Do Think Tags Really Help LLMs Plan? A Critical Evaluation of ReAct-Style Prompting

Mudit Verma[*] [†]
School of Computing and AI
Arizona State University
muditverma@asu.edu

Siddhant Bhambri[*]
School of Computing and AI
Arizona State University
siddhantbhambri@asu.edu

Subbarao Kambhampati
School of Computing and AI
Arizona State University
rao@asu.edu

Abstract

The reasoning abilities of Large Language Models (LLMs) remain a topic of debate, which are critically tested in sequential decision-making problems. ReAct, a recently popular method has gained popularity for claiming to enhance LLM reasoning abilities while directly prompting them by "*interleaving reasoning trace with action execution*" in text-based planning domains such as AlfWorld and WebShop. However, given the different components of ReAct-style prompting, it remains unclear what the source of improvement in LLM performance is. In this paper, we critically examine the claims of ReAct-style prompting for sequential decision-making problems. By introducing systematic variations to the input prompt, we perform a sensitivity analysis along the original claims of ReAct. Contrary to these claims and common use-cases that utilize ReAct-style prompting, we find that the performance is minimally influenced by the interleaved reasoning trace or by the content of these generated reasoning traces. Instead, the performance of LLMs is primarily driven by the unreasonably high degree of similarity between input example tasks and queries, implicitly forcing the prompt designer to provide instance-specific examples which significantly increases the cognitive burden on the human. Our empirical results, on the same suite of domains as ReAct, show that the perceived reasoning abilities of LLMs stem from the exemplar-query similarity and approximate retrieval rather than any inherent reasoning abilities.

1 Introduction

Large Language Models (LLMs) have seen rapid advancements specifically in Natural Language Processing and Understanding (NLP & NLU). LLMs have unparalleled capabilities in text generation, summarization, translation, question answering to name a few. [Bubeck et al., 2023]. Motivated by these capabilities of LLMs, there has also been a rush to look for other emergent abilities–especially for reasoning and planning. A popular way of enhancing LLM performance on reasoning/planning tasks has been in-context prompting or prompt-engineering [Sahoo et al., 2024] to include instructions [Giray, 2023], syntax structure [Marvin et al., 2023], criticism and plan guidance with verification [Kambhampati et al., 2024] etc. Among these approaches, ReAct [Yao et al., 2022b], presented at ICLR 2023, stands out which claims to improve LLM planning abilities through the use of reasoning traces interleaved with action execution given as plan guidance.

Given the seemingly widespread adoption of ReAct methodology (as of this writing, it has 1,408 citations), the brittleness we witnessed in our initial experiments calls for a systematic study of the factors contributing to the performance of ReAct-based LLM Agents. Moreover, recent studies have

[*]Equal contribution.
[†]Work done while a PhD student at ASU, now at Google DeepMind.

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

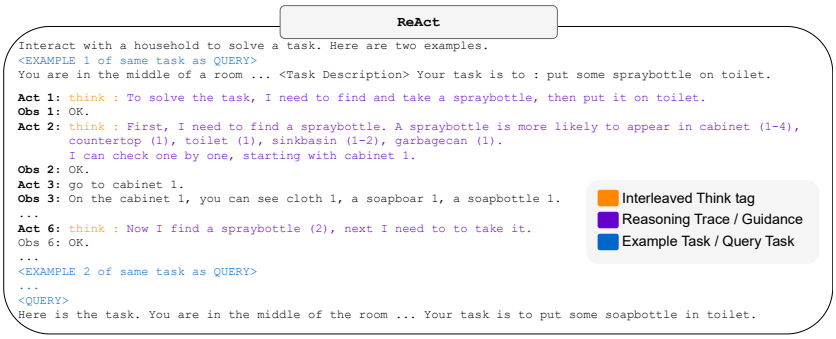

Figure 1: An example of ReAct in AlfWorld. We highlight the main components of ReAct, i.e., Interleaved reasoning and acting, the reasoning trace / plan guidance and the example and query task.

highlighted similar case studies on the original Chain-of-Thought idea [Stechly et al., 2024a, Sprague et al., 2024]. In this work, we systematically evaluate the brittleness of ReAct by studying which potential factors contribute to its performance. This analysis is conducted through variations in input prompts to understand how a ReAct LLM Agent responds to (1) where the guidance is provided, (2) the different types and structure of this guidance, and finally, (3) on varying the resemblance of example prompt to the queried problem. We investigate the research questions : **RQ1:** Does the agent performance depend on interleaving reasoning trace with action execution?
RQ2: How does the nature of the reasoning trace affect the performance of LLM Agents?
RQ3: How does the similarity between the example ⟨problem, solution⟩and the query ⟨problem, ? ⟩, which are present in the prompt, affect LLM Agent performance?

We conduct extensive experiments on the AlfWorld and WebShop domain using various LLM Models, including GPT-3.5-turbo, GPT-3.5-instruct, GPT-4-0314, GPT-4-0613 (latest variant), GPT-4o, Claude-Opus and Llama 3.1-8b. Our findings highlight that the benefits of ReAct-style prompting are present when prompt engineers can curate highly instance-specific examples. This may not scale for domains with a large number of problem instance classes, and it places an undue burden on prompt engineers to provide instance specific examples. Finally, our experiments call into question claims of enhanced "emergent reasoning" of LLMs with prompt engineering efforts; corroborating contemporary research [Verma et al., 2024, Valmeekam et al., 2024, Ullman, 2023, Schaeffer et al., 2023, McCoy et al., 2023, Stechly et al., 2024a] questioning reasoning abilities of LLMs.

2 Related Work

Large Language Models have been shown to be successful in several natural language tasks [Kocoń et al., 2023, Gilardi et al., 2023, Zhu et al., 2023, Bubeck et al., 2023, Bhattacharjee et al., 2024]. However, there are two schools of thought when it comes to utilizing off-the-shelf LLMs for planning and reasoning tasks. Chain of Thought, ReAct, and other works that followed [Wei et al., 2022, Yao et al., 2023, Long, 2023, Yao et al., 2024, Besta et al., 2024, Fu et al., 2024, Aksitov et al., 2023], have argued about the reasoning abilities of LLMs by proposing prompting methods. On the other hand, [Valmeekam et al., 2024, Stechly et al., 2024a,b, Sprague et al., 2024] have refuted these claims by showing the inability of LLMs to solve deterministic planning and classical reasoning problems.

In particular, for investigating the use of LLMs in solving decision making problems, AlfWorld [Shridhar et al., 2020] is a popular domain that was originally proposed for training text-based Reinforcement Learning agents. Lately, works such as ReAct, Reflexion, and their variants [Yao et al., 2022b, Shinn et al., 2023] have argued on the prowess of LLMs' reasoning abilities on AlfWorld. Furthermore, there have been several extensions to ReAct that boost their generalization abilities across more domains including multi-modal domains [Yang et al., 2023, Castrejon et al., 2024], autonomous vehicles [Cui et al., 2024], table question answering [Zhang et al., 2023], etc. While the effectiveness of ReAct is celebrated across different areas, these works only depend on anthropomorphization of LLMs for using ReAct based prompting with no justification on the source of improvement in performance. This motivates our work in investigating the components of ReAct with respect to sequential decision-making problems and analyzing the role each component plays.

3 Preliminaries

AlfWorld:[Shridhar et al., 2020] is a synthetic text-based game built on top of a STRIPS-style PDDL domain description [Fikes and Nilsson, 1971]. ReAct [Yao et al., 2022b] defines six tasks (or problem classes) within this domain namely - Put, Clean, Heat, Cool, Examine, and PutTwo. Each problem class consists of several problem instances, such as *put a spraybottle on toilet* (see Fig. 1 is an example instance of Put class.

WebShop: [Yao et al., 2022a] is an online shopping website environment with 1.18M real-world products and 12K human instructions. The agent is provided with an initial human instruction (for example, "I am looking for a nightstand with drawers. It should have a nickel finish, and priced lower than $140"). The agent's task is to crawl the shopping environment using actions such as `search 'nightstand drawers', choose 'white buttons', back to search`, etc.

