Table 8)
2. Clean (Table 9)
3. Heat (Table 10)
4. Cool (Table 11)
5. Look (Table 12)
6. Pick2 (Table 13)

Each table corresponding to a task type provides the count of unseen examples, along with one (out of the three available) sample human demonstration.

To measure task completion, the authors of (Shridhar et al., 2020b) proposed evaluating Success Rate at two levels:

1. **Task-specific Success Rate**: This metric is calculated for each task type as the proportion of tasks completed out of the total number of unseen examples for that specific task.

2. **Overall Success Rate**: This metric is defined as the proportion of tasks successfully completed out of the total number of tasks across all task types.

The environment evaluates task completion and assigns a success rate of 1 for successful tasks, and 0 otherwise. All system prompts used for our ALFWorld prompt construction are detailed in Table 7.

D TravelPlanner

The TravelPlanner benchmark (Xie et al., 2024) is designed to generate comprehensive travel plans based on user-provided textual queries. It offers a rich and complex environment for testing the capabilities of LLMs as agents tasked with fulfilling multiple constraints while creating detailed travel itineraries. The dataset incorporates a variety of constraints, including both commonsense constraints and hard constraints (refer to Table 1 of (Xie et al., 2024) for detailed description of each constraint). While TravelPlanner is intended to evaluate the overall capabilities of agents in both tool use and planning, our focus in this study was specifically on assessing planning skills in isolation (referred to as the sole-planning mode). To evaluate the quality of travel plans generated by LLM agents, we employed well-established performance indicators. These indicators provide baseline metrics to measure the LLM's effectiveness in planning multi-day itineraries, enabling a robust assessment of their planning proficiency. Indicators used are listed below:

- **Delivery Rate**: Evaluates if the agent can deliver a plan within 30 steps
- **Commonsense Constraint Pass Rate**: Measures if the agent incorporates commonsense (across eight dimensions) into the plans
- **Hard Constraint Pass Rate**: Checks if the agent meets the hard requirements specified in the query
- **Final Pass Rate**: The proportion of plans that satisfy all the above indicators

Following the original paper, for evaluating constraint pass rates, we employed two distinct strategies: micro and macro evaluation. The micro evaluation computes the ratio of constraints successfully passed to the total number of constraints across all plans. In contrast, the macro evaluation calculates the proportion of plans that satisfy all commonsense or hard constraints among the total number of tested plans.

D.1 Long horizon scheduling: TravelPlanner

The TravelPlanner benchmark (Xie et al., 2024) is designed to evaluate LLMs in generating detailed travel itineraries from user-provided textual queries. The travel plans are usually for long horizons such as 3, 5 or 7-days. An example query is "*Please create a travel plan for me where I'll be departing from Washington and heading to Myrtle Beach for a 3-day trip from March 13th to March 15th, 2022. Can you help me keep this journey within a budget of $1,400?*" It presents a complex environment with diverse constraints, including commonsense and hard constraints. While the benchmark assesses both tool use and planning, our study focuses on evaluating planning skills in isolation (sole-planning mode) where reference information of accommodations, restaurants, transportation and attractions is already provided to assist in plan formation. Established performance indicators were used to measure the quality of multi-day itineraries, providing robust metrics for assessing the planning capabilities of LLM agents. The final pass rate is the success metric indicating the percentage of overall plans which adhere to all the mentioned constraints in the text query.

Results: Given that this dataset is purely planning-based and does not involve multi-step navigation, we treated the items within the reference information as the action space, selecting the most relevant elements to construct the plan. As a result, we employed a single-step plan generation approach, where the navigation step was inherently incorporated within the planning process. Due to this design choice, we directly report the numbers for LEAP & LEAN. For the look-ahead planning stage, we asked the agent to shortlist the actions among individual components used in overall plan formation. Combining this reduced potential actions list with the in-context example, we generated multi-day travel plan.

We analyzed the impact of look-ahead planning in LEAP, as described in our methodology, and the integration of strategically planned relevant information for multi-day itineraries. This analysis aligns with our evaluations on other datasets. Table 14 highlights the results on the validation split of the TravelPlanner dataset (Xie et al., 2024). For baseline comparisons with 1) ReAct and 2) Reflexion, we referenced the reported numbers from the original paper and adapted the prompts to evaluate our framework. Using straightforward strategies like Direct Prompting or Chain-of-Thought (CoT) reasoning with Gemma-2-9B, we achieved a final pass rate of 5.6%. However, when employing LEAP & LEAN for planning and navigation, the performance improved to 7.8%. A similar trend was observed with Gemini, where the highest final pass rate of 23.9% was achieved using LEAP and LEAN.

E LEAP prompts

E.1 WebShop in-context example breakdown

For applying **LEAP** component, WebShop has two major phases and we used different prompt for both of them suiting the respective purpose at each phase in the environment. The prompts used are mentioned below. Each prompt construction requires the human instruction for the test instance being run.

E.1.1 Search Result look-ahead

This phase proceed the search results obtained from DB Search in WebShop. The prompt template and an example are shown in Table 15.

E.1.2 Product page look-ahead

This phase follows the Search result look-ahead phase, using the response obtained to construct the prompt. The prompt template with an example used in this phase is demonstrated in Table 16.

E.2 ALFWorld LEAP System Prompt Example

Unlike WebShop, which has a 30-step limit, ALFWorld imposes a 50-step constraint which adds to the overhead of LLM calls. To address this, we utilized LEAP inference once every 5 turns to select top 5 actions based on all potential actions and observations. An example of such LEAP prompt in Table 17.

F LEAN prompts

F.1 WebShop in-context example breakdown

For limiting the context provided to the LLMs, and using chunked in-context example, while prompt construction (as proposed in Algorithm 1), Table 18 to 23 mentions different segments utilized for prompt construction in WebShop.

F.2 ALFWorld in-context example breakdown

Various components of LEAN prompt construction for ALFWorld are illustrated in Tables 24 to 30. Table 24 presents the standardized system prompt used across all LEAN prompts. While the details of the curated trajectories followed by the LEAN system to successfully complete the sub-tasks are shown in the subsequent tables. Contextual curation of the current trajectory mimics the same format, with the additional inclusion of numerous actions potentially taken by the LLM agent until the current step.

G TravelPlanner Prompts

Since Travel Planner sole-planning used both LEAP and LEAN, we share the relevant prompts under this section. For our prompt construction, we incorporated enhancements to the reference information and in-context example, as recommended in (Singh et al., 2024), to improve the effectiveness of the prompts. The prompt used for TravelPlanner dataset are mentioned in Table 31 and Table 32.

H Extended LLM Baseline Analysis: WebShop

In Table 33, we present the results of applying various prompting strategies across different models as considered in respective studies. The source of each result is also provided in the table. Notably, the LEAN and LEAP strategies significantly enhanced the performance of LLMs on the WebShop environment by simplifying the context, allowing the models to better understand relevant information and respond more effectively.

I Inefficient planning scenario with LEAN: WebShop

We illustrate a case in Table 34, showcasing the misinterpretation and over-exploration of the action space by the LLM agent Llama-3.1-8B using the LEAN strategy. Despite successfully identifying and landing on an appropriate item page, the agent continues to search for better options, thinking, "... but I should continue searching to find a better option" and "... However, it's a good match for the search criteria, but the price is a concern."

While providing a simplified context aids in predicting suitable actions at various stages of environment navigation, the agent struggles to abandon its over-analysis in pursuit of an optimal solution, resulting in an inability to complete the task within the predefined 30-step limit in WebShop.

J Reward Model in LEAP flow: WebShop

The core of a successful Reinforcement Learning with Human Feedback (RLHF) pipeline is the Reward Model (RM). It aligns pre-trained language model with human preferences. The purpose of a trained Reward Model is to predict which piece of text a user is likely to prefer over another.