ReAct:[Yao et al., 2022b] claims to increase LLM's performance on text-based planning tasks such as AlfWorld and WebShop primarily by augmenting the original action space of the agent with a *think* action. The *think* action tag provided by ReAct is claimed to comprise of **Re**asoning + **Act**ion trace that is provided in the solution for the example problems (exemplars) as part of the prompt. During execution, the LLM is expected to generate a *think* action tag for the queried problem instance that is semantically similar to the one provided for the examples in the prompt. **Location of *THINK* tag:** In ReAct, the integration of the *think* tag within actions serves to expand the action space. This allows the language model (LLM) agent to execute a *think* action, prompting an *'OK'* response. **Content of *THINK* tag:** In ReAct, the *think* action consistently provides the decision-making agent with success-oriented guidance for task completion. For instance, upon encountering a `spraybottle`, the prompt might include: `think: Now I find a spraybottle (2). Next, I need to take it.` This guidance exposes forthcoming actions and sub-tasks for the agent. **Few shot *EXAMPLE*s:** In the AlfWorld domain (wihch is a PDDL domain), ReAct authors [Yao et al., 2022b] classify six problem classes or tasks: `Put, Clean, Heat, Cool, Examine, PutTwo`. Despite representing different tasks, they share the same environment dynamics and action space, allowing for very similar execution trace.

4 A Critical Evaluation of ReAct-Style Prompting

We examine the claims of ReAct to understand the performance of ReAct-based LLM agents. It is crucial to assess whether ReAct's fundamental claims hold, particularly in planning. We perform a sensitivity analysis by proposing alternatives along the three dimensions of ReAct (Section 3). We provide detailed examples and prompts for each proposed variation in Appendix B, C and D.

4.1 RQ1 : Interleaving *think*ing with acting

Does the agent performance depend on interleaving reasoning trace with action execution?

Variation 1: Exemplar-based CoT AlfWorld is a partially observable environment where an agent can only observe objects after reaching that location. Hence, we remove specific location and object identifiers to modify the *think* actions that are originally interleaved with other actions in the environment, and append all the *think* actions together at the beginning of the example problem. **Intuition:** Problem-specific guidance for a sequential decision-making agent can be given step-by-step (as in ReAct) or all at once.

Variation 2: Anonymized Exemplar-CoT We take one step further and modify the *think* tag to remove references to specific locations and objects, making it more general. **Intuition:** Exemplars can be made more general by providing abstract guidance and exploiting LLMs ability to identify necessary semantic entity relations.

4.2 RQ2 : Plan Guidance following *think* tag

How does the nature of the reasoning trace or guidance information affect the performance of LLM?

Variation 1: Failure We inject two invalid actions in the execution trace : the first that attempts to execute the action pertinent to the task (such as `put spraybottle 2 in/on toilet`) when not possible and, second, executes some other invalid action. We include the expected simulator response,

Table 1: Average Success % of LLM for RQ1 and RQ2 on six AlfWorld tasks.

Model / Prompt	Act	ReAct	RQ1		RQ2			
			CoT	Anon. CoT	Placebo	Order	Failure	Explanation
GPT-3.5-Turbo	34.3	27.6	46.6	41	30	28.3	43.3	41.6
GPT-3.5-Instruct	44	50.7	61.9	50.7	41	42.5	47	44.7
GPT-4-0314 (Old)	-	23.3	43.3	33.3	36.6	30	50	36.6
GPT-4-0613 (Latest)	70.0	26.7	40.0	26.6	36.6	30	60	36.6
Claude-Opus	43.3	56.6	50	46.6	30	50	53.3	30

`Nothing happens.`, when invalid actions are taken. **Intuition:** Reasoning trace can be about *what to do* such as subgoals of the future, or *what not to do* such as mistakes in hindsight.

Variation 2: Failure + Explanation We place *think* actions after invalid actions injected in Failure Variation which consist of explanations for the failure such as `think: Nothing happens because I do not have a spraybottle 2`. **Intuition:** We can augment pointing out mistakes in hindsight with explanations to avoid similar failures. This is stronger guidance signal than Failure, however, the exemplars still not provide information on what to do next.

Variation 3: Guidance Ordering LLMs are known to be susceptible to minor syntactic perturbations to inputs. We test whether it is true for guidance information given as prompt as well. We identify chain of sub-tasks in a reasoning trace and reverse it. **Intuition:** LLM agent should be invariant to the syntax of reasoning trace if the semantic information is preserved. This does not change the reasoning trace from the perspective of information content.

Variation 4: Placebo Guidance It is unclear to what extent LLM agent uses the supposed helpful thoughts for the decision making task. In this variation we replace *think* tag guidance with a placebo thought that does not contain any task relevant information, but has been widely used as prompt engineering trick [Kojima et al., 2022]. **Intuition:** According to claims of ReAct, we expect the performance to get worse when the guidance does not have any information useful for task success.

4.3 RQ3 : Similarity between *EXAMPLE*s and *QUERY*

How does the similarity between the example ⟨problem, solution⟩and the query ⟨problem, ? ⟩, which are present in the prompt, affect LLM Agent performance?

Variation 1: Synonyms - (`Domain`) For this variation, we replace the object and location names in the example prompts with their synonyms. **Intuition:** Exemplar guidance maybe specified with alternate synonymous object and location names. Reasoning agents should be invariant to variable name substitution for closed world dynamics such as PDDL based AlfWorld.

Variation 2: Problem Instance-level - `Instance` We change the goal location in exemplar problem to ensure that it does not match with any of the goal locations in query problem. **Intuition:** Ensuring a different goal location in exemplar from the queried problem is a natural use-case.

Variation 3: Problem Level - `Both, One, All` In general, all the AlfWorld tasks share a large portion of actions (such as exploring cabinets and locations, picking objects etc.). Motivated by how tight relationship of these tasks we come up with three variations. First, `One`, uses one exemplar of an arbitrarily picked task and the other exemplar of the same task as the query. Second, `Both`, uses both exemplars from an arbitrarily picked task. Finally, `All`, uses a total of six exemplars (this is the only variation where we provide more than the standard two examples as in ReAct) corresponding to each task under consideration. **Intuition:** With a very similar action execution trace (such as exploration, picking/placing objects) across tasks, and shared dynamics, LLM agent should be minimally affected by the use of exemplars of a different task.

Variation 4: Exploration Strategy - `Optimal` An important ingredient to the exemplars is the exploration strategy used. In this variation we provide exemplars which serendipitously take the optimal actions (as if the environment were fully observable) and therefore the example plan is the shortest possible. **Intuition:** Exploration strategy exposed in exemplars (that too for the same problem task) should not impact ReAct's performance if the LLM agent is reasoning instead of retrieval (or pattern matching).

Table 2: Average Success % of LLM for RQ3 on six AlfWorld tasks. OC: Out of context limit

Model / Prompt	Act	ReAct	RQ3					
			Domain	Instance	Optimal	All	One	Both
GPT-3.5-Turbo	34.3	27.6	1.6	30	20.1	32	28.3	1.6
GPT-3.5-Instruct	44	50.7	47.6	42.5	39.5	OC	17.9	5.2
GPT-4-0314 (Old)	–	23.3	13.3	23.3	50	23.3	16.6	0
GPT-4-0613 (Latest)	70.0	26.7	10.0	20.0	53.3	23.3	20	3.3
Claude-Opus	43.3	56.6	50	46.6	43.3	50	60	6.6

5 Results

In this section, we discuss our findings using our experiments on the AlfWorld domain. We refer readers to Table 3 in Appendix C.4 for results on WebShop domain, and a thorough discussion of our results in Appendix C.5.