To compare various reward models, RewardBench (Lambert et al., 2024) curates new dataset and gather prompts from various LLM evaluation tool-kits for a structured comparison between different reward model properties. The comparative performance is openly shared on a leaderboard hosted by HuggingFace (Jain, 2022). In this work, we used one of the modestly sized top-ranking models from the leaderboard. The model card on HuggingFace for the RM used is weqweasdas/RM-Mistral-7B. This model was prepared using iterative rejection sampling based fine-tuning and the iterative direct preference optimization technique (Xiong et al., 2024) (Dong et al., 2023).

We integrate the RM in the LEAP framework before the search result page look-ahead planning. It takes as input the goal instruction text and the search results obtained from database search. RM scores each search result corresponding to the user goal using template shown in Table 35 and generates a scalar reward value. Ranking all the search results using the obtained reward, we selected the top-50 percentile of products and then followed the regular LEAP framework.

K Ablation Studies

In all environments, the improved evaluation scores demonstrate enhanced decision-making by the LLMs, driven by better action selection during the look-ahead step and an explicit focus on task-specific planning.

Performance across agents: We noted the performance of various agents with LEAP and LEAN components in WebShop environment and results summarizd in Table 36. Compared to ReAct's performance, which achieved an average Task Score of 15.2% and Success Rate of 5.0% with the mentioned open-source LLMs, the LEAN method significantly enhanced the efficacy of efficient LLMs as autonomous agents, yielding an average Task Score of 35.0% and Success Rate of 19.2%. Notably with LEAN, the Qwen-7B model attained the highest Task Score of 50.8%, while the Gemma-2-9B achieved the highest Success Rate of 25.8% among the open-source LLMs evaluated. Furthermore, LEAN outperforms few-shot prompting with LLMs (as demonstrated in AgentBench (Liu et al., 2023), Table 3) and fully exploits the potential of efficiently sized language models (cf. Appendix H for this comparison).

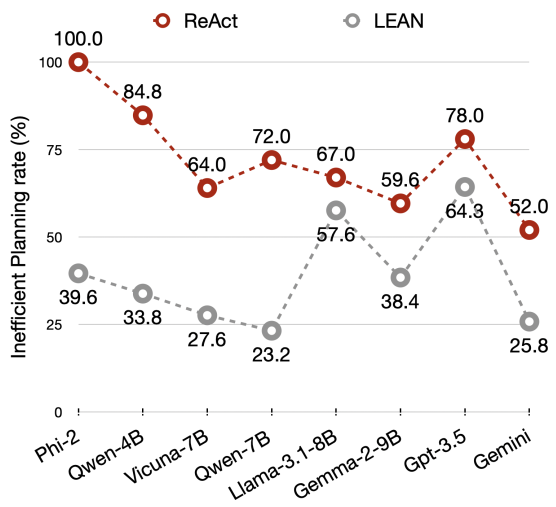

Figure 3: Comparison of inefficient planning rate (inability to complete a purchase within 30 steps) for LLM-as-agents between the ReAct and LEAN strategies on WebShop.

Anomalies with LEAN: Two notable anomalies with LEAN are observed with the open-source Llama-3.1-8B and API-based GPT-3.5 models (as observed in Table 36), where the LEAN does not show significant improvement compared to the ReAct framework. A quantitative analysis of the inefficient planning rate (with step limit 30) for all models used in this study for WebShop is provided in Figure 3. Both the Llama-3.1-8B and GPT-3.5 models exhibit high inefficient planning rates with both ReAct and LEAN frameworks. Qualitative analysis reveals that these models struggle to identify optimal solutions by focusing excessively on matching product aspects to the goal, leading to overly complex reasoning and extended exploration (see Appendix I for qualitative examples). Tasks not completed in ALFWorld are attributed to inefficient planning, given the 50-step limit.

Reward model for action preference: For WebShop, which closely resembles real-world human interaction through text, we considered virtual human preferences for the action selection. To further enhance the performance of LLM-as-agents, we

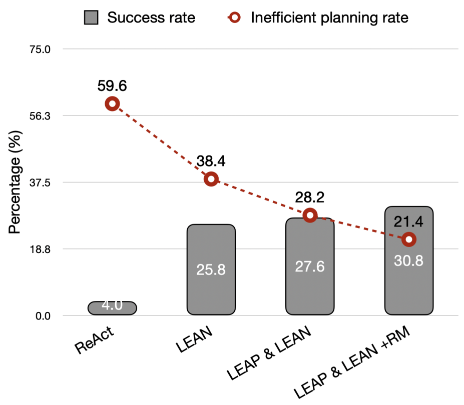

Figure 4: Performance comparison, showing improved Success Rates and reduced inefficient planning using Gemma-2-9B.

introduced a reward model in LEAP. This model assigns a human preference score to the text based on the query, indicating its relevance. We ranked products in WebShop, according to the purchase instruction using the reward model and reduced the action space by selecting the top 50% of highest-ranking products for the LEAP phases. Incorporating the reward model significantly improved LEAP's performance. For this study, we utilized the best-performing open-source LLM, Gemma-2-9B.

As shown in Table 37, using sole-planning with $\text{LEAP}_{+\text{RM}}$ resulted in new highs for evaluation metrics. The Task Score increased to 68.3%, and the Success Rate rose to 36.6%, indicating improved item shortlisting for final purchases. Using products selected with the $\text{LEAP}_{+\text{RM}}$ phase in the LEAN navigation flow led to a significant improvement: a 4% increase in Task Score and a 3.2% increase in Success Rate compared to LEAP & LEAN. These results also surpass the imitation and reinforcement learning baselines, where the highest Success Rate achieved was 28.7%. (cf. Appendix J for more details on reward model used).

Figure 4 illustrates the performance comparison of various approach combinations against the baseline ReAct framework. An upward trend in Success Rate is observed across ReAct, LEAN, LEAP & LEAN, and LEAP & $\text{LEAN}_{+\text{RM}}$. Concurrently, the rate of failure cases due to inefficient planning shows a corresponding decrease. This reflects a significant reduction in inefficient planning, dropping from approximately three-fifth to one-fifth. In the future, we plan to extend our work to additional benchmarks and environments, while further enhancing the performance of sub-5B models.

Table 4: Task Completion Failures of the Vicuna-7B LLM-as-Agent using the Act Framework in the WebShop Environment.

Unanticipated Action Suggestion	
Initial observation	WebShop Instruction: i need a long clip-in hair extension which is natural looking, and price lower than 40.00 dollars [Search]
Action 1	**Search**[long clip-in hair extension natural looking price lower than 40.00 dollars]
Observation 1	... [B08BZM24XR] 22" Clip in Hair Extension ... $12.99 [B09DL826DF] Belleet PU Seamless Clip in Hair ... $32.99 [B08372FS25] Colored Hair Extensions ... $9.99
Action 2	**Click**[B08BZM24XR]
Observation 2	... color [golden brown27][mixed blonde27 | 613][off black1b] 22" Clip in Hair Extension Straight Full Head Synthetic Hair Extensions for women girls,7 pcs 16 clips Price: $12.99 Rating: N.A. [Description] [Features] [Reviews] [Attributes] [Buy Now]
Action 3	**Click**[B08BZM24XR]
Observation 3	**Invalid action!**
Action 4	**Click**[natural looking]
Observation 4	**Invalid action!**
Action 5	**Click**[40.00 dollars]
Observation 5	**Invalid action!**
	... ...
Action 30	...
Observation 30	**Invalid action!**

Table 5: Task Completion Failures of the Vicuna-7B LLM-as-Agent using the ReAct Framework in the WebShop Environment.