Utility of interleaving reasoning trace with action execution: From Table 1 (RQ1), note that the exemplar CoT and the anonymized exemplar CoT performs significantly better than base ReAct for all GPT-X family of models. Moreover, the performance dips slightly for Claude-Opus along these variations. A surprising result consistent in both the domains is the performance of `Act` baseline (where think tags are absent and actions are generated directly). `Act` baseline is weaker only for two models `GPT-3.5-Instruct, Claude-Opus` for both the domains, which further questions the utility of using ReAct style paradigm in the first place.

Utility of Guidance Information following *think* tag: Table 1 indicates that hindsight guidance (`Failure, Explanation`) improves all GPT models' performances. The Claude-Opus model's performance remains stable with hindsight (`Failure`) guidance and declines with placebo guidance. Finally, contrary to the general perception that better GPT models would improve over reasoning, we find that GPT-4-(Old)'s performance is the worst among GPT-X family further highlighting the brittleness in ReAct's generalizability. GPT-4-(Latest) performs similarly to GPT-4-(Old), except for the Act baseline which again shows the futility of ReAct prompting.

Utility of Exemplar similarity to Query task: Table 2 shows the severe brittleness of ReAct based LLM agent to even minor variations (such as `Domain, Instance`). Specifically, performance of GPT-3.5-Turbo and GPT-4 plummets for `Domain`. Claude-Opus which was more robust in RQ1, RQ2, is also impacted severely by `Domain, Instance` variations. Furthermore, when we do not expose the exploration strategy and only provide Optimal exemplars, the performance of LLM agents further drops (except in GPT4). However, overloading the LLMs with more exemplars `All` does not impact its performance. We posit that this is because the query-task exemplar is still part of the large input prompt. Among the two exemplars, as provided in ReAct, when one of them is of a different task (`One`) then the performance significantly reduces for LLMs. When both of the exemplars are of a different task then the performance collapses to single digit success rates for all the models. This is a key result of this work highlighting the severe dependence of LLMs on the similarity of the exemplars to the query task. Through sensitivity analysis using our RQ3 variations we could find parts of the input (the task similarity of the exemplar with query) which is the source of ReAct performance. Essentially, the LLM is mimicking / performing approximate retrieval from the context presented to it.

6 Conclusion

ReAct-based prompting methods have been claimed to improve planning abilities of Large Language Models. In this study, we critically examine ReAct along three dimensions, informed by its claims and our hypotheses regarding its performance sources. Contrary to ReAct's claims, our findings reveal that its performance is **neither** due to interleaving the *think* tag with action execution, **nor** due to the content of this *think* tag. Instead, we identify that the true source of LLM performance in sequential decision-making tasks, is the high degree of similarity between exemplar problems (few-shot) and the query task. We also showed that ReAct is susceptible to trivial variations in exemplar prompts (such as with the use of synonyms, or `Unrolling` and `Subtask Similarity` cases). Our findings caution against an uncritical adoption of ReAct-style frameworks for their putative abilities to enhance performance in domains requiring planning.

Acknowledgement

This research is supported in part by ONR grant N0001423-1-2409, and gifts from Qualcomm, J.P. Morgan and Amazon.

References

Renat Aksitov, Sobhan Miryoosefi, Zonglin Li, Daliang Li, Sheila Babayan, Kavya Kopparapu, Zachary Fisher, Ruiqi Guo, Sushant Prakash, Pranesh Srinivasan, et al. Rest meets react: Self-improvement for multi-step reasoning llm agent. arXiv preprint arXiv:2312.10003, 2023.

Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Michal Podstawski, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Hubert Niewiadomski, Piotr Nyczyk, et al. Graph of thoughts: Solving elaborate problems with large language models. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 38, pages 17682–17690, 2024.

Amrita Bhattacharjee, Raha Moraffah, Joshua Garland, and Huan Liu. Towards llm-guided causal explainability for black-box text classifiers. 2024.

Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, et al. Sparks of artificial general intelligence: Early experiments with gpt-4. arXiv preprint arXiv:2303.12712, 2023.

Lluis Castrejon, Thomas Mensink, Howard Zhou, Vittorio Ferrari, Andre Araujo, and Jasper Uijlings. Hammr: Hierarchical multimodal react agents for generic vqa. arXiv preprint arXiv:2404.05465, 2024.

Can Cui, Yunsheng Ma, Xu Cao, Wenqian Ye, and Ziran Wang. Receive, reason, and react: Drive as you say, with large language models in autonomous vehicles. IEEE Intelligent Transportation Systems Magazine, 2024.

Richard E Fikes and Nils J Nilsson. Strips: A new approach to the application of theorem proving to problem solving. Artificial intelligence, 2(3-4):189–208, 1971.

Dayuan Fu, Jianzhao Huang, Siyuan Lu, Guanting Dong, Yejie Wang, Keqing He, and Weiran Xu. Pre-act: Predicting future in react enhances agent's planning ability. arXiv preprint arXiv:2402.11534, 2024.

Fabrizio Gilardi, Meysam Alizadeh, and Maël Kubli. Chatgpt outperforms crowd workers for text-annotation tasks. Proceedings of the National Academy of Sciences, 120(30):e2305016120, 2023.

Louie Giray. Prompt engineering with chatgpt: a guide for academic writers. Annals of biomedical engineering, 51(12):2629–2633, 2023.

Subbarao Kambhampati, Karthik Valmeekam, Lin Guan, Kaya Stechly, Mudit Verma, Siddhant Bhambri, Lucas Saldyt, and Anil Murthy. Llms can't plan, but can help planning in llm-modulo frameworks. arXiv preprint arXiv:2402.01817, 2024.

Jan Kocoń, Igor Cichecki, Oliwier Kaszyca, Mateusz Kochanek, Dominika Szydło, Joanna Baran, Julita Bielaniewicz, Marcin Gruza, Arkadiusz Janz, Kamil Kanclerz, et al. Chatgpt: Jack of all trades, master of none. Information Fusion, 99:101861, 2023.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. Advances in neural information processing systems, 35: 22199–22213, 2022.

Jieyi Long. Large language model guided tree-of-thought. arXiv preprint arXiv:2305.08291, 2023.

Ggaliwango Marvin, Nakayiza Hellen, Daudi Jjingo, and Joyce Nakatumba-Nabende. Prompt engineering in large language models. In International Conference on Data Intelligence and Cognitive Informatics, pages 387–402. Springer, 2023.

R Thomas McCoy, Shunyu Yao, Dan Friedman, Matthew Hardy, and Thomas L Griffiths. Embers of autoregression: Understanding large language models through the problem they are trained to solve. arXiv preprint arXiv:2309.13638, 2023.

Shamik Roy, Sailik Sengupta, Daniele Bonadiman, Saab Mansour, and Arshit Gupta. Flap: Flow adhering planning with constrained decoding in llms. arXiv preprint arXiv:2403.05766, 2024.

Pranab Sahoo, Ayush Kumar Singh, Sriparna Saha, Vinija Jain, Samrat Mondal, and Aman Chadha. A systematic survey of prompt engineering in large language models: Techniques and applications. arXiv preprint arXiv:2402.07927, 2024.

Rylan Schaeffer, Kateryna Pistunova, Samar Khanna, Sarthak Consul, and Sanmi Koyejo. Invalid logic, equivalent gains: The bizarreness of reasoning in language model prompting. arXiv preprint arXiv:2307.10573, 2023.

Noah Shinn, Beck Labash, and Ashwin Gopinath. Reflexion: an autonomous agent with dynamic memory and self-reflection. arXiv preprint arXiv:2303.11366, 2023.

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

Zayne Sprague, Fangcong Yin, Juan Diego Rodriguez, Dongwei Jiang, Manya Wadhwa, Prasann Singhal, Xinyu Zhao, Xi Ye, Kyle Mahowald, and Greg Durrett. To cot or not to cot? chain-of-thought helps mainly on math and symbolic reasoning. arXiv preprint arXiv:2409.12183, 2024.

Kaya Stechly, Karthik Valmeekam, and Subbarao Kambhampati. Chain of thoughtlessness: An analysis of cot in planning. arXiv preprint arXiv:2405.04776, 2024a.