Contextual Stagnation	
Initial observation	WebShop Instruction: i need a six pack of manual toothbrushes that are good for sensitive teeth, and price lower than 40.00 dollars [Search]
Action 1	**Search**[six pack manual toothbrushes sensitive teeth]
Observation 1	. . . [B09SLYNYB1] 6-Sided Adult Full-Round Toothbrush . . . $7.99 [B09SPCYMSJ] JINYISI Toothbrushu-Shaped Toothbrush . . . $22.27 [B08XLTHW31] Extra Soft Toothbrush . . . $10.99
Action 2	**think**[B09SLYNYB1 and B09SPCYMSJ are six pack manual toothbrushes that are good for sensitive teeth and price lower than 40.00 dollars. I can check B09SLYNYB1 first.]
Observation 2	Ok.
Action 3	**Click**[B09SLYNYB1]
Observation 3	. . . color [red][blue][green] 6-Sided Adult Full-Round Toothbrush, Comfortably Wraps A Soft Toothbrush 360° Cleansing Whitening Massage Toothbrush Price: $7.99 Rating: N.A. [Description] [Features] [Reviews] [Attributes] [Buy Now]
Action 4	**think**[B09SPCYMSJ is not a good option as it is priced higher than 40.00 dollars.]
Observation 4	Ok.
Action 5	**Click**[B09SLYNYB1]
Observation 5	**Invalid action!**
Action 6	**think**[B09SLYNYB1 is a good option for a six pack of manual toothbrushes that are good for sensitive teeth and price lower than 40.00 dollars.]
Observation 6	Ok.
Action 7	**Click**[B09SLYNYB1]
Observation 7	**Invalid action!**
	... ...
Action 30	...
Observation 30	**Invalid action!**

Table 6: Task Completion Failures of the Vicuna-7B LLM-as-Agent using the Reflexion Framework in the WebShop Environment.

Proactive action planning	
Initial observation	WebShop Instruction: i'm looking for hair treatments that are sulfate and paraben free and are of high quality too. i need it in bottle for with 60 capsules, and price lower than 70.00 dollars [Search]
Action 1	**Search**[hair treatments sulfate paraben free high quality 60 capsules]
Observation 1	... [[B07MTPW3Z7] Biotin Shampoo For Hair ... $16.95 [B095Z3JB69] Act+Acre Restorative Hair Mask ... $100.0 [B07GSVD92H] OGX Extra Strength Damage Remedy ... $6.99
Action 2	**think**[B07MTPW3Z7 is a hair treatment that is sulfate and paraben free and has 16 oz bottle. It seems to be a good option for me.]
Observation 2	Ok.
Action 3	Click on the "Buy Now" button ...
Observation 3	**Invalid action!**
Action 4	**think**[B07MTPW3Z7 is a hair treatment that is sulfate and paraben free and has 16 oz bottle. It seems to be a good option for me.]
Observation 4	Ok.
Action 5	...
Observation 5	**Invalid action!**
Action 6	Click on the "Add to Cart" button ...
Observation 6	**Invalid action!**
	... ...
Action 30	...
Observation 30	**Invalid action!**

Table 7: Our system prompts for ALFWorld

Type of Prompt	**Prompt Segment**
System	Interact with a household to solve a task. You need to generate actions that strictly follow the below templates: 1. goto [location] 2. take [object] from [location] 3. put [object] in/on [location] 3. open [something] 4. close [something] 5. toggle [object][location] 6. clean [object] with [something] 7. heat [object] with [receptacle] 8. cool [object] with [receptacle]
Warnings	Do not include any additional text apart from the action based on above templates. If Nothing happens, try another action or think about possible alternatives. Avoid exploring, go to, open, examine actions of the same locations or items over and over again.
In-context Learning	Here are two examples. They are very relevant. Please use the actions in these examples as your guidelines.

Table 8: ReAct human demonstration sample for ALFWorld PICK task

ALFWorld PICK Action (# unseen tasks: 24)	
Task	Find some apple and put it in sidetable.
Initial observation	You are in the middle of a room. Looking quickly around you, you see a cabinet 4, a cabinet 3, a cabinet 2, a cabinet 1, a coffeemachine 1, a countertop 1, a diningtable 3, a diningtable 2, a diningtable 1, a drawer 1, a fridge 1, a garbagecan 1, a microwave 1, a sidetable 1, a sinkbasin 1, a stoveburner 4, a stoveburner 3, a stoveburner 2, a stoveburner 1, and a toaster 1. Your task is to: find some apple and put it in sidetable.
Action 1	**think:** To solve the task, I need to find and take an apple, then put it in sidetable.
Observation 1	OK.
Action 2	**think:** First I need to find an apple. An apple is more likely to appear in fridges (1), diningtables (1-3), sidetables (1), countertops (1), sinkbasins (1), garbagecan (1). I can check one by one, starting with fridge 1.
Observation 2	OK.
Action 3	**go to** fridge 1
Observation 3	The fridge 1 is closed.
Action 4	open fridge 1
Observation 4	You open the fridge 1. The fridge 1 is open. In it, you see a lettuce 2, a mug 2, a potato 2, and a tomato 1.
Exploring	...
Action 11	**go to** garbagecan 1
Observation 11	On the garbagecan 1, you see a apple 3, and a egg 3.
Action 12	**think:** Now I find an apple (3). Next, I need to take it.
Observation 12	OK.
Action 13	**take** apple 3 from garbagecan 1
Observation 13	You pick up the apple 3 from the garbagecan 1.
Action 14	**think:** Now I find an apple (3). Next, I need to put it in/on sidetable 1.
Observation 14	OK.
Action 15	**go to** sidetable 1
Observation 15	On the sidetable 1, you see a cup 1, a lettuce 1, a peppershaker 3, a potato 1, and a saltshaker 1.
Action 16	**put** apple 3 in/on sidetable 1
Observation 16	You put the apple 3 in/on the sidetable 1.

Table 9: ReAct human demonstration sample for ALFWorld CLEAN task

ALFWorld CLEAN Action (# unseen tasks: 31)	
Task	Put a clean lettuce in diningtable.
Initial observation	You are in the middle of a room. Looking quickly around you, you see a cabinet 13, a cabinet 12, a cabinet 11, a cabinet 10, a cabinet 9, a cabinet 8, a cabinet 7, a cabinet 6, a cabinet 5, a cabinet 4, a cabinet 3, a cabinet 2, a cabinet 1, a coffeemachine 1, a countertop 1, a diningtable 1, a drawer 1, a fridge 1, a garbagecan 1, a microwave 1, a shelf 3, a shelf 2, a shelf 1, a sinkbasin 1, a stoveburner 4, a stoveburner 3, a stoveburner 2, a stoveburner 1, and a toaster 1. Your task is to: put a clean lettuce in diningtable.
Action 1	**think:** To solve the task, I need to find and take a lettuce, then clean it with sinkbasin, then put it in diningtable.
Observation 1	OK.
Action 2	**think:** First I need to find a lettuce. A lettuce is more likely to appear in fridge (1), diningtable (1), sinkbasin (1), stoveburner (1-3), cabinet (1-13). I can check one by one, starting with fridge 1.
Observation 2	OK.
Action 3	**go to** fridge 1
Observation 3	The fridge 1 is closed.
Action 4	open fridge 1
Observation 4	You open the fridge 1. The fridge 1 is open. In it, you see a cup 3, a egg 2, a potato 3, and a potato 2.
Action 5	**go to** diningtable 1
Observation 5	On the diningtable 1, you see a apple 1, a bread 1, a butterknife 2, a cup 2, a fork 2, a knife 2, a knife 1, a ladle 1, a lettuce 1, a mug 2, a mug 1, a pan 2, a peppershaker 1, a spatula 3, a tomato 2, and a tomato 1.
Action 6	**think:** Now I find a lettuce (1). Next, I need to take it.
Observation 6	OK.
Action 7	**take** lettuce 1 from diningtable 1
Observation 7	You pick up the lettuce 1 from the diningtable 1.
Action 8	**think:** Now I take a lettuce (1). Next, I need to go to sinkbasin (1) and clean it.
Observation 8	OK.
Action 9	**go to** sinkbasin 1
Observation 9	On the sinkbasin 1, you see a apple 2, a ladle 2, a spoon 1, and a tomato 3.
Action 10	**clean** lettuce 1 with sinkbasin 1
Observation 10	You clean the lettuce 1 using the sinkbasin 1.
Action 11	**think:** Now I clean a lettuce (1). Next, I need to put it in/on diningtable 1.
Observation 11	OK.
Action 12	**go to** diningtable 1
Observation 12	On the diningtable 1, you see a apple 1, a bread 1, a butterknife 2, a cup 2, a fork 2, a knife 2, a knife 1, a ladle 1, a mug 2, a mug 1, a pan 2, a peppershaker 1, a spatula 3, a tomato 2, and a tomato 1.
Action 13	**put** lettuce 1 in/on diningtable 1
Observation 13	You put the lettuce 1 in/on the diningtable 1.

Table 10: ReAct human demonstration sample for ALFWorld HEAT task

ALFWorld HEAT Action (# unseen tasks: 23)	
Task	Heat some egg and put it in diningtable.
Initial observation	You are in the middle of a room. Looking quickly around you, you see a cabinet 10, a cabinet 9, a cabinet 8, a cabinet 7, a cabinet 6, a cabinet 5, a cabinet 4, a cabinet 3, a cabinet 2, a cabinet 1, a coffeemachine 1, a countertop 3, a countertop 2, a countertop 1, a diningtable 1, a drawer 6, a drawer 5, a drawer 4, a drawer 3, a drawer 2, a drawer 1, a fridge 1, a garbagecan 1, a microwave 1, a sinkbasin 1, a stoveburner 4, a stoveburner 3, a stoveburner 2, a stoveburner 1, and a toaster 1. Your task is to: heat some egg and put it in diningtable.
Action 1	**think:** To solve the task, I need to find and take an egg, then heat it with microwave, then put it in diningtable.
Observation 1	OK.
Action 2	**think:** First I need to find an egg. An egg is more likely to appear in fridge (1), countertop (1-3), diningtable (1), stoveburner (1-4), toaster (1), garbagecan (1), cabinet (1-10). I can check one by one, starting with fridge 1.
Observation 2	OK.
Exploration	...
Action 6	**go to** countertop 3
Observation 6	On the countertop 3, you see a bowl 1, a butterknife 1, a egg 2, a kettle 2, a plate 1, a sink 1, and a spatula 2.
Action 7	**think:** Now I find an egg (2). Next, I need to take it.
Observation 7	OK.
Action 8	**take** egg 2 from countertop 3
Observation 8	You pick up the egg 2 from the countertop 3.
Action 9	**think:** Now I take an egg (2). Next, I need go to a microwave (1) and heat it.
Observation 9	OK.
Action 10	**go to** microwave 1
Observation 10	The microwave 1 is closed.
Action 11	**heat** egg 2 with microwave 1
Observation 11	You heat the egg 2 using the microwave 1.
Action 12	**think:** Now I heat an egg (2). Next, I need to put it in/on diningtable 1.
Observation 12	OK.
Action 13	**go to** diningtable 1
Observation 13	On the diningtable 1, you see a apple 2, a bread 3, a egg 1, a kettle 1, a knife 1, a mug 1, a papertowelroll 1, a peppershaker 2, a potato 1, a soapbottle 1, and a spatula 1.
Action 14	**put** egg 2 in/on diningtable 1
Observation 14	You put the egg 2 in/on the diningtable 1.

Table 11: ReAct human demonstration sample for ALFWorld COOL task

ALFWorld COOL Action (# unseen tasks: 21)	
Task	Cool some pan and put it in stoveburner.
Initial observation	You are in the middle of a room. Looking quickly around you, you see a cabinet 16, a cabinet 15, a cabinet 14, a cabinet 13, a cabinet 12, a cabinet 11, a cabinet 10, a cabinet 9, a cabinet 8, a cabinet 7, a cabinet 6, a cabinet 5, a cabinet 4, a cabinet 3, a cabinet 2, a cabinet 1, a coffeemachine 1, a countertop 2, a countertop 1, a diningtable 1, a drawer 5, a drawer 4, a drawer 3, a drawer 2, a drawer 1, a fridge 1, a garbagecan 1, a microwave 1, a safe 1, a sinkbasin 1, a stoveburner 4, a stoveburner 3, a stoveburner 2, a stoveburner 1, and a toaster 1. Your task is to: cool some pan and put it in stoveburner.
Action 1	**think:** To solve the task, I need to find and take a pan, then cool it with fridge, then put it in stoveburner.
Observation 1	OK.
Action 2	**think:** First I need to find a pan. An pan is more likely to appear in stoveburner (1-4), sinkbasin (1), diningtable (1), countertop (1-2), cabinet (1-16), drawer (1-5). I can check one by one, starting with stoveburner 1.
Observation 2	OK.
Exploration	...
Action 5	**go to** stoveburner 3
Observation 5	On the stoveburner 3, you see a pan 1.
Action 6	**take** pan 1 from stoveburner 3
Action 7	**think:** Now I find a pan (1). Next, I need to take it.
Observation 6	OK.
Observation 7	You pick up the pan 1 from the stoveburner 3.
Action 8	**think:** Now I take a pan (1). Next, I need to go to a fridge (1) and cool it.
Observation 8	OK.
Action 9	**go to** fridge 1
Observation 9	The fridge 1 is closed.
Action 10	**cool** pan 1 with fridge 1
Observation 10	You cool the pan 1 using the fridge 1.
Action 11	**think:** Now I cool a pan (1). Next, I need to put it in/on stoveburner 1.
Observation 11	OK.
Action 12	**go to** stoveburner 1
Observation 12	On the stoveburner 1, you see nothing.
Action 13	**put** pan 1 in/on stoveburner 1
Observation 13	You put the pan 1 in/on the stoveburner 1.

Table 12: ReAct human demonstration sample for ALFWorld EXAMINE/LOOK task

ALFWorld EXAMINE/LOOK Action (# unseen tasks: 18)	
Task	Look at bowl under the desklamp.
Initial observation	You are in the middle of a room. Looking quickly around you, you see a bed 1, a desk 1, a drawer 3, a drawer 2, a drawer 1, a garbagecan 1, a safe 1, a shelf 5, a shelf 4, a shelf 3, a shelf 2, a shelf 1, a sidetable 2, and a sidetable 1. Your task is to: look at bowl under the desklamp.
Action 1	**think:** To solve the task, I need to find and take a bowl, then find and use a desklamp.
Observation 1	OK.
Action 2	**think:** First I need to find a bowl. A bowl is more likely to appear in drawer (1-3), desk (1), sidetable (1-2), shelf (1-5), garbagecan (1). I can check one by one, starting with drawer 1.
Observation 2	OK.
Exploration	...
Action 11	**go to** shelf 2
Observation 11	On the shelf 2, you see a bowl 1.
Action 12	**think:** Now I find a bowl (1). Next, I need to take it.
Observation 12	OK.
Action 13	**take** bowl 1 from shelf 2
Observation 13	You pick up the bowl 1 from the shelf 2.
Action 14	**think:** Now I take a bowl (1). Next, I need to find a desklamp. A desklamp is more likely to appear in desk (1), sidetable (1-2), shelf (1-5), bed (1), drawer (1-3). I can check one by one, starting with desk 1.
Observation 14	OK.
Action 15	**go to** desk 1
Observation 15	On the desk 1, you see a alarmclock 2, a book 1, a cellphone 1, a keychain 1, a laptop 2, a laptop 1, and a pen 3.
Action 16	**go to** sidetable 1
Observation 16	On the sidetable 1, you see a cd 1, a pen 1, and a pencil 1.
Action 17	**go to** sidetable 2
Observation 17	On the sidetable 2, you see a alarmclock 1, a desklamp 1, and a pen 2.
Action 18	**think:** Now I find a desklamp (1). Next, I need to use it.
Observation 18	OK.
Action 19	use desklamp 1
Observation 19	You turn on the desklamp 1.