Kaya Stechly, Karthik Valmeekam, and Subbarao Kambhampati. On the self-verification limitations of large language models on reasoning and planning tasks. arXiv preprint arXiv:2402.08115, 2024b.

Tomer Ullman. Large language models fail on trivial alterations to theory-of-mind tasks. arXiv preprint arXiv:2302.08399, 2023.

Karthik Valmeekam, Matthew Marquez, Sarath Sreedharan, and Subbarao Kambhampati. On the planning abilities of large language models-a critical investigation. Advances in Neural Information Processing Systems, 36, 2024.

Mudit Verma, Siddhant Bhambri, and Subbarao Kambhampati. Theory of mind abilities of large language models in human-robot interaction: An illusion? In Companion of the 2024 ACM/IEEE International Conference on Human-Robot Interaction, pages 36–45, 2024.

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

Zhengyuan Yang, Linjie Li, Jianfeng Wang, Kevin Lin, Ehsan Azarnasab, Faisal Ahmed, Zicheng Liu, Ce Liu, Michael Zeng, and Lijuan Wang. Mm-react: Prompting chatgpt for multimodal reasoning and action. arXiv preprint arXiv:2303.11381, 2023.

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

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In The Eleventh International Conference on Learning Representations, 2022b.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. Advances in Neural Information Processing Systems, 36, 2024.

Yao Yao, Zuchao Li, and Hai Zhao. Beyond chain-of-thought, effective graph-of-thought reasoning in large language models. arXiv preprint arXiv:2305.16582, 2023.

Yunjia Zhang, Jordan Henkel, Avrilia Floratou, Joyce Cahoon, Shaleen Deep, and Jignesh M Patel. Reactable: Enhancing react for table question answering. arXiv preprint arXiv:2310.00815, 2023.

Yiming Zhu, Peixian Zhang, Ehsan-Ul Haq, Pan Hui, and Gareth Tyson. Can chatgpt reproduce human-generated labels? a study of social computing tasks. arXiv preprint arXiv:2304.10145, 2023.

A Resources Used

In this work we leverage OpenAI API and Claude API for prompting the Language Models. We use `gpt-4-0613` for GPT4, `gpt-3.5-turbo-0125`, `gpt-3.5-turbo-instruct`, `claude-3-opus-20240229`, `claude-3-sonnet-20240229` and `claude-3-haiku-20240307` for all our experimentation in April-May 2024. ReAct and corresponding experiments use approximately 14M input tokens (due to repeated prompting after each action execution) and 150K output tokens for 134 problem instances as used by ReAct.

B Additional Considerations

B.1 Fine-grained performance on each task on AlfWorld

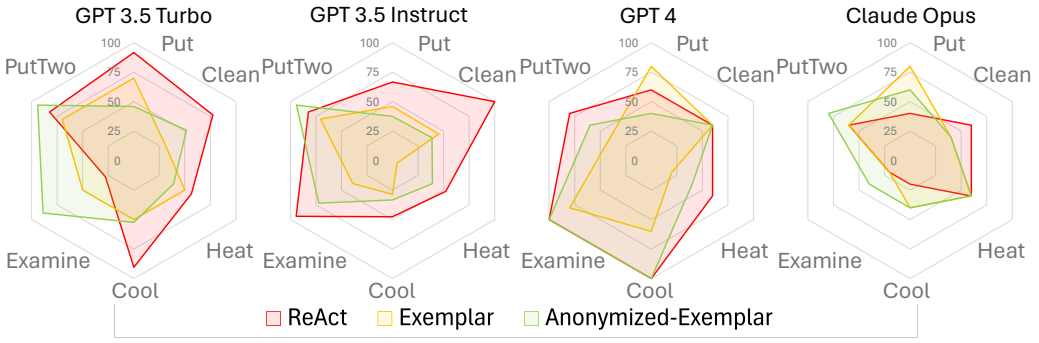

Figure 2: The radar chart shows the **failure rates** of various LLMs with different ReAct-based prompt settings for RQ1 (Base React, Global, Anonymized) across six Alfworld tasks (hexagon vertices). Higher values / Larger shaded region indicate worse performance.

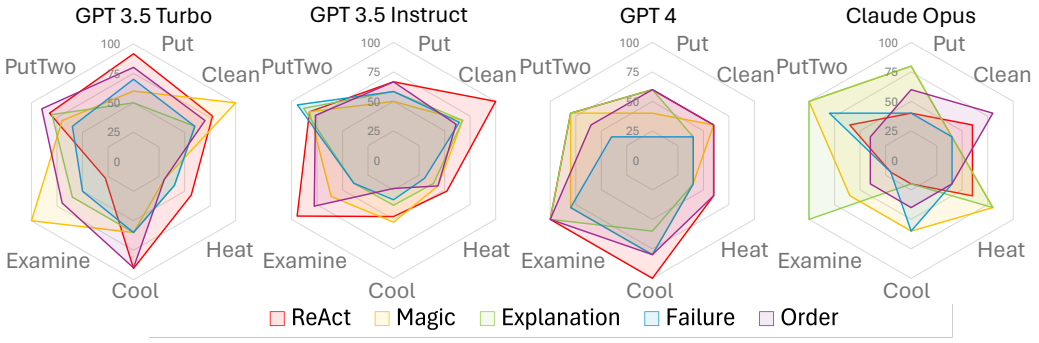

Figure 3: The radar chart shows the **failure rates** of various LLMs with different ReAct-based prompt settings for RQ2 (Base React, Magic, Failure, Failure+Explanation, Ordering) across six Alfworld tasks (hexagon vertices). Higher values / Larger shaded region indicate worse performance.

Figures 2, 3 and 4 shows a radar chart highlighting the failure cases of various LLMs with different ReAct based prompting variations for RQ1, RQ2 and RQ3 respectively on AlfWorld domain. Note that GPT-4 in the figures refer to GPT-4-0314 (Old Variant).

B.2 Failure Rates

We report failure rates in the radar chart as in Figs. 2, 3, 4 and 5 instead of success rates. We attempted to visualize the severe brittleness given by the larger area of the shaded region. Since, for various of our RQ variations the LLMs performance was very low, we decided to report failure rate given as (100 - Success Rate %) instead.

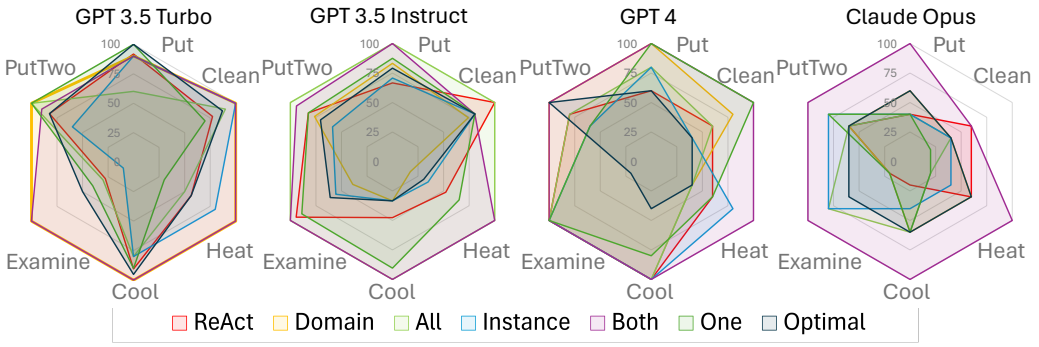

Figure 4: The radar chart shows the **failure rates** of various LLMs with different ReAct-based prompt settings for RQ3 (Base React, Domain, Instance, All, Both, One) across six Alfworld tasks (hexagon vertices). Higher values / Larger shaded region indicate worse performance.