Table 13: ReAct human demonstration sample for ALFWorld PICK2 task

ALFWorld PICK2 Action (# unseen tasks: 17)	
Task	Put two saltshaker in drawer.
Initial observation	You are in the middle of a room. Looking quickly around you, you see a cabinet 20, a cabinet 19, a cabinet 18, a cabinet 17, a cabinet 16, a cabinet 15, a cabinet 14, a cabinet 13, a cabinet 12, a cabinet 11, ... , a stoveburner 1, and a toaster 1. Your task is to: put two saltshaker in drawer.
Action 1	**think:** To solve the task, I need to find and take the first saltshaker, then put it in drawer, then find and take the second salkshaker, then put it in drawer.
Observation 1	OK.
Action 2	**think:** First I need to find the first saltshaker. A saltshaker is more likely to appear in drawer (1-3), countertop (1-3), shelf (1-3), cabinet (1-20). I can check one by one, starting with drawer 1.
Observation 2	OK.
Exploration	...
Action 11	**go to** countertop 3
Observation 11	On the countertop 3, you see a bowl 2, a houseplant 1, a knife 2, a peppershaker 1, a saltshaker 4, a soapbottle 1, a spatula 2, and a tomato 3.
Action 12	**think:** Now I find the first saltshaker (4). Next, I need to take it.
Observation 12	OK.
Action 13	**take** saltshaker 4 from countertop 3
Observation 13	You pick up the saltshaker 4 from the countertop 3.
Action 14	**think:** Now I take the first saltshaker (4). Next, I need to put it in/on drawer 1.
Observation 14	OK.
Action 15	**go to** drawer 1
Observation 15	The drawer 1 is open. In it, you see nothing.
Action 16	**put** saltshaker 4 in/on drawer 1
Observation 16	You put the saltshaker 4 in/on the drawer 1.
Action 17	**think:** Now I put the first saltshaker in drawer. Next, I need to find the second saltshaker. I have visited drawer (1-3), countertop (1-3), so the second saltshaker is more likely to appear in shelf (1-3), cabinet (1-20). I can check one by one, starting with shelf 1.
Observation 17	OK.
Exploration	...
Action 25	**go to** cabinet 3
Observation 25	The cabinet 3 is closed.
Action 26	open cabinet 3
Observation 26	You open the cabinet 3. The cabinet 3 is open. In it, you see a saltshaker 3, and a saltshaker 2.
Action 27	**think:** Now I find the second saltshaker (2). Next, I need to take it.
Observation 27	OK.
Action 28	**take** saltshaker 2 from cabinet 3
Observation 28	You pick up the saltshaker 2 from the cabinet 3.
Action 29	**think:** Now I take the second saltshaker (2). Next, I need to put it in/on drawer 1.
Observation 29	OK.
Action 30	**go to** drawer 1
Observation 30	The drawer 1 is open. In it, you see a saltshaker 4.
Action 31	**put** saltshaker 2 in/on drawer 1
Observation 31	You put the saltshaker 2 in/on the drawer 1.

Strategy	Delivery Rate	Commonsense Pass Rate		Hard Constraint Pass Rate		Final Pass Rate
		Micro	Macro	Micro	Macro	
ReAct$_{\text{GPT-3.5}}$	82.2	47.6	3.9	11.4	6.7	0.6
Reflexion$_{\text{GPT-3.5}}$	93.9	53.8	2.8	11.0	2.8	0
	Open-source LLM - Gemma-2-9B					
Direct	**100**	**85.3**	**41.7**	37.9	13.3	5.6
CoT	**100**	79.1	13.9	**47.1**	**27.8**	5.6
LEAP & LEAN	**100**	72.5	15.6	26.3	18.9	**7.8**
	API-based LLM - Gemini					
Direct	**100**	90.3	42.2	**67.9**	**47.8**	19.4
CoT	**100**	**92.4**	**52.2**	67.1	**47.8**	**23.9**
LEAP & LEAN	**100**	84.9	40.0	50.7	36.1	**23.9**

Table 14: Performance indicators for LLM agent using LEAP & LEAN on TravelPlanner's validation split. Best results are shown in **bold**.

Table 15: Search Result LEAP prompt template and example for WebShop

Prompt Template	Follow my instructions properly.
	You are a real world agent who is shopping on the web.
	Select for me top-5 products with best matching options and features for "**[Search_Instruction]**"
	The details of the products available on the web are as below in json format.
	Please select only best matching product_ids.
	{
	[Search_Result_Products]
	}
	Only return 5 product ids from the json provided.

Prompt Example	Follow my instructions properly.
	You are a real world agent who is shopping on the web.
	Select for me top-5 products with best matching options and features for "black high quality cenglings womens cowl neck sweatshirt"
	The details of the products available on the web are as below in json format.
	Please select only best matching product_ids.
	{
	"**B09MTX95LM**": "ViYW Women's Floral Print Shirts Button Cowl Neck Long Sleeve Tunic Tops Fashion Autumn Warm Blouses Casual Soft Tee ; Price: $7.99 to $20.99",
	"**B09M472NR1**": "JJSUnS Women's Warm Long Sleeve Jackets With Hood Full Zip Up Fall Winter Tie Waist Coats Hoodie Windproof Outwear ; Price: $28.99 ",
	...
	...
	"**B09H599BPH**": "Women Y2K Hooded Sweatshirt, Unisex Los Angeles California Hoodies Retro Long Sleeve Pullovers Distressed Tops ; Price: $6.98 to $15.99",
	"**B07Y9K759Z**": "Barlver Women's Casual Long Sleeve Sweatshirts Fleece Cowl Neck Pullover Top Tunic Blouse Outwear ; Price: $12.99",
	...
	...
	"**B09PL8RNS9**": "WENKOMG1 Men's Thin Henley Shirts Comfy Casual T-Shirt Long Sleeve V-Neck Tops Regular-Fit Oversize Blouse Business Solid Color Polo Shirts Spring/Summer Sweatshirt(Gray,3X-Large) ; Price: $5.59",
	"**B09PLJ9RDX**": "WENKOMG1 Oversize T-Shirt for Men Long Sleeve Henley Shirts Casual Thin Tops Loose Solid Color Polo Shirts V-Neck Business Blouse Comfy Spring/Summer Regular-Fit Sweatshirt(Blue,XX-Large) ; Price: $5.19 "
	}
	Only return 5 product ids from the json provided.

Table 16: Product page LEAP prompt template and example for WebShop

Prompt Template	Follow my instructions properly.
	You are a real world agent who is shopping on the web.
	Select for me ONE best product with matching options and features for "**[Human_Instruction]**"
	The details of the products available on the web are as below in json format.
	Please select only best matching product_ids.
	{
	[Partial_lookup_response_Products]
	}
	Only return ONE of the selected best product's id.

Prompt Example	Follow my instructions properly.
	You are a real world agent who is shopping on the web.
	Select for me ONE best product with matching options and features for "black high quality cenglings womens cowl neck sweatshirt".
	The details of the products available on the web are as below in json format.
	Please select only best matching product_ids.
	{
	...
	"**B09M472NR1**": {
	"title_price": "JJSUnS Women's Warm Long Sleeve Jackets With Hood Full Zip Up Fall Winter Tie Waist Coats Hoodie Windproof Outwear ; Price: $28.99",
	"options": "size [small][medium][large][x-large]",
	"attributes": "long sleeve ; imported zipper ; light weight ; jacket women ; faux fur ; pullover hoodie ; loose fit ; daily wear ; slim fit ; fashion ; women's fashion hoodies & sweatshirts",
	"description": "Special V neck/High Neck/Crew Neck/U-Neck/Open Neck/Boat Neck/Scoop Neck/Leopard Print/Turtle Neck/Half Zip/-Cowl Neck design" },
	...
	}
	Only return ONE of the selected best product's id.

Table 17: LEAP Prompt example in ALFWorld

```
Interact with a household to solve a task. You should do thinking and acting
periodically.
Do not think more than thrice consecutively.You need to generate actions that
strictly follow the below templates:
1. goto [location] 2. take [object] from [location] 3. put [object] in/on [location]
3. open [location] 4. close [location] 5. toggle [object][location]
6. clean [object] with [something] 7. heat [object] with [receptacle] 8. cool [
object] with [receptacle]
Do not include any additional text apart from the action based on above templates.
If Nothing happens, try another action or think about possible alternatives.


Here are two examples. They are very relevant. Please use the actions in these
examples as your guidelines.
Example 1
...

Example 2
...

You are in the middle of a room. Looking quickly around you, you see a cabinet 4, a
cabinet 3, a cabinet 2, a cabinet 1, a countertop 1, a garbagecan 1, a
handtowelholder 2, a handtowelholder 1, a sinkbasin 2, a sinkbasin 1, a toilet 1, a
toiletpaperhanger 1, and a towelholder 1.

Your task is to: find two soapbar and put them in cabinet.

You should select 5 actions that are relevant to solving this task. You should ONLY
output the list action index in the format of [Option a, Option b, Option c, Option
d, Optione].

Option 1: Action: `go to handtowelholder 1` leads to Observation: `On the
handtowelholder 1, you see nothing.`.
Option 2: Action: `go to sinkbasin 1` leads to Observation: `On the sinkbasin 1, you
 see nothing.`.
...
Option 13: Action: `go to toilet 1` leads to Observation: `On the toilet 1, you see
a candle 1, a soapbar 2, and a spraybottle 3.`.
```

Table 18: System prompt (p_s) that is used at the beginning of the prompt for LEAN in WebShop

```
You are a web shopping agent.
Follow the illustration and perform in similar fashion to buy some product.
Make sure RESPONSE is in either of the format only \newline
* search[RESPONSE]
* click[RESPONSE]
* think[RESPONSE]
```

Table 19: In-context example chunk used by LEAN for “Search" action in WebShop

```
WebShop
Instruction:
i would like a 3 ounce bottle of bright citrus deodorant for sensitive skin, and
price lower than 50.00 dollars  \newline
[Search]

Action: Search[3 ounce bright citrus deodorant sensitive skin]
```

Table 20: In-context example chunk used by LEAN for reasoning after "Search" action in WebShop

```
WebShop
Instruction:
i would like a 3 ounce bottle of bright citrus deodorant for sensitive skin, and
price lower than 50.00 dollars
[Search]

Action: Search[3 ounce bright citrus deodorant sensitive skin]
Observation:
[Back to Search]
Page 1 (Total results: 50)
[Next >]
[B078GWRC1J]
Bright Citrus Deodorant by Earth Mama | Natural and Safe for Sensitive Skin,
Pregnancy and Breastfeeding, Contains Organic Calendula 3-Ounce
$10.99
[B078GTKVXY]
Ginger Fresh Deodorant by Earth Mama | Natural and Safe for Sensitive Skin,
Pregnancy and Breastfeeding, Contains Organic Calendula 3-Ounce
$10.99
[B08KBVJ4XN]
Barrel and Oak - Aluminum-Free Deodorant, Deodorant for Men, Essential Oil-Based
Scent, 24-Hour Odor Protection, Cedar & Patchouli Blend, Gentle on Sensitive Skin (
Mountain Sage, 2.7 oz, 2-Pack)
$15.95

Action: think[B078GWRC1J and B078GTKVXY are bright citrus deodorant less then 50
dollars. I can check B078GWRC1J first.]
```

Table 21: In-context example chunk used by LEAN for "Click" action in WebShop

```
WebShop
Instruction:
i would like a 3 ounce bottle of bright citrus deodorant for sensitive skin, and
price lower than 50.00 dollars
[Search]

Action: Search[3 ounce bright citrus deodorant sensitive skin]
Observation:
[Back to Search]
Page 1 (Total results: 50)
[Next >]
[B078GWRC1J]
Bright Citrus Deodorant by Earth Mama | Natural and Safe for Sensitive Skin,
Pregnancy and Breastfeeding, Contains Organic Calendula 3-Ounce
$10.99
[B078GTKVXY]
Ginger Fresh Deodorant by Earth Mama | Natural and Safe for Sensitive Skin,
Pregnancy and Breastfeeding, Contains Organic Calendula 3-Ounce
$10.99
[B08KBVJ4XN]
Barrel and Oak - Aluminum-Free Deodorant, Deodorant for Men, Essential Oil-Based
Scent, 24-Hour Odor Protection, Cedar & Patchouli Blend, Gentle on Sensitive Skin (
Mountain Sage, 2.7 oz, 2-Pack)
$15.95

Action: think[B078GWRC1J and B078GTKVXY are bright citrus deodorant less then 50
dollars. I can check B078GWRC1J first.]
Observation: OK.

Action: click[B078GWRC1J]
```

Table 22: In-context example chunk used by LEAN for reasoning after “Click" action in WebShop

```
WebShop
Instruction:
i would like a 3 ounce bottle of bright citrus deodorant for sensitive skin, and
price lower than 50.00 dollars
[Search]

Action: click[B078GWRC1J]
Observation:
[Back to Search]
[< Prev]
scent [assorted scents][bright citrus][calming lavender][ginger fresh][simply non-
scents]
size [travel set (4-pack)][3 ounce (pack of 1)][3-ounce (2-pack)]
Bright Citrus Deodorant by Earth Mama | Natural and Safe for Sensitive Skin,
Pregnancy and Breastfeeding, Contains Organic Calendula 3-Ounce
Price: $10.99
Rating: N.A.
[Description]
[Features]
[Reviews]
[Buy Now]

Action: think[For 3 ounce bottle of bright citrus deodorant for sensitive skin, the
item has options 'bright citrus' and '3 ounce (pack of 1)' and seems good to buy.]
```

Table 23: In-context example chunk used by LEAN for “Buy Now" action in WebShop

```
WebShop
Instruction:
i would like a 3 ounce bottle of bright citrus deodorant for sensitive skin, and
price lower than 50.00 dollars
[Search]

Action: click[B078GWRC1J]
Observation:
[Back to Search]
[< Prev]
scent [assorted scents][bright citrus][calming lavender][ginger fresh][simply non-
scents]
size [travel set (4-pack)][3 ounce (pack of 1)][3-ounce (2-pack)]
Bright Citrus Deodorant by Earth Mama | Natural and Safe for Sensitive Skin,
Pregnancy and Breastfeeding, Contains Organic Calendula 3-Ounce
Price: $10.99
Rating: N.A.
[Description]
[Features]
[Reviews]
[Buy Now]

Action: think[For 3 ounce bottle of bright citrus deodorant for sensitive skin, the
item has options 'bright citrus' and '3 ounce (pack of 1)' and seems good to buy.]
Observation: OK.

Action: click[Buy Now]
```

Table 24: System prompt (p_s) that is used at the beginning of the prompt for LEAN in ALFWorld

```
ALFWorld
Interact with a household to solve a task. You should do thinking and acting
periodically.
You need to generate actions that strictly follow the below templates:
1. goto [location] 2. take [object] from [location] 3. put [object] in/on [location]
3. open [something] 4. close [something] 5. toggle [object][location]
6. clean [object] with [something] 7. heat [object] with [receptacle] 8. cool [
object] with [receptacle]
If Nothing happens, try another action or think about possible alternatives.
Avoid exploring, go to, open, examine actions of the same locations or items over
and over again.
```