B.3 Performance of Claude-Haiku

We skip on mentioning the performance of Claude-Haiku, since it was not able to generate syntactically correct actions for any of the instances. We found that following our instruction to generate specific actions as in the exemplar was difficult. We improved the prompt to have specific instructions for generating actions (See D.3) but it did not yield any improvements for Claude-Haiku. However, the instruction did help with Claude-Sonnet and Claude-Opus. We find that Claude-Sonnet follows a similar pattern as GPT-3.5-Instruct as presented in our results, and decided to focus ourselves on the strongest/largest Claude model (Claude-Opus) for our evaluation.

B.4 Extension to other Models

We are in the process of experimenting with GPT-4o and Google Gemini models, APIs for which were released in May 2024 which does not allow enough time for thorough and verified evaluation before the conference submission. For completeness, however, we will experiment with these APIs as they become accessible and append our results.

B.5 Main Results on Exemplar CoT variant

While this work does not investigates effectiveness of exemplar Chain of Thought as presented in RQ1, we do however test the main results of the work with Exemplar CoT to identify whether our findings hold true there as well. That is, we test `RQ3-Both, RQ3-One`. For `GPT-3.5-Turbo` we find that the average performance drops from 46.6% (RQ3-Exemplar CoT) as in Table 1 to 28.3% in `One` and 10.4% in `Both` variation cases, and remains at 40.3% for `All` variation.

C Experiment Design

Each of the variations proposed along RQ1, RQ2 and RQ3 modifies the few-shot examples only. Remaining aspects such as the query problem or the interaction with the simulator is directly inherited from the ReAct code-base Yao et al. [2022b] at publicly available at `https://github.com/ysymyth/ReAct`. Our code can be found in the attached supplementary material.

Except `All` RQ3 variation, all other settings use the standard two examplars for prompting the LLM. Depending on the variation we change the content of the exemplar. Full prompts can be found in the attached supplementary code.

C.1 Running the experiments

In our experiments, according to the variation style we take the exemplar prompts and use the same exemplar prompts across the instances of the query task. Other than `RQ3-Both, One` we use the exemplar of the same task as the query as done in ReAct (and still find brittleness of ReAct). For `RQ3-Both, One` we use exactly two exemplars but of a different task than query. Finally, `RQ3-All`

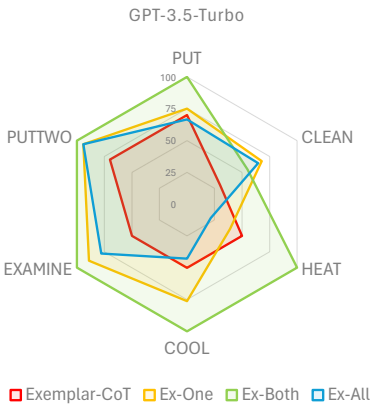

Figure 5: The radar chart shows the failure rates of GPT-3.5-Turbo with our RQ3 variants on ReAct across six Alfworld tasks. Higher values / Larger shaded region indicate worse performance.

is the only variation that provides six exemplars (instead of two) and we force the exemplar of the query-task to be appended at the end in the prompt. This was the best performing prompting strategy (on GPT-3.5-Turbo) amongst when the query-task exemplar was placed at the beginning, at position 4 (middle) and at the end.

While the original ReAct experiments were carried out on PaLM (currently decommissioned), we reproduce their results with newer set of models. We use GPT-3.5-Turbo, GPT-3.5-Instruct, GPT-4, GPT-4o, and Claude-Opus, which are all newer models than those benchmarked in ReAct [Yao et al., 2022b]. Note, that despite using newer models, our results shed doubts on the reproducibility and consistency across models of the original paper's results. As noted, we use the setup in [Yao et al., 2022b] for all our experiments. In AlfWorld, GPT3.5(Turbo, Instruct) results are on 134 instances across six tasks, GPT-4/Claude-Opus on 60 instances (10 for each task) due to cost considerations. In WebShop, GPT3.5(Turbo, Instruct), GPT-4o, LLAMA-3.1-8B results are on 500 samples, GPT-4/Claude-Opus are on 50 instances due to cost considerations.

The reported success-rate from the ReAct paper Yao et al. [2022b] on the WebShop domain is 40%. Due to the absence of the exact queries used in the paper, we randomly sampled queries from the WebShop dataset comprising 12K records. This approach possibly resulted in the decoupling of any relationship between the exemplars and the queries. Referring to Table 3, it is evident that the performance of the WebShop ReAct agent significantly declined, reaching single digit percentages (as well as other variants). This mirrors the trends observed in the `Both` variant of the Alfworld in Table 2, further supporting our arguments.

C.2 Hyperparameters

We use $temperature = \tau = 0$ for all of the GPT and Claude models and set `max-tokens = 100` which is borrowed from ReAct's hyperparameters. Rest of the parameters are kept to be default as specified in the respective model's API documentation.

C.3 Additional Experiments

Unrolling and Subtask Similarity We perform additional experiments where the query task is to essentially repeat the task in the exemplar (`Unrolling`). For instance, in AlfWorld, the exemplar is `Put` and the query is `PutTwo` to put two objects at given location. In this case, the LLM has to unroll the given advice and repeat exemplar task execution to solve the query. The success rate of GPT-3.5-Instruct (the best performing GPT model in our experiments) drops down from 52% to 9%. Similarly, we experiment with a `Subtask Similarity` variation where the exemplar task subsumes execution of the query task. For instance, the `Heat` task requires the agent to pick and place object in the microwave (which is an instantiation of `Put` task). One would expect that `Heat` is a good exemplar for `Put`, however, the performance of GPT-3.5-instruct model goes from 18% to 0% in this

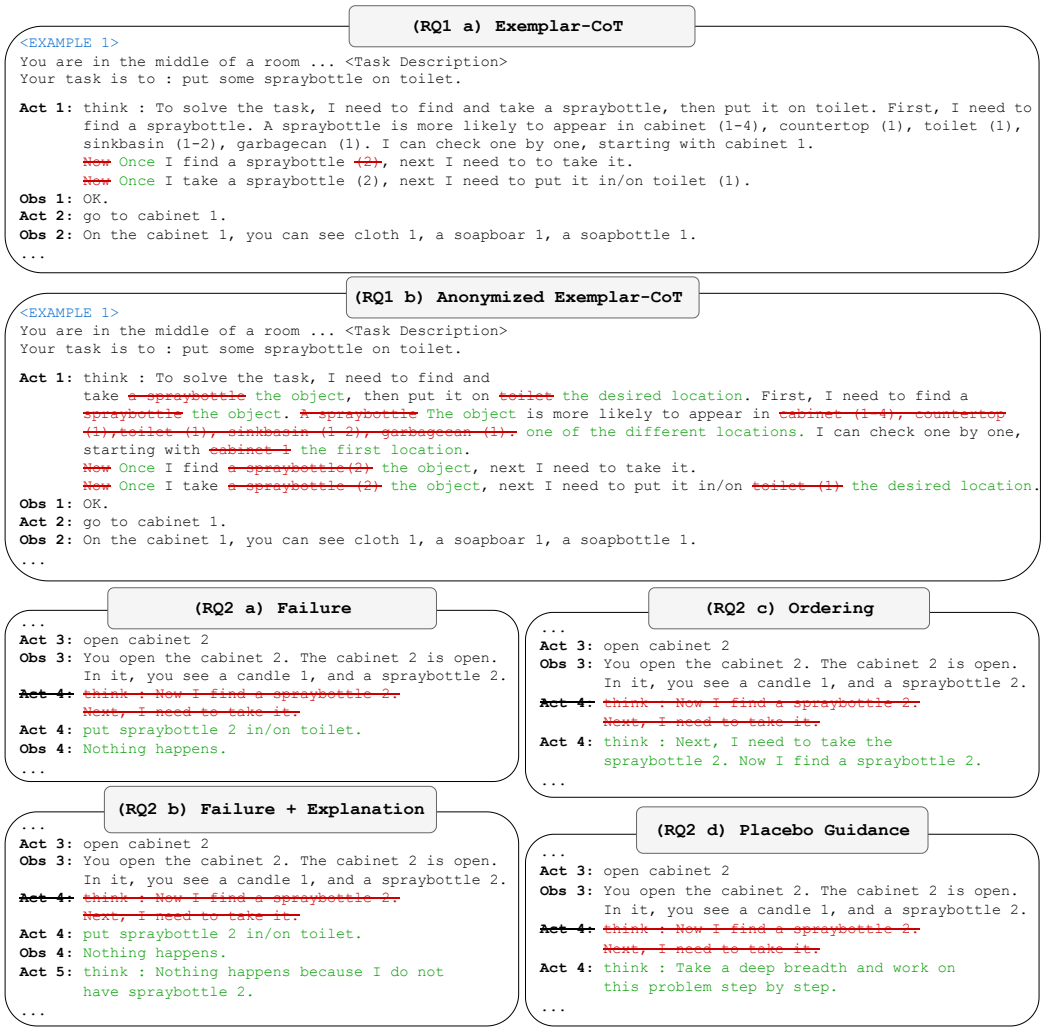

Figure 6: Example of prompt variations considered for RQ1 and RQ2.

case. These results further underscore the brittleness and the need for instance-specific exemplars in ReAct.

Thought operationalization ability of LLMs Given the free form nature of thought generation and arbitrary nature of thought (about subtask, common-sense next steps etc.), checking whether the generated thoughts are in-fact reasonable is a challenging problem. For completeness, we find that 40% of the times after generation of a *think* tag, subsequent environment action taken by the LLM was invalid (for GPT-3.5-instruct) in AlfWorld. It is much higher (80% for GPT-3.5-Turbo, 90% for Claude-Haiku) for weaker LLM models. This further highlights the inability of LLMs to operationalize its generated thought as also seen in [Roy et al., 2024]. From manual inspection we find that the typical thoughts would enlist all possible locations as next locations to visit for most of the tasks. As demonstrated in Section 5, the performance of LLMs actually decreases when provided with foresight guidance, as seen with the base ReAct model. A detailed investigation into the validity of the generated reasoning traces is beyond the scope of this work and is suggested as future research.

Table 3: Average Success % of LLM for RQ1 and RQ2 on WebShop tasks.

Model / Prompt	Act	ReAct	RQ1		RQ2		
			CoT	Anon. CoT	Placebo	Failure	Explanation
GPT-3.5-Turbo	1.12	1.04	2.20	1.88	1.52	3.48	3.48
GPT-3.5-Instruct	7.24	7.16	7.52	6.12	7.40	7.20	7.24
GPT-4-0613 (Latest)	8	4	8	8	6	8	8
GPT-4o	4.64	2.24	4.68	4.52	4.08	4.68	4.68
Claude-Opus	4	4	4	2	4	2	4
LLAMA-3.1-8B	1.44	3.16	3.28	3.92	2.04	1.20	2.16

C.4 WebShop Results

C.5 Discussion

In this sub-section, we aim to draw insights from our experiments across the three RQs which can be further extended to understanding the limitations of LLMs for planning problems. Specifically, we discuss a) the pitfalls of using ReAct-style prompting for planning domains which could further be exacerbated by approaches that build on top of ReAct framework, and b) scalability and generalizability issues as observed across multiple LLMs.

Pitfalls of ReAct-Style Prompting: Recall, that ReAct claims an improved performance for text-based planning domains, namely - AlfWorld and WebShop, where the presence of a *think* tag acts as guidance for the LLM to generate the next set of actions during the LLM-environment interaction. Through our sensitivity analysis, we dissect each component of ReAct-style prompting in a critical effort to understand the factor that leads to the observed success rates in these domains. With variations on the placement (RQ1) and content (RQ2) of the *think* tag, we eliminate it as the primary cause of any improvement. Furthermore, slight variations in exemplar tasks (RQ3) lead to a stark decline in success rate, clearly indicating the dependence of performance on the highly curated instance-specific examples by domain experts. While newer research in the art of prompting has pointed out the impact of well-curated examples, our work specifically highlights exemplar-query similarity as the cause of ReAct's performance and rejects contemporary belief that the heavy-lifting of LLM reasoning & planning is done through the `think` tag.

Relevance of ReAct to newer LLMs: ReAct uses the `Act` baseline in their work to showcase improvements due to the presence of the proposed *think* tag. For AlfWorld, ReAct reports 45% success rate for `Act` baseline and 71% success rate for `ReAct` prompting using the PaLM model. For WebShop, ReAct reports 30.1% success rate for `Act` baseline and 40% success rate for `ReAct` prompting on PaLM. However, we note from our results on both domains that the `Act` baseline performs much better than ReAct for several LLMs, which questions on the compatibility of ReAct to newer-age LLMs. ReAct performs worse with newer models as compared to the results they report on PaLM, which is currently decommissioned. This observation also questions the contemporary belief that such prompting strategies are generalizable throughout different LLM families, including newer models.

We re-iterate our key result, given any LLM model, the success rates plummet with our RQ3 variations showing a consistent pattern of dependence on the provided examples irrespective of the LLM. Moreover, the performance of all the LLMs remain quite high (if not better) when we vary the location and content of the `think` tags. This highlights the need for higher rigor in agentic LLM experimentation and in-depth evaluation seeking source of improvements. Finally, we highlight our previous discussion on *unrolling, subtask-similarity* (discussing the brittleness of perceived reasoning abilities of LLMs) and the inability of LLMs to perform reliable *thought operationalization* as key limitations which exist despite ReAct style prompting.

D Example Prompts

The full list of curated variations can be found in the supplementary materials. However, for completeness we reference the prompt used for base ReAct (as in [Yao et al., 2022b]) and our variations along RQ1, RQ2 and RQ3 for the `Put` task.

D.1 Synonym Substitution mapping for `Domain`

We make the following substitutions to the object names / locations in the exemplar prompt in the `Domain` variation. Note that these substitutions are done only to the exemplar, and the query problem and subsequent interaction with the AlfWorld simulator uses the original vocabulary mapping.

```
spraybottle -> aerosolbottle
cabinet -> cupboard
countertop -> worktop
sinkbasin -> sinkbowl
toilet -> lavatory
toiletpaperhanger -> toiletpaperholder
towelholder -> towelrack

microwave -> oven
shelf -> rack
drawer -> deskdrawer
stoveburner -> hob
diningtable -> table
garbagecan -> trashbin

fridge -> refrigerator
peppershaker -> pepperpot
room -> livingroom
bread -> breadloaf
pan -> fryingpan
pot -> saucepan
book -> notebook

creditcard -> amexcard
mirror -> lookingglass
dresser -> chestofdrawers
sofa -> couch
cellphone -> mobilephone
coffeemachine -> coffeemaker
knife -> kitchenknife
spatula -> turner
soapbottle -> liquidsoapdispenser
saltshaker -> saltpot
statue -> sculpture
vase -> flowerpot
dishsponge -> spongewipe
desklamp -> tablelamp
sidetable -> nightstand
```

D.2 For `All, Both, One`

All: We take the exemplar prompt for each task and concatenate it together.

Both : We use the following mapping generated arbitrarily, to replace the exemplar prompt for the query task.

```
VARIATION_MAPPING = {
    'put':'examine',
    'clean':'cool',
    'heat':'put',
    'cool':'puttwo',
    'examine':'clean',
    'puttwo':'heat'
}
```

One : We use the first example from the above mapping, and the second example from the same task as the query.

D.3 Building the input prompt

We inherit ReAct's way of stiching together the exemplars and the query using the following prompt template as :

```
Interact with a household to solve a task. Here are two examples.' + eg1
+ eg2 + INSTRUCTION + Here is the task. (TASK)´
```

In the case of GPT models `INSTRUCTION` is an empty string. Claude Models gave 0% success rate (regardless of the model size, i.e. Haiku, Sonnet and Opus) with the original ReAct prompt design. We identified that Claude required an explicit instruction to generate only the actions for the simulator. We added a task independent INSTRUCTION prompt as :

```
INSTRUCTION = " Only provide the one action at a time. Be concise, and
do not provide any extra information. Always start with the action. For
example, 'pick up apple' is correct, but 'I want to pick up the apple' is
not."
```

Interestingly, GPT-3-Turbo performance dropped to 1% (averaged across 6 independent runs over 134 examples) with the inclusion of this INSTRUCTION tag. This further highlights how brittle prompt engineering is, and cautions us against misplacing our trust with ReAct in enhancing reasoning abilities of LLMs.