Table 25: In-context example chunk used by LEAN for "Finding object" action in ALFWorld

```
You are in the middle of a room. Looking quickly around you, you see a cabinet 4, a
cabinet 3, a cabinet 2, a cabinet 1, a countertop 1, a garbagecan 1, a
handtowelholder 2, a handtowelholder 1, a sinkbasin 2, a sinkbasin 1, a toilet 1, a
toiletpaperhanger 1, and a towelholder 1.

Your task is to: put some spraybottle on toilet.,

> think: To solve the task, I need to find and take a sparybottle, then put it on
toilet.
OK.
> think: First I need to find a spraybottle. A spraybottle is more likely to appear
in cabinet (1-4), countertop (1), toilet (1), sinkbasin (1-2), garbagecan (1). I can
 check one by one, starting with cabinet 1.
OK.
> go to cabinet 1
On the cabinet 1, you see a cloth 1, a soapbar 1, a soapbottle 1.
> go to cabinet 2
The cabinet 2 is closed.
> open cabinet 2
You open the cabinet 2. The cabinet 2 is open. In it, you see a candle 1 and a
spraybottle 2.
```

Table 26: In-context example chunk used by LEAN for "Picking up object" action in ALFWorld

```
You are in the middle of a room. Looking quickly around you, you see a cabinet 4, a
cabinet 3, a cabinet 2, a cabinet 1, a countertop 1, a garbagecan 1, a
handtowelholder 2, a handtowelholder 1, a sinkbasin 2, a sinkbasin 1, a toilet 1, a
toiletpaperhanger 1, and a towelholder 1.

Your task is to: put some spraybottle on toilet.

> think: Now I find a spraybottle (2). Next, I need to take it.
OK.
> take spraybottle 2 from cabinet 2
You pick up the spraybottle 2 from the cabinet 2.
```

Table 27: In-context example chunk used by LEAN for “Put down an object" action in ALFWorld

```
You are in the middle of a room. Looking quickly around you, you see a cabinet 4, a
cabinet 3, a cabinet 2, a cabinet 1, a countertop 1, a garbagecan 1, a
handtowelholder 2, a handtowelholder 1, a sinkbasin 2, a sinkbasin 1, a toilet 1, a
toiletpaperhanger 1, and a towelholder 1.

Your task is to: put some spraybottle on toilet.

> think: Now I take a spraybottle (2). Next, I need to put it in/on toilet 1.
OK.
> go to toilet 1
On the toilet 1, you see a soapbottle 2.
> put spraybottle 2 in/on toilet 1
You put the spraybottle 2 in/on the toilet 1.
```

Table 28: In-context example chunk used by LEAN for “Clean an object" action in ALFWorld

```
You are in the middle of a room. Looking quickly around you, you see a cabinet 13, a
 cabinet 12, a cabinet 11, a cabinet 10, a cabinet 9, a cabinet 8, a cabinet 7, a
cabinet 6, a cabinet 5, a cabinet 4, a cabinet 3, a cabinet 2, a cabinet 1, a
coffeemachine 1, a countertop 1, a diningtable 1, a drawer 1, a fridge 1, a
garbagecan 1, a microwave 1, a shelf 3, a shelf 2, a shelf 1, a sinkbasin 1, a
stoveburner 4, a stoveburner 3, a stoveburner 2, a stoveburner 1, and a toaster 1.

Your task is to: put a clean lettuce in diningtable.

> think: Now I take a lettuce (1). Next, I need to go to sinkbasin (1) and clean it.
OK.
> go to sinkbasin 1
On the sinkbasin 1, you see a apple 2, a ladle 2, a spoon 1, and a tomato 3.
> clean lettuce 1 with sinkbasin 1
You clean the lettuce 1 using the sinkbasin 1.
```

Table 29: In-context example chunk used by LEAN for “Cool an object" action in ALFWorld

```
You are in the middle of a room. Looking quickly around you, you see a cabinet 16, a
 cabinet 15, a cabinet 14, a cabinet 13, a cabinet 12, a cabinet 11, a cabinet 10, a
 cabinet 9, a cabinet 8, a cabinet 7, a cabinet 6, a cabinet 5, a cabinet 4, a
cabinet 3, a cabinet 2, a cabinet 1, a coffeemachine 1, a countertop 2, a countertop
 1, a diningtable 1, a drawer 5, a drawer 4, a drawer 3, a drawer 2, a drawer 1, a
fridge 1, a garbagecan 1, a microwave 1, a safe 1, a sinkbasin 1, a stoveburner 4, a
 stoveburner 3, a stoveburner 2, a stoveburner 1, and a toaster 1.

Your task is to: cool some pan and put it in stoveburner.

> think: Now I take a pan (1). Next, I need to go to a fridge (1) and cool it.
OK.
> go to fridge 1
The fridge 1 is closed.
> cool pan 1 with fridge 1
You cool the pan 1 using the fridge 1.
```

Table 30: In-context example chunk used by LEAN for “Heat an object" action in ALFWorld

```
You are in the middle of a room. Looking quickly around you, you see a cabinet 10, a
 cabinet 9, a cabinet 8, a cabinet 7, a cabinet 6, a cabinet 5, a cabinet 4, a
cabinet 3, a cabinet 2, a cabinet 1, a coffeemachine 1, a countertop 3, a countertop
 2, a countertop 1, a diningtable 1, a drawer 6, a drawer 5, a drawer 4, a drawer 3,
 a drawer 2, a drawer 1, a fridge 1, a garbagecan 1, a microwave 1, a sinkbasin 1, a
 stoveburner 4, a stoveburner 3, a stoveburner 2, a stoveburner 1, and a toaster 1.

Your task is to: heat some egg and put it in diningtable.

> think: Now I take an egg (2). Next, I need go to a microwave (1) and heat it.
OK.
> go to microwave 1
The microwave 1 is closed.
> heat egg 2 with microwave 1
You heat the egg 2 using the microwave 1.
```

Table 31: Prompt used for LEAP & LEAN in TravelPlanner

```
BASIC_TASK_INSTRUCTIONS:
You are a proficient travel planner.
You are provided with a Travel Query, Reference Information and illustration of
Travel Plan.
Using the provided Reference Information and Travel Query, please give me a detailed
 Travel Plan.
Make sure to include specific information for each day of trip, such as
    * Flights/Self-Driving/Taxi: flight numbers (e.g., F0123456) with arrival and
    departure times or self-driving/taxi details. Do not combine 'self-driving' and
    'flight' in the same trip
    * Restaurant: Suggest unique restaurants for Breakfast, lunch and dinner (e.g.
    restaurants_XXXX)
    * Attractions: In the city of visit (e.g. attractions_XXXX)
    * Accommodation (e.g. accommodations_XXX) names for each day of the trip
Each day plan should include 'day', 'current\_city', 'transportation', 'breakfast',
'attraction', 'lunch', 'dinner', and 'accommodation'.
Strictly follow the format provided in the illustration plan.
The information for each plan should be derived only from the reference information.
Use the symbol '-' to indicates that information is unavailable/unnecessary.
Most importantly, ensure that the total trip cost, stays within the specified budget
.
The travel plan should begin and end at the same city forming a closed circle.

RESTAURANTS_SHORTLISTING_PROMPT:
You are a proficient travel planner.
You are given a Travel Query along with a list of Restaurants Information.
Filter the restaurants that meet the travel criteria, ensuring no duplicates.
For each city in the itinerary, provide diverse selection of restaurants.
Do not create a travel plan, but only suggest restaurants.

ACCOMMODATIONS_SHORTLISTING_PROMPT:
You are a proficient travel planner.
You are given a Travel Query along with a list of Accommodation Information.
Filter the accommodations that meet the travel criteria, ensuring no duplicates.
For each city in the itinerary, provide a diverse selection of accommodations.
Do not create a travel plan, but only suggest accommodations.

ATTRACTIONS_SHORTLISTING_PROMPT:
You are a proficient travel planner.
You are given a Travel Query along with a list of Attractions Information.
Filter the attractions that meet the travel criteria, ensuring no duplicates.
For each city in the itinerary, provide a diverse selection of attractions.
Do not create a travel plan, but only suggest attractions.

FINAL_PROMPT_LEAP_LEAN:
BASIC_TASK_INSTRUCTIONS
## Travel Query
travel_query_for_task
## Reference information
reference_information # Obtained by shortlisting through SHORTLISTING_PROMPTs
## Illustration Travel Plan
ILLUSTRATION_TRAVEL_QUERY
ILLUSTRATION_TRAVEL_PLAN
## Illustration ends
```