D.4 RQ1

Prompt 1: rq1-anon-exemplar

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.
> think: To solve the task, I need to find and take the object, then put it on the desired location. First I need to find the object. The object is more likely to appear in one of the different locations. I can check one by one, starting with the first location. Once I find the object, next I need to take it. Once I take the object, next I need to put it in/on the desired location.
OK.
> go to fridge 1
The fridge 1 is closed.
> open fridge 1
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.
> go to diningtable 1
On the diningtable 1, you see nothing.
> go to diningtable 2
On the diningtable 2, you see a pineapple 1.
> go to diningtable 3
On the diningtable 3, you see a bowl 2, a bowl 1, a bread 2, and a bread 1.
> go to sidetable 1
On the sidetable 1, you see a cup 1, a lettuce 1, a peppershaker 3, a potato 1, and a saltshaker 1.
> go to countertop 1
On the countertop 1, you see a butterknife 1, a dishsponge 1, and a soapbottle 1.
> go to sinkbasin 1
On the sinkbasin 1, you see a cup 2, a egg 1, a mug 1, and a potato 3.
> go to garbagecan 1
On the garbagecan 1, you see a apple 3, and a egg 3.
> take apple 3 from garbagecan 1
You pick up the apple 3 from the garbagecan 1.
> go to sidetable 1
On the sidetable 1, you see a cup 1, a lettuce 1, a peppershaker 3, a potato 1, and a saltshaker 1.
> put apple 3 in/on sidetable 1
You put the apple 3 in/on the sidetable 1.

Prompt 2: rq1-exemplar

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.
> think: To solve the task, I need to find and take an apple, then put it in sidetable. 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. Once I find an apple, next I need to take it. Once I find an apple, next I need to put it in/on sidetable.
OK.
> go to fridge 1
The fridge 1 is closed.
> open fridge 1
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.
> go to diningtable 1
On the diningtable 1, you see nothing.
> go to diningtable 2
On the diningtable 2, you see a pineapple 1.
> go to diningtable 3
On the diningtable 3, you see a bowl 2, a bowl 1, a bread 2, and a bread 1.
> go to sidetable 1
On the sidetable 1, you see a cup 1, a lettuce 1, a peppershaker 3, a potato 1, and a saltshaker 1.
> go to countertop 1
On the countertop 1, you see a butterknife 1, a dishsponge 1, and a soapbottle 1.
> go to sinkbasin 1
On the sinkbasin 1, you see a cup 2, a egg 1, a mug 1, and a potato 3.
> go to garbagecan 1
On the garbagecan 1, you see a apple 3, and a egg 3.
> take apple 3 from garbagecan 1
You pick up the apple 3 from the garbagecan 1.
> go to sidetable 1
On the sidetable 1, you see a cup 1, a lettuce 1, a peppershaker 3, a potato 1, and a saltshaker 1.
> put apple 3 in/on sidetable 1
You put the apple 3 in/on the sidetable 1.

Prompt 3: base-react

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.
> 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.
OK.
> go to fridge 1
The fridge 1 is closed.
> open fridge 1
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.
> go to diningtable 1
On the diningtable 1, you see nothing.
> go to diningtable 2
On the diningtable 2, you see a pineapple 1.
> go to diningtable 3
On the diningtable 3, you see a bowl 2, a bowl 1, a bread 2, and a bread 1.
> go to sidetable 1
On the sidetable 1, you see a cup 1, a lettuce 1, a peppershaker 3, a potato 1, and a saltshaker 1.
> go to countertop 1
On the countertop 1, you see a butterknife 1, a dishsponge 1, and a soapbottle 1.
> go to sinkbasin 1
On the sinkbasin 1, you see a cup 2, a egg 1, a mug 1, and a potato 3.
> go to garbagecan 1
On the garbagecan 1, you see a apple 3, and a egg 3.
> take apple 3 from garbagecan 1
You pick up the apple 3 from the garbagecan 1.
> think: Now I find a apple (3). Next, I need to put it in/on sidetable 1.
OK.
> go to sidetable 1
On the sidetable 1, you see a cup 1, a lettuce 1, a peppershaker 3, a potato 1, and a saltshaker 1.
> put apple 3 in/on sidetable 1
You put the apple 3 in/on the sidetable 1.

D.5 RQ2

Prompt 4: rq2-Explanation

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.
> go to fridge 1
The fridge 1 is closed.
> close cabinet 2.
Nothing Happens.
> think: Nothing happens because you are not at cabinet 2.
OK.
> open fridge 1
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.
> go to diningtable 1
On the diningtable 1, you see nothing.
> go to diningtable 2
On the diningtable 2, you see a pineapple 1.
> go to diningtable 3
On the diningtable 3, you see a bowl 2, a bowl 1, a bread 2, and a bread 1.
> go to sidetable 1
On the sidetable 1, you see a cup 1, a lettuce 1, a peppershaker 3, a potato 1, and a saltshaker 1.
> go to countertop 1
On the countertop 1, you see a butterknife 1, a dishsponge 1, and a soapbottle 1.
> go to sinkbasin 1
On the sinkbasin 1, you see a cup 2, a egg 1, a mug 1, and a potato 3.
> go to garbagecan 1
On the garbagecan 1, you see a apple 3, and a egg 3.
> put apple 3 in/on sidetable.
Nothing Happens.
> think: Nothing happens because I do not have apple 3.
OK.
> take apple 3 from garbagecan 1
You pick up the apple 3 from the garbagecan 1.
> go to sidetable 1
On the sidetable 1, you see a cup 1, a lettuce 1, a peppershaker 3, a potato 1, and a saltshaker 1.
> put apple 3 in/on sidetable 1
You put the apple 3 in/on the sidetable 1.

Prompt 5: rq2-ordering

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.
> think: To solve the task, I need to put an apple in sidetable, for which I need to find and take an apple.
OK.
> think: An apple is more likely to appear in sidetables (1), fridges (1), countertops (1), diningtables (1-3), garbagecan (1), sinkbasins (1). First I need to find an apple. I can check one by one, starting with fridge 1.
OK.
> go to fridge 1
The fridge 1 is closed.
> open fridge 1
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.
> go to diningtable 1
On the diningtable 1, you see nothing.
> go to diningtable 2
On the diningtable 2, you see a pineapple 1.
> go to diningtable 3
On the diningtable 3, you see a bowl 2, a bowl 1, a bread 2, and a bread 1.
> go to sidetable 1
On the sidetable 1, you see a cup 1, a lettuce 1, a peppershaker 3, a potato 1, and a saltshaker 1.
> go to countertop 1
On the countertop 1, you see a butterknife 1, a dishsponge 1, and a soapbottle 1.
> go to sinkbasin 1
On the sinkbasin 1, you see a cup 2, a egg 1, a mug 1, and a potato 3.
> go to garbagecan 1
On the garbagecan 1, you see a apple 3, and a egg 3.
> think: Next, I need to take the apple.Now I find an apple (3).
OK.
> take apple 3 from garbagecan 1
You pick up the apple 3 from the garbagecan 1.
> think: Next, I need to put the apple in/on sidetable 1. Now I find an apple (3).
OK.
> go to sidetable 1
On the sidetable 1, you see a cup 1, a lettuce 1, a peppershaker 3, a potato 1, and a saltshaker 1.
> put apple 3 in/on sidetable 1
You put the apple 3 in/on the sidetable 1.

Prompt 6: rq2-Placebo

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.
> think: Take a deep breath and work on this problem step-by-step.
OK.
> go to fridge 1
The fridge 1 is closed.
> open fridge 1
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.
> go to diningtable 1
On the diningtable 1, you see nothing.
> go to diningtable 2
On the diningtable 2, you see a pineapple 1.
> go to diningtable 3
On the diningtable 3, you see a bowl 2, a bowl 1, a bread 2, and a bread 1.
> go to sidetable 1
On the sidetable 1, you see a cup 1, a lettuce 1, a peppershaker 3, a potato 1, and a saltshaker 1.
> go to countertop 1
On the countertop 1, you see a butterknife 1, a dishsponge 1, and a soapbottle 1.
> go to sinkbasin 1
On the sinkbasin 1, you see a cup 2, a egg 1, a mug 1, and a potato 3.
> go to garbagecan 1
On the garbagecan 1, you see a apple 3, and a egg 3.
> think: Take a deep breath and work on this problem step-by-step.
OK.
> take apple 3 from garbagecan 1
You pick up the apple 3 from the garbagecan 1.
> think: Take a deep breath and work on this problem step-by-step.
OK.
> go to sidetable 1
On the sidetable 1, you see a cup 1, a lettuce 1, a peppershaker 3, a potato 1, and a saltshaker 1.
> put apple 3 in/on sidetable 1
You put the apple 3 in/on the sidetable 1.

Prompt 7: rq2-Failure

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.
> go to fridge 1
The fridge 1 is closed.
> close cabinet 2.
Nothing Happens.
> open fridge 1
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.
> go to diningtable 1
On the diningtable 1, you see nothing.
> go to diningtable 2
On the diningtable 2, you see a pineapple 1.
> go to diningtable 3
On the diningtable 3, you see a bowl 2, a bowl 1, a bread 2, and a bread 1.
> go to sidetable 1
On the sidetable 1, you see a cup 1, a lettuce 1, a peppershaker 3, a potato 1, and a saltshaker 1.
> go to countertop 1
On the countertop 1, you see a butterknife 1, a dishsponge 1, and a soapbottle 1.
> go to sinkbasin 1
On the sinkbasin 1, you see a cup 2, a egg 1, a mug 1, and a potato 3.
> go to garbagecan 1
On the garbagecan 1, you see a apple 3, and a egg 3.
> put apple 3 in/on sidetable.
Nothing Happens.
> take apple 3 from garbagecan 1
You pick up the apple 3 from the garbagecan 1.
> go to sidetable 1
On the sidetable 1, you see a cup 1, a lettuce 1, a peppershaker 3, a potato 1, and a saltshaker 1.
> put apple 3 in/on sidetable 1
You put the apple 3 in/on the sidetable 1.

D.6 RQ3

Prompt 8: rq3-Domain

You are in the middle of a livingroom. Looking quickly around you, you see a cupboard 4, a cupboard 3, a cupboard 2, a cupboard 1, a coffeemaker 1, a worktop 1, a table 3, a table 2, a table 1, a deskdrawer 1, a refrigerator 1, a trashbin 1, a oven 1, a nightstand 1, a sinkbowl 1, a hob 4, a hob 3, a hob 2, a hob 1, and a toaster 1.
Your task is to: find some apple and put it in nightstand.
> think: To solve the task, I need to find and take an apple, then put it in nightstand.
OK.
> think: First I need to find an apple. An apple is more likely to appear in refrigerators (1), tables (1-3), nightstands (1), worktops (1), sinkbowls (1), trashbin (1). I can check one by one, starting with refrigerator 1.
OK.
> go to refrigerator 1
The refrigerator 1 is closed.
> oballpointpen refrigerator 1
You oballpointpen the refrigerator 1. The refrigerator 1 is oballpointpen. In it, you see a lettuce 2, a mug 2, a saucepanato 2, and a tomato 1.
> go to table 1
On the table 1, you see nothing.
> go to table 2
On the table 2, you see a pineapple 1.
> go to table 3
On the table 3, you see a bowl 2, a bowl 1, a breadloaf 2, and a breadloaf 1.
> go to nightstand 1
On the nightstand 1, you see a cup 1, a lettuce 1, a peppersaucepan 3, a saucepanato 1, and a saltpot 1.
> go to worktop 1
On the worktop 1, you see a butterknife 1, a spongewipe 1, and a liquidsoapdispenser 1.
> go to sinkbowl 1
On the sinkbowl 1, you see a cup 2, a egg 1, a mug 1, and a saucepanato 3.
> go to trashbin 1
On the trashbin 1, you see a apple 3, and a egg 3.
> think: Now I find an apple (3). Next, I need to take it.
> take apple 3 from trashbin 1
You pick up the apple 3 from the trashbin 1.
> think: Now I find an apple (3). Next, I need to put it in/on nightstand 1.
OK.
> go to nightstand 1
On the nightstand 1, you see a cup 1, a lettuce 1, a peppersaucepan 3, a saucepanato 1, and a saltpot 1.
> put apple 3 in/on nightstand 1
You put the apple 3 in/on the nightstand 1.

Prompt 9: rq3-Instance

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.
> think: To solve the task, I need to find and take an apple, then put it in sidetable.
OK.
> 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.
OK.
> go to fridge 1
The fridge 1 is closed.
> open fridge 1
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.
> go to diningtable 1
On the diningtable 1, you see nothing.
> go to diningtable 2
On the diningtable 2, you see a pineapple 1.
> go to diningtable 3
On the diningtable 3, you see a bowl 2, a bowl 1, a bread 2, and a bread 1.
> go to sidetable 1
On the sidetable 1, you see a cup 1, a lettuce 1, a peppershaker 3, a potato 1, and a saltshaker 1.
> go to countertop 1
On the countertop 1, you see a butterknife 1, a dishsponge 1, and a soapbottle 1.
> go to sinkbasin 1
On the sinkbasin 1, you see a cup 2, a egg 1, a mug 1, and a potato 3.
> go to diningtable 2
On the diningtable 2, you see a pineapple 1.
> go to diningtable 3
On the diningtable 3, you see a bowl 2, a bowl 1, a bread 2, and a bread 1.
> go to sidetable 1
On the sidetable 1, you see a cup 1, a lettuce 1, a peppershaker 3, a potato 1, and a saltshaker 1.
> go to countertop 1
On the countertop 1, you see a butterknife 1, a dishsponge 1, and a soapbottle 1.
> go to sinkbasin 1
On the sinkbasin 1, you see a cup 2, a egg 1, a mug 1, and a potato 3.
> go to garbagecan 1
On the garbagecan 1, you see a apple 3, and a egg 3.
> think: Now I find an apple (3). Next, I need to take it.
> take apple 3 from garbagecan 1
You pick up the apple 3 from the garbagecan 1.
> think: Now I find an apple (3). Next, I need to put it in/on sidetable 1.
OK.
> go to sidetable 1
On the sidetable 1, you see a cup 1, a lettuce 1, a peppershaker 3, a potato 1, and a saltshaker 1.
> put apple 3 in/on sidetable 1
You put the apple 3 in/on the sidetable 1.

Prompt 10: rq3-Optimal

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.
> think: To solve the task, I need to find and take an apple, then put it in sidetable.
OK.
> 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 garbagecan 1.
OK.
> go to garbagecan 1
On the garbagecan 1, you see a apple 3, and a egg 3.
> think: Now I find an apple (3). Next, I need to take it.
> take apple 3 from garbagecan 1
You pick up the apple 3 from the garbagecan 1.
> think: Now I find an apple (3). Next, I need to put it in/on sidetable 1.
OK.
> go to sidetable 1
On the sidetable 1, you see a cup 1, a lettuce 1, a peppershaker 3, a potato 1, and a saltshaker 1.
> put apple 3 in/on sidetable 1
You put the apple 3 in/on the sidetable 1.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification:

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Limitations present at the end of the paper.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

Justification:

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification:

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification:

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification:

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Experimentation is consistent with previously accepted NeurIPS papers.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.

- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification:

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification:

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Discussed at the end of the paper.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to

generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.

- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification:

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification:

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification:

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification:

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification:

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.