Table 32: Prompts for Step Back strategy in TravelPlanner

```
RESTAURANTS_SHORTLISTING_PROMPT:
You are a proficient travel planner.
You are given a Travel Query along with a list of Restaurants Information.
Filter the restaurants that meet the travel criteria, ensuring no duplicates.
For each city in the itinerary, provide diverse selection of restaurants.
Do not create a travel plan, but only suggest restaurants.

ACCOMMODATIONS_SHORTLISTING_PROMPT:
You are a proficient travel planner.
You are given a Travel Query along with a list of Accommodation Information
Filter the accommodations that meet the travel criteria, ensuring no duplicates.
For each city in the itinerary, provide a diverse selection of accommodations.
Do not create a travel plan, but only suggest accommodations.

ATTRACTIONS_SHORTLISTING_PROMPT:
You are a proficient travel planner.
You are given a Travel Query along with a list of Attractions Information.
Filter the attractions that meet the travel criteria, ensuring no duplicates.
For each city in the itinerary, provide a diverse selection of attractions.
Do not create a travel plan, but only suggest attractions.
```

Table 33: Task Score and Success Rate (%) of LLMs using various prompting strategies on WebShop.

WebShop environment		
	Task Score	**Success Rate**
Human Expert	82.1	59.6
Baselines (Our runs with (Yao et al., 2022a)'s code)		
Rule-based	44.8	9.2
IL	60.4	28.0
IL+RL	62.4	28.7
Few-shot CoT strategy(Liu et al., 2023)		
Chatglm-6b	0.5	-
Vicuna-7B	2.2	-
Llama-2-7B	11.6	-
Codegeex2-6b	20.9	-
Codellama-7B	25.2	-
GPT-4-0613	61.1	-
GPT-3.5-turbo-0613	64.1	-
ReAct strategy (Our runs)		
Phi-2	0	0
Qwen-4B	9.3	2.8
Vicuna-7B	18.1	3.4
Qwen-7B	15.7	4.6
Llama-3.1-8B	22.0	9.2
Gemma-2-9B	26.3	10.0
GPT-3.5	12.6	3.2
Gemini	35.4	21.8
PaLM-540B(Yao et al., 2022b)	66.6	40.0
LEAN Strategy		
Phi-2	9.6	3.0
Qwen-4B	26.9	13.6
Vicuna-7B	47.2	25.4
Qwen-7B	50.8	25.0
Llama-3.1-8B	30.7	16.2
Gemma-2-9B	45.0	25.8
GPT-3.5	22.2	6.8
Gemini	53.6	35.0
LEAP Strategy		
Phi-2	52.7	16.6
Qwen-4B	55.3	17.2
Vicuna-7B	57.2	20.2
Qwen-7B	58.8	21.4
Llama-3.1-8B	59.6	24.2
Gemma-2-9B	63.1	27.4
GPT-3.5	62.9	24.2
Gemini	70.4	42.8

Table 34: Inefficient Planning by Llama-3.1-8B Using LEAN in WebShop: Misinterpretation and Over-Exploration.

Misinterpretation and Over-Exploration	
Initial observation	WebShop Instruction: i'm looking for a meals with zero added sugar and also free from gluten and bpa. also, choose applesauce flavored one, and price lower than 40.00 dollars [Search] Action:
Action 1	**Search**[zero added sugar gluten free bpa free applesauce flavored meal]
Observation 1	... [B07ZJXRJJ1] GoGo squeeZ Organic Applesauce on the Go, ... $32.76 [B00GSQ6DSS] ... 3.2 Ounce (4 Pouches), Gluten Free, Vegan Friendly, ... $100.0 **[B091DL4V6F]** Yummy Organics Baby Food - Organic Fruit and Vegetable Puree, Non-GMO, Gluten-Free, No Added Sugar or Salt, Resealable and Reusable BPA-Free Cups, Applesauce, 12 Pack **$21.0**
Action 2	**think**[B07ZJXRJJ1 and B091DL4V6F are both options, but B091DL4V6F is a 12-pack and has a lower price. I can check B091DL4V6F first.]
Observation 2	Ok.
Action 3	**Click**[B091DL4V6F]
Observation 3	...
Action 4	**think**[The product has applesauce flavor, is gluten-free, and has no added sugar, but the price is higher than the desired $40.00. **However, it's a good match for the search criteria, but the price is a concern.**]
Observation 4	Ok.
Action 5	**Search**[applesauce flavored baby food, gluten-free, no added sugar, bpa-free, price lower than 40.00 dollars]
Observation 5	... **[B001VJ70UC]** GoGo SqueeZ Fruit on The Go, Apple, Tasty Kids Applesauce Snacks Made from Apples, Gluten, Nut & Dairy Free, Vegan, 18 Pouches, 57.6 Oz **$11.06**
Action 6	**think**[B001VJ70UC is applesauce flavored, gluten-free, and bpa-free, but I need to check if it has added sugar. I'll check the product details.]
Observation 6	Ok.
Action 7	**Click**[B001VJ70UC]
Observation 7	...
Action 8	**think**[The product has applesauce flavor, is gluten-free, and has no added sugar, but the price is slightly above $40.00. However, it's a good match for the search criteria, **but I should continue searching to find a better option** within the price range.]
Observation 8	Ok.
Action 9	...
Observation 9	...
	... ...
Action 30	...
Observation 30	...

Table 35: Reward Model input text

Input template	[{ "role": "user", "content": **goal_instruction** }, { "role": "assistant", "content": **product** },]
Input example	[{ "role": "user", "content": I need gluten free vegetarian smoked peppered bacon - 4 ounce (pack of 2), and price lower than 50.00 dollars. }, { "role": "assistant", "content": $64.99 - OMEALS Pasta Fagioli Six Vegetarian MRE Sustainable Premium Outdoor Fully Cooked Meals w/Heater - Extended Shelf Life - No Refrigeration - Perfect for Travelers, Emergency Supplies - USA 6 Pack },]

WebShop environment						
	Task Score	**Success Rate**	**Task Score**	**Success Rate**	**Task Score**	**Success Rate**
Rule-based	44.8	9.2	-	-	-	-
Human Expert	82.1	59.6	-	-	-	-
	ReAct		**LEAP**		**LEAN**	
Open-source LLMs						
Phi-2	0	0	52.7	16.6	9.6	3.0
Qwen-4B	9.3	2.8	55.3	17.2	26.9	13.6
Vicuna-7B	18.1	3.4	57.2	20.2	47.2	25.4
Qwen-7B	15.7	4.6	58.8	21.4	**50.8**	25.0
Llama-3.1-8B	22.0	9.2	59.6	24.2	30.7	16.2
Gemma-2-9B	**26.3**	**10.0**	**63.1**	**27.4**	45.0	**25.8**
Average	$15.2_{\pm 9.4}$	$5.0_{\pm 3.9}$	$57.8_{\pm 3.6}$	$21.2_{\pm 4.1}$	$35.0_{\pm 15.7}$	$19.2_{\pm 9.1}$
API-based LLMs						
GPT-3.5	12.6	3.2	62.9	24.2	22.2	6.8
Gemini	**35.4**	**21.8**	**70.4**	**42.8**	**53.6**	**35.0**

Table 36: Task Score and Success Rate (%) of LLMs using ReAct, LEAP and LEAN strategies on WebShop.

WebShop environment		
	Task Score	**Success Rate**
LEAP	63.1	27.4
LEAP$_{+\text{ RM}}$	**68.3**	**36.6**
LEAN	45.0	25.8
LEAP & LEAN	50.8	27.6
LEAP & LEAN$_{+\text{ RM}}$	**54.8**	**30.8**

Table 37: Task Score and Success Rate (%) of utilizing Gemma-2-9B LLM with LEAP, LEAN and Reward Model (RM) combination on WebShop